PROCEEDINGS OF

REBNITA 2005

1ST INTERNATIONAL WORKSHOP ON REQUIREMENTS ENGINEERING FOR BUSINESS NEED AND IT ALIGNMENT

29-30 AUGUST 2005
THE SORBONNE, PARIS at RE'05

NATIONAL
ICT AUSTRALIA
LIMITED

REBNITA is supported by National ICT Australia Ltd
www.nicta.com.au

Editors: Karl Cox, Eric Dubois, Yves Pigneur, Steven J. Bleistein, June Verner, Alan M. Davis, Roel Wieringa

ISBN: 0 7334 2276 4
University of New South Wales Press

Welcome to the REBNITA proceedings!

REBNITA – the 1st International Workshop on Requirements Engineering for Business Need and IT Alignment – was run at the Sorbonne, Paris, on 29-30 August 2005, as part of the 13th IEEE International Conference on Requirements Engineering.

It is no longer possible to consider IT separate from the business organization it supports, and hence requirements engineering should address the business needs of an organization. Business needs can be described through IT alignment with business strategy, including alignment, explicit value analysis of IT, integrated market analysis and product development, as well other types of analysis of business processes, organisational infrastructures, business goals and objectives. Though it is recognised that requirements engineering (RE) is a natural bridge that connects the business world and the IT world, much of RE research continues to be solution-oriented and avoids addressing the hard, real-world business problems that confront business practitioners every day. This trend, if continued unchecked, threatens to ultimately make requirements engineering research of little relevance or importance to industry. As such, the goal of this workshop is to provide a specific forum for research that is motivated by requirements engineering approaches that encompass organizational business needs.

Objectives

1. To promote the connection between business needs and requirements engineering.
2. To investigate and develop new approaches for meeting business need.
3. To empirically evaluate existing approaches in industrial settings.
4. To bring together a diverse audience who recognise the need to apply requirements engineering research on real problems and set an agenda for the future of this field.

With 40-50 attendees, 19 paper presentations, lively discussions, an opening, enthralling keynote from Peter Reynolds, General Manager of Commonwealth Bank Australia, all set in the backdrop of an amazing lecture theatre in the Sorbonne, REBNITA went a long way to achieving these objectives! Let's do it all again next year!

All papers were peer-reviewed by at least 3 members of the internationally renowned programme committee we put together for REBNITA. A full and standard review process and paper selection took place.

We thank all our programme committee, the organizing committee, all at RE'05, especially Camille Salinesi and Anne Etien, who helped get us going and then kept us going, all the volunteers at RE'05 for helping us keep sane, all authors for submitting and especially everyone who came along to attend. We hope you enjoyed it as much as we did.

Organizing Committee

Karl Cox, National ICT Australia Ltd., karl.cox@nicta.com.au
Eric Dubois, Public Research Centre Henri Tudor, Luxembourg, edu@info.fundp.ac.be
Yves Pigneur, HEC Lausanne, Switzerland, yves.pigneur@unil.ch
June Verner, National ICT Australia Ltd., june.verner@nicta.com.au
Steven Bleistein, National ICT Australia Ltd., steven.bleistein@nicta.com.au
Alan M. Davis, University of Colorado, adavis@uccs.edu
Roel Wieringa, University of Twente, R.J.Wieringa@ewi.utwente.nl

Program Committee

REBNITA web site: http://homepage.mac.com/karlalancox/REBN2005.htm

Contents

A Fact Based Collaboration Modeling and its Application

Atsushi KOKUNE, Masuhiro MIZUNO, Kyoichi KADOYA and Shuichiro YAMAMOTO
Research and Development Headquarter, NTT DATA CORPORATION
e-mail: {kokunea, mizunomsh, kadoyak, yamomotosui } @nttdata.co.jp

Abstract

The Balanced Scorecard has received wide attention as a management technique of modeling enterprise strategy. But there is a problem that no method is developed for evaluating the model. We propose a fact based collaboration modeling methodology. Based on facts and data of business processes, it enables business persons to develop enterprise strategic model from the viewpoints of collaboration between organizations. This paper describes a basic concept and a procedure of the methodology. We also show the case study to develop an SCM strategy of a Japanese automobile enterprise. The research project was conducted in seven months to develop the strategy for a complete cars' logistics process among five different departments of the company. The result shows the effectiveness of the proposed methodology.

1. Introduction

When examining the validity, especially completeness, of software requirements, it is necessary to check if software function requirements are consistent with management goals and business processes. The software function requirements can be examined by the strategy model generated by modeling the management strategy and the goal of software development. Therefore, ensuring the validity of the strategy model itself is essential (see also Figure 1).

Against this background, a modeling methods such as Balanced Scorecard (BSC) [1][2][3] are used to develop enterprise strategies. Some methods include a repetition of interviews of stakeholders and reviews.

However, methods to verify the strategies from the view points of real business fields and data haven't been developed, and it results in the failures to define software function requirements that meet the management strategy and the goal of software development.

In this paper, we propose the Fact Based Collaboration Modeling methodology. Proposed methodology is a technological approach to defining non-functional requirements that are used to set the business goal. We also show the effectiveness of the methodology based on a case study of developing a business strategy of an actual Japanese automotive company.

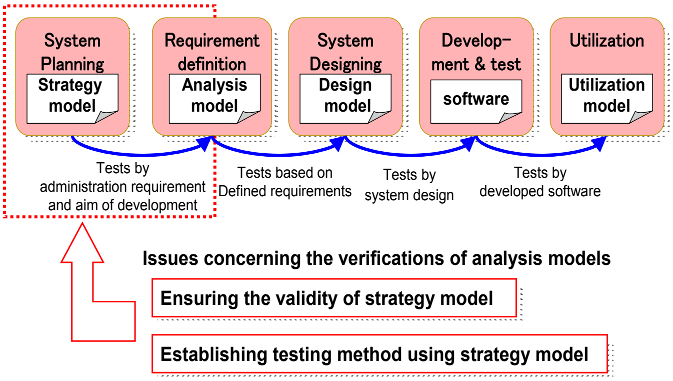

System Planning
Strategy model
Requirement definition
Analysis model
System Designing
Design model
Develop-ment & test
software
Utilization
Utilization model
Tests by administration requirement and aim of development
Tests based on Defined requirements
Tests by system design
Tests by developed software
Issues concerning the verifications of analysis models
Ensuring the validity of strategy model
Establishing testing method using strategy model

Figure 1. Testing models in software development

2. Fact Based Collaboration Modeling (FBCM)

2.1 Elements of Strategy

The strategic model consists of the following two typical elements.

- Strategic goals: These business goals are extracted based on enterprise goals and its vision, such as the perspectives and environments of the enterprise.
- Key performance indicator (KPI): Each KPI is used to measure the degree of achieving the specific goal. Indicators are assigned to every strategic goal.

A strategy consists of at least one strategic goal and multiple KPI indicators that correspond to the goals.

2.2 Modeling steps

The methodology uses the fact information of the business fields and the field data that is stored as a result of business processes in the business fields.

The FBCM consists of four steps as shown in Figure 2. STEP 1 and 3 are the same as BSC methods. The FBCM extends the BSC in STEP 2 and 4 to utilize the field facts and KPI data.

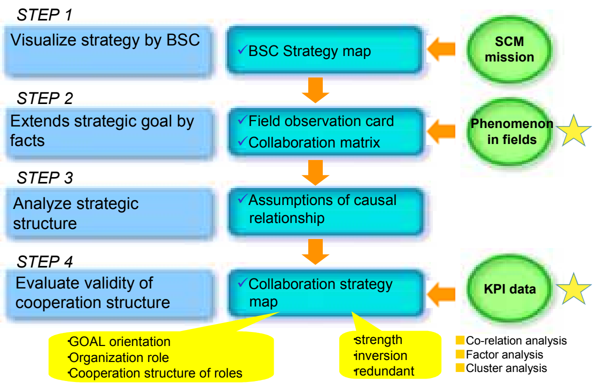


Figure 2. Steps on FBCM

2.2.1 Visualization of strategy

Strategy developers describe a strategy as a BSC by extracting strategic goals from existing papers such as enterprise annual papers and action plans. They also make interviews to stakeholders for reviewing the strategy.

2.2.2 Elicitation of strategic goals by facts

Strategy developers make observations in the actual business fields and elicit additional goals to the strategic model in BSC.

The problem of STEP 1 is that strategic goals are only extracted from vision document without consideration on the real business fields.

In the FBCM, the developers extend strategic goals by facts that are occurred in business fields. If the developers use these facts, they could add supplemental goals that have real business values. In addition, the strategy model could become close to the actual situation in the business fields.

To collect and analyze the facts that are occurred in the business fields, the developers use the "Field Observation Card" and the "Collaboration Matrix".

The Field Observation Card has seven elements to be filled:

1) Fact name: a unique name for the phenomenon occurred in the field

2) Phenomenon: detailed content of the phenomenon

3) Countermeasure: how to handle the phenomenon in the fields

4) Opportunity: solution and things to be improved

5) Critical Success Factor (CSF): the bottlenecks among people to be solved in order to prevent the phenomenon,

6) Key Performance Indicator (KPI): possible KPI that shows whether the phenomenon is occurred or not, or how much the phenomenon is occurred

7) Organization: the organization(s) that could improve the KPI value

Next, the developers categorize collected information in the collaboration matrix. The collaboration matrix is a 3 x 4 matrix. Vertical three columns are for collaboration:

- Collaboration goal: whether all organizations share the purpose and goal for the collaboration
- Role of the organizations: whether the organizations understand their expertise mutually
- Collaboration structure among the organizations: whether each organization understands the whole situation, and acts for problems in a timely manner

Horizontal four rows are the following kinds of bottlenecks:

- Bottleneck between management and business process
- Bottleneck between business processes
- Bottleneck between business process and IT (Information Technology) systems
- Bottleneck between IT systems

Phenomena written in each field observation card are classified into twelve categories on the 3x4-collaboration matrix. The developers confirm the validity of strategic goals and consider additional strategic goals based on the categorized field observation cards on the collaboration matrix.

2.2.3 Strategy structure analysis

The developers make causal relationships between strategic goals. They also assign the KPI(s) for each strategic goal. This is a normal step in creating a BSC.

2.2.4 Verification of strategy structure

The developers evaluate the validity of the causal relationships between strategic goals by analyzing stored KPI data on business processes statistically. They analyze the relationships from the following three viewpoints:

- Co-relation analysis: The developers evaluate the validity of the causal relationship between strategic goals by co-relation of KPIs.
- Factor Analysis: For the strategic goals which have multiple possible KPIs, the developers decide the critical KPI(s) by factor analysis.
- Cluster analysis: For too many KPIs, the developers decide which they should take either the approach using average for the KPIs or the approach doing cluster analysis.

Through the analysis, the developers evaluate the causal relationship between strategic goals, and the validity of KPIs assigned to each goal.

In the BSC strategy map, strategic goals are illustrated from four perspectives: the financial, the customer, business process, and learning and growth perspective. BSC strategy map, however, does not show the relationship between strategic goals and the representative organizations definitely. We have developed a "collaboration strategy map" to resolve the problem. The collaboration strategy map uses the following perspectives:

- Whole organization: It is filled shared strategic goals for all organizations concerned.

- Inter-organizations: It is filled shared strategic goals for intercommunicating organizations.
- Intra-organization: It is filled strategic goals for the specific organization.
- Common: It is filled common strategic goals regarding to employee's basic strategic goals regarding to learning and skill-up.

Strategic goals on the BSC strategy map are remapped onto the collaboration strategy map from these four perspectives.

In the collaboration strategy map, it can be definitely described which organization is responsible for what goals because the map is filled each organization's strategic goals. Collaboration between organizations can be facilitated because it is illustrated what strategic goals are achieved among interfered organizations on the map. The map enables the developers to find clearly that strategic goals for collaboration are not set.

2.3 KPI library

A method for deciding KPIs affects the model quality and the cost for modeling. The FBCM offers KPI library for developers to choose KPIs easily.

The KPI library has about 700 KPIs. These are categorized by four perspectives of the BSC, and subdivided according to the following concepts:

a) The financial perspective: 72 KPIs are categorized by general financial properties such as stability and growth potential.

b) The customer perspective: 60 KPIs are categorized by four categories such as market-level and customer-level.

c) The business process perspective: 580 KPIs are classified by the process categories according to APQC standardized processes.

d) The Learning and Growth Perspective: There are 82 KPIs which are categorized by proprietary categories by personal and organizational learning.

This library has a feature that it is distinguished KPIs strongly relating to collaboration into the others. There are 250 KPIs relating to collaboration for all 700 KPIs.

3. Experiment and evaluation

We have applied the FBCM to develop a strategic model for a complete cars' logistics process in a Japanese actual automobile company. The business process includes the following organizations cooperate with each other:

- Production organization: factories in which complete cars are assembled
- Logistics organization: organizations that prepare transportation of complete cars
- Maintenance organization: organizations that adjust the cars and install options before shipment
- Logistics bases: bases in which the cars are load onto trailers or ships
- Stores: to which customers visit
- SCM organization: the business planning organization for this modeling project

Although many KPIs had been set in the company, it had been difficult for the company to assess the effectiveness of KPIs for their business processes. We modeled the strategy based on the FBCM for seven months from December 2003 to June 2004, and wrote a report 160 pages long.

3.1 Visualization of strategic goals

We selected ten strategic goals from the publicly available brochures of medium-range economic planning and policy materials used in those organizations, had some interviews from people concerned, and developed the strategy as a BSC strategy map.

3.2 Elicitation of strategic goals by facts

We had some interviews from field workers about their business processes and the objective at their fields. We also observed their situation and what they worked, and recorded some problems and characteristic actions onto the 60 field observation cards. We elicited four strategic goals by using the collaboration matrix.

3.3 Strategy structure analysis

We had some interviews from members of SCM organization, defined the causal relationships between strategic goals, and assigned KPIs to each strategic goal (Figure 3).

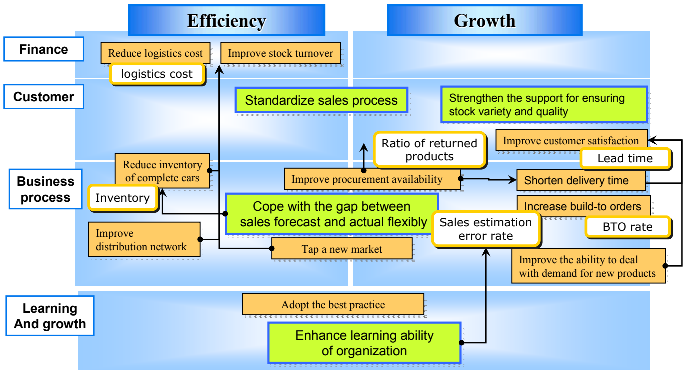

Efficiency
Growth
Finance
Customer
Business process
Learning And growth
Reduce logistics cost
logistics cost
Improve stock turnover
Standardize sales process
Strengthen the support for ensuring stock variety and quality
Ratio of returned products
Improve customer satisfaction
Lead time
Reduce inventory of complete cars
Inventory
Improve procurement availability
Shorten delivery time
Cope with the gap between sales forecast and actual flexibly
Sales estimation error rate
Increase build-to orders
BTO rate
Improve distribution network
Tap a new market
Improve the ability to deal with demand for new products
Adopt the best practice
Enhance learning ability of organization

Figure. 3. Finding causal relationships between strategic goals and allocating KPI for each goal

3.4 Evaluation for the validity of strategy structure

We performed statistical analysis of data of 30 KPIs that were accumulated for about one year, and found that the validity of the following three types of relationships between KPIs must be confirmed by the correlation analysis:

1) Causal relationships between KPIs with especially strong correlations. For example, the increase of build-to-orders led to large difference between sales forecast and actual sales. In addition, this difference causes the increase of distribution cost.

2) Relationships between KPIs that are not correlated with each other strongly enough to decide that their relationships are causal. The correlation between the error of sales forecast and the amount of inventory was not always strong.

3) Reversed causal relationships between KPIs. We had assumed that the rise in the ratio of returned products extends lead time. However, the data analysis revealed that the extension of lead time increases the ratio of returned products about two months later.

We examined the above results on the basis of our experiences and the actual situations of business processes, and modified the BSC strategy map.

However, the BSC strategy map has a weak point: it doesn't give a clear picture about which organization is responsible for achieving what goals. In the FBCM, the collaboration strategy map based on the concept of business collaboration is developed. The map has four layers:

1) The first layer shows the final goals of all organizations.

2) The second layer shows goals which cannot be achieved without the cooperation between multiple organizations.

3) The third layer shows the goals of individual organization.

4) The bottom layer shows the fundamental goals that each person and organization must have in common.

Figure 4 is the collaboration strategy map developed in this case study. It shows that some goals are shared by multiple divisions. For example, the goal, "Improving product quality in logistics process", is shared by logistics and sales divisions. On the other hand, the production and procurement division don't have any shared goals, as indicated by the dotted rectangle (A) in the figure. This shows that strategic goals are not enough for facilitating collaborations between organizations. We pointed out that further strategic goals should be defined between production and procurement division.

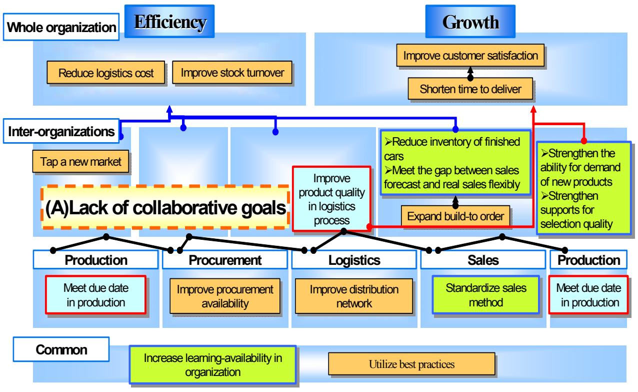

Whole organization
Efficiency
Growth
Reduce logistics cost
Improve stock turnover
Improve customer satisfaction
Shorten time to deliver
Inter-organizations
Tap a new market
(A)Lack of collaborative goals
Improve product quality in logistics process
➢Reduce inventory of finished cars
➢Meet the gap between sales forecast and real sales flexibly
Expand build-to order
➢Strengthen the ability for demand of new products
➢Strengthen supports for selection quality
Production
Procurement
Logistics
Sales
Production
Meet due date in production
Improve procurement availability
Improve distribution network
Standardize sales method
Meet due date in production
Common
Increase learning-availability in organization
Utilize best practices

Figure 4. Collaboration Strategy Map

3.5 Evaluation result

We analyzed the strategy for complete cars' logistics process that is operated by five organizations.

At first, we picked out ten strategic goals from existing materials and interviews from people concerned.

Next, we observed actual fields, made 60 field observation cards, analyzed them, and could extract five strategic goals, especially for the customer perspective on the BSC.

Having defined 11 causal relationships, we analyzed one-year 30 kinds of KPI data statistically. Finally, we could point out that three causal relationships should be confirmed their validity.

The company to be applied the FBCM valued the following three points.

The first point is that the whole strategy for complete cars' logistics process was visualized than they have ever had. They had been trying to work out how to deal with visualization for the strategy among multiple organizations because they had developed the strategy only for each organization. The strategy map enabled them to understand their whole strategy at a glance.

The second is that the map made the relationships among KPIs clear. They had made data analysis for KPIs, but they had just broken the KPIs into parts, and hadn't made co-relation analysis among the KPIs. They valued that they confirmed the wider possibility to be found by the co-relation.

The last value is that they could develop the strategy based on the real situation in the business fields. The strategy should be developed after they understand the situation in the fields and doesn't make sense if the strategy isn't put into practice. In many cases for strategy development, people far from the fields develop the strategy, so the strategy involves the risk that the strategy is widely different from the field situation. The FBCM incorporates the field observation to enforce possible strategy and evaluate the validity of it. Therefore the FBCM enables users to develop their strategy which reflects the field situation.

4. Consideration

4.1 Effectiveness for observing field occurrence

Developing a BSC map lets users to point out that there could be problems for over-and-short strategic goals or unbalanced them. The users, however, cannot guess the lack of specific strategic goals.

FBCM enables the users to propose strategic goals that are guessed from existing materials or interviews for people concerned by classifying field observation information.

Using the field observation cards to develop the strategy is useful for strategy taking into account both business and the fields.

In addition, the users can develop specific and practicable polities because they consider counter-measures for a problem on every field observation card.

4.2. Effectiveness for analyzing business track records

Users could determine causal relationship based on their experiences, and that deliverable varies widely depending on personal skill. Determined causal relationships, however, are very strategy structure, and concerned organizations review their business processes and their systems based on the relationships, so it is required high rationality to determine the causal relationship.

The FBCM enables the users to evaluate the validity of causal relationships between strategic goals by utilizing statistic analysis for business track record (KPI data).

It was too difficult to decide KPI data for strategic goals because there were too many possible KPIs. However, it is effective to use results of statistically analyzed KPI data, evaluate the availability as KPIs, and provide information for choosing and deleting KPIs. We can satisfy the requirements for choosing KPIs easily.

4.3. Effectiveness for the collaboration strategy map

BSC strategy map doesn't show which organization has the responsible for what strategic goals explicitly. On the other hand, collaboration strategy map shows that each strategic goal for each organization and goals to be achieved among multiple organizations are described in the structured manner. So the map solves the problem that strategy isn't carried into action. The users can also find organizations for unclear strategic goals, and confirm the common strategic goals. Therefore the collaboration strategy map can improve the validity of whole strategy.

4.4. Limitation in FBCM

The FBCM is available for any types of business and for any scale of business. Note that the FBCM works strongly on the situation in which multiple organizations aim at the global optimum collaborating each other such as SCM.

For BPR and IT development, it is need to clarify which parts of business processes and how must be changed. The idea of FBCM is just to clarify objectives and goals, and how to measure their achievement. Therefore, it is need to collaborate with business process modeling.

5. Related works

The i* Framework[4] describes a Strategic Dependency model among goals, actors and tasks. The Goal-Exception-Dependency framework [5] allows the qualitative reasoning needed for process redesign based on goal/exception and dependency diagrams.

Seddon et al. [6] developed the model for the virtual factory approach that incorporated simulating the flow of material integrated with the flow of information through business processes.

Jain et al. [7] proposed the model to simulate the flow of material integrated with the flow of information on the virtual factory.

Kavakl [8] provides guidelines to producing customized goal modeling methods. These guidelines describe knowledge regarding the situations under which a method fragment is applicable.

Brock et al. [9] showed a balanced approach to IT project management.

List and Machaczek [10] proposed a performance measurement system based on balanced scorecard. They also developed the performance data model for the performance measurement of the organization.

6. Summary and future issues

In this paper, we proposed the FBCM, and described its effectiveness and issues based on the case study to actual business activities.

Future issues include that it is necessary to integrate the proposed strategic modeling with business process modeling method. And it is also needed to evaluate performance measurement of the strategy using the FBCM.

7. References

[1] Paul R. Niven: Balanced Scorecard Step-by-Step: Maximizing Performance and Maintaining Results, ISBN 0-471-07872-7.

[2] Robert S. Kaplan, and David P. Norton: The Strategy-Focused Organization: How Balanced Scorecard Companies Thrive in the New Business Environment, ISBN 1-57851-250-6.

[3] Robert S. Kaplan and David P. Norton: Strategy Maps: Converting Intangible Assets into Tangible Outcomes, ISBN 1-59139-134-2, 2004/02.

[4] S. K. Yu and Mylopoulos, J., Enterprise Modelling

for Business Redesign: the i* Framework, SIGGROUP Bulletin, vol. 18, No. 1, 1997.

[5] Katzenstein, G. and Lerch, J., Beneath the Surface of Organizational Processes: A Social Representation Framework for Business Process Redesign, ACM trans. On Information systems, Vol.18, No.4, pp.383-422, 2000.

[6] Seddon, P., Graeser, V. and Willcocks, L.: Measuring Organizational IS Effectiveness: An Overview and Update of Senior Management Perspectives, The Data Base for Advances in Information Systems, vol. 33, No. 2, pp.11-28, 2002.

[7] Jain, S. and Ngai Fong Choong: MODELING COMPUTER ASSEMBLY OPERATIONS FOR SUPPLY CHAIN INTEGRATION, Proceedings of the 2002 Winter Simulation Conference, pp.1165-1173.

[8] Kavakli, E.: Modeling organizational goals: Analysis of current methods, ACM Symposium on Applied Computing, pp.1339-1343, 2004.

[9] BROCK, S., HENDRICKS, D., LINNELL, S. and SMITH, D.: A Balanced Approach to IT Project Management, Proceedings of SAICSIT 2003, Pages 2 –10.

[10] Beate List, Karl Machaczek: Organizational engineering (OE): Towards a Corporate Performance Measurement System, Proceedings of the 2004 ACM symposium on Applied computing, pp.1344-1350, 2004.

Adapting Business Processes to Varying Business Needs: A Case-Based Analysis

Klaus Schmid, Tom Koenig

Fraunhofer IESE
Sauerwiesen 6, 67661 Kaiserslautern, Germany
{klaus.schmid, tom.koenig}@iese.fraunhofer.de

Abstract

Many organizations successfully develop systems within a certain, limited domain like inventory management. While the resulting systems share a lot of commonality, they may still differ significantly as individual customers may have strongly varying business needs.

In this paper we discuss the issue of representing these business needs from the point of view of the adaptation of systems. Thus, we focus on the question: how should we represent business demands and requirements that make a difference for systems within a certain domain?

We will discuss different approaches, like decision-based variability modeling and domain-specific languages that support an efficient and effective adaptation of systems to varying business needs. As, with regards to these approaches, we identified business processes as a key ingredient to the definition and adaptation of systems, the paper will especially focus on an analysis of the customization potential of business process notations and suggest a categorization of the underlying business process vocabulary.

1 Introduction

The development of systems within a limited domain, like for example inventory management, usually results in systems sharing a lot of commonality at the one hand but differing significantly with regards to varying business needs across individual customers at the other hand:

- the business models may differ, e.g., one company may focus on cost leadership, while another aims at a lead in quality
- the basic products or services offered may differ, and so forth.

In order to satisfy their customers' business needs, these software development organizations must efficiently adapt the software solutions to the specific needs of their clients. Thus, we are taking here the perspective of specifying the *delta* in terms of business needs (among different customers) and mapping this to a *delta* in terms of the requirements. In particular, we are asking: how can we most efficiently deal with such a situation (i.e., the system adaptation) and how can we ensure a high degree of effectiveness by providing means with adequate expressiveness to specify the mentioned *delta*. The expressiveness here refers to the type of needs, i.e. the complexity that can be expressed using a certain notation or notational element.

The expressiveness and ease of use of the adaptation is particular relevant as we see the adaptation of systems to changing business needs as a typical scenario.

The work we are presenting here is driven from several real-world case studies in which we were involved over recent years. Due to the specific context of the case studies, we will focus on information systems in the broad sense. Thus, we are including such systems as: inventory management systems, IT infrastructure systems, content management systems, etc. Even though these systems are significantly different, they have one thing in common: as a key ingredient to their definition – and thus to their adaptation – they must support business processes.

The main contribution of this paper is to present different approaches for supporting the customization of information systems for varying business needs. As the different case studies mentioned above showed that business processes were a key factor to define and adapt the respective systems, we will in particular analyze the customization potential of business process description languages and will propose a categorization of the business process vocabulary in order to differentiate among different forms of customization problems. The underlying approach is illustrated with several real-world examples.

2 Construction Time Adaptation of Business Process Systems

Different approaches to the adaptation of business information systems can be imagined and are actually used in practice. We will summarize them as:

- Business process modification
- Product line approaches
- Domain-specific languages

We will now discuss each of these approaches in turn.

2.1 Business Process Modification

The most straight-forward approach to adapting a system to changing business needs is of course to identify new business processes that are able to satisfy the underlying needs. These business processes are then implemented. This implementation can take several forms. The most prominent ones are: direct implementation and business process languages.

Currently, there is a strong emphasis – especially in the context of web services – on business process languages, most notably BPEL [1] and BPML [2], BPMN [3]. Modern tools like WebSphere Business Integration Modeler [4] or Oracle Business Process Manager [5] support this approach by providing specialized development environments.

While this approach seems at first sight rather elegant, it is actually the most cumbersome, as it does not provide any support in addressing the relation between business needs and solutions (modified business processes). Moreover, it does not support any direct reuse of this knowledge across different systems (with respect to a product line).

2.2 Product Line Approaches

Product line approaches focus on modeling a set of systems in an integrated manner. As a result, the adaptation of systems – at least to already modeled changes – becomes extremely efficient. The key to product line approaches is the explicit modeling of variability [1, 7]. Many different approaches exist to perform this type of modeling, the principle, however always remains the same: a special model (or modeling element) is introduced in order to represent variation explicitly. This model represents variation through aspects that may be optional (e.g., business processes only exist in certain customizations and do not exist in others) or alternative (customizations may use a specific approach out of possible alternative approaches to realize a certain business goal). The different variation elements have to be resolved in order to instantiate the model for a specific system.

Fundamentally, two major, systematic approaches to modeling product lines from a requirements perspective can be distinguished:

- Feature-based approaches
- Decision-based approaches

Feature-based approaches focus on the specific functionality that is present in some systems – while it is not present in others. On the other hand, decision-based approaches focus on the main decisions that must be made in order to differentiate among different system variants [8, 7]. Hence, the decisions typically embed major business needs that are valid only for a subset of the systems. Thus, the decisions can be regarded as comparable to goals in goal-oriented approaches to representing business needs [9]. As far as we will refer to product line approaches further, we will in particular focus on decision-based approaches.

2.3 Domain-Specific Languages

The key idea of domain-specific languages is to develop a specific language, which is used particularly for the customization of systems within a domain [10, 11]. The restriction to a specific domain allows representing only minimal information in the language. In particular, invariant assumptions and technological aspects are not represented explicitly. In order to define such a language it is important to derive a rather thorough understanding of the domain. This is usually done in terms of a detailed domain analysis [11]. One of the key problems of this approach is that it will usually require rather large investments in order to develop language specific development environments.

While this approach is probably the most elegant in terms of deriving a specific application for a specific set of business needs, it is usually also rarely done, due to the considerable up-front investments. However, from a technical point of view it combines ease of use for the end-user with the capability to stay as close as possible to the concepts that are most relevant and natural to him.

3 Adapting Systems to Varying Business Needs

The context of work that we imply in this paper is not the development of a new system from scratch, but focuses on the adaptation of a system (or the development of a system variant) for different business needs.

According to our experience this scenario is particularly relevant in two different – but conceptually strongly related – contexts:

- Companies often focus on a specific business area in which they build systems. For example a

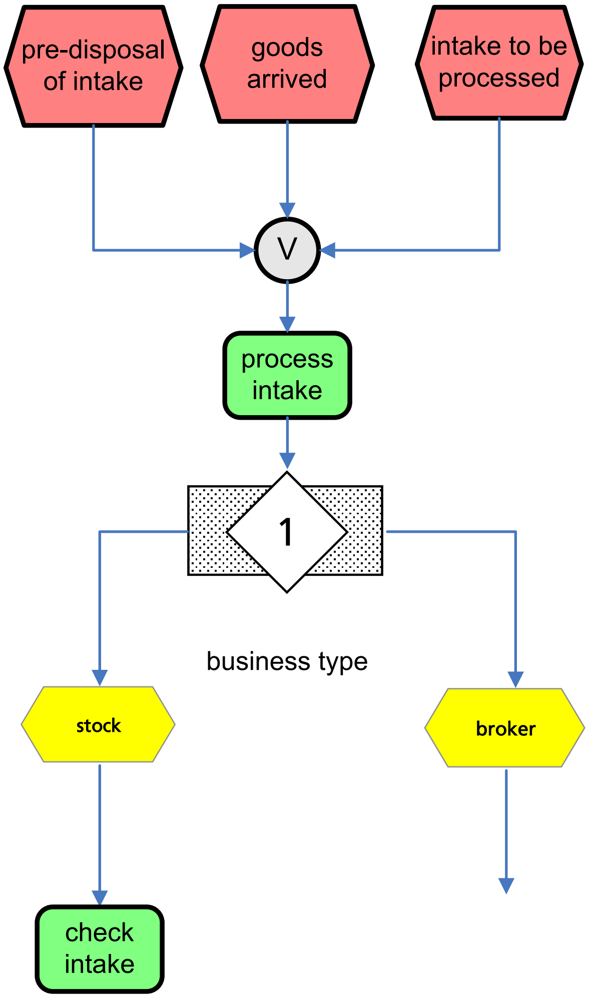


Figure 1 Business process with decision annotations

company that repeatedly builds systems for inventory management will hardly start to build systems for flight booking. This is basically the product line scenario [12].

- Often the customers themselves would prefer to be able to customize their systems to changing needs. Here, part of the variability resolution must actually be possible during runtime – perhaps even to the end-customer.

Both scenarios do strongly overlap. It is essential that the variable business needs are supported in a manner that enables easy and fast adaptation. Especially in the second scenario simplicity of the customization is a key issue as it is strongly related to the overall usability of the final system. Thus, the mapping to the changing business needs a person experiences should be as straightforward as possible. This requires providing a limited – but sufficient – vocabulary for expressing the relevant adaptations.

In the following section, we will discuss how the change in business needs can be adequately represented.

4 Describing Adaptation to Varying Business Needs

Based on the discussion given in Section 2, we can see that different approaches can be used to adapt systems to varying business needs. In particular, we will focus on two approaches in this paper:

- Decision-based product line adaptation for general multi-valued adaptation.
- Using domain-specific languages as a subset of business process languages to describe adaptations that cannot be described as a single decision.

This selection is driven by the aim to provide a customization approach that is as close as possible to adapting business systems in an user-centric manner.

4.1 Decision-Based Customization

As described in Section 2.2, the adaptation of systems based on decision models can be seen as a systematic approach for customization. A typical decision in an information system could be, for example:

- Whether a web-interface should be supported or not – this will be related to the usage context and thus to the business needs relevant to the product.
- Whether car park management should be supported by the system – this is driven from a business need.

These are major, cross-cutting issues in a business system. They do thus reflect variations in business needs. Here, decision-based variability management links an ultimate business decision (i.e., a business need which may be present or absent) with its impact on data models and business processes. In order to describe the impact on the underlying business processes the base models (e.g., the business processes) needs to be annotated by the decision impact. We show an example of this approach in Figure 1. As we do not aim here at a description of variability management, we will not discuss further the details of this approach, but rather refer the reader to descriptions like [7].

4.2 Deriving Domain-Specific Languages for Business Process Systems

While major differences among systems can and should be reflected using a decision-based approach, there are some cases, for which the application of such an approach is not as straight-forward. Examples of such cases are:

- If adaptations are driven by specific situations of the individual customer (e.g., idiosyncrasies of his business environment need to be modeled).
- If variations are too manifold to describe by a simple set of values.

While in these cases one could resort to a general approach based on redefining the business processes, this would contradict our underlying goals of efficiency and usability.

This observation lead us to search for limited sub-classes of expressiveness that enable a more efficient description of the change in business needs, while simultaneously supporting the relevant change.

As a result of our analysis of existing techniques for describing business processes – especially such techniques as BPML, BPEL, etc. – we identified the following classes of primitives:

Basic Primitives – this class contains the most fundamental constructs like assignment, process, etc.

Control – this class contains the constructs relevant to defining the flow control like loops and conditions.

Event-/Error-Management – this contains constructs for managing asynchronous control-flow like event-handling, error- and exception-management.

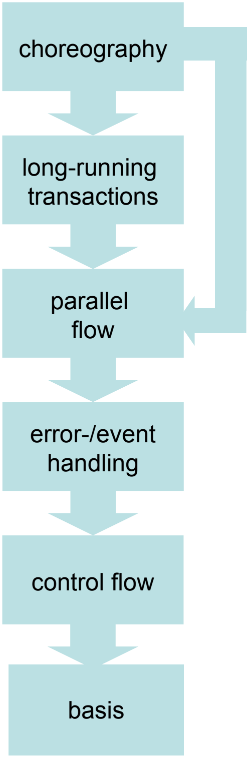


Figure 2 Business Process Primitives

Parallelism – this captures constructs relevant to handling parallel execution. Examples are setting up parallel flows, synchronization of control flow, etc.

Long Transaction – as opposed to short transactions, which can be treated as atomic, long transactions demand for roll-back mechanisms.

Choreography – this aims at managing multiple instances of the same process type.

The different classes can be used to successively extend the expressiveness of the underlying language (cf. Figure 2). Of course one could say that all these classes will be required in every case in order to adequately represent an information system controlled by business processes. However, the major issue here is: while they may be needed to define a system from bottom-up, these distinctions may not be particularly relevant to describe differences among systems in the domain. Thus, characterizing business needs may well be described with a subset. Additional information, which may be relevant to such a system, but does not contribute to differentiate among system instances, will then be realized within the platform.

Depending on the required class of expressiveness alternative modes of representing business processes can be formed – e.g., the combination of base class and control can also be represented by activity diagrams in UML.

In the following section, we will provide some examples that illustrate how these restricted forms of business process language can be used to efficiently represent the business needs – and thus (in combination with the underlying functionality of the infrastructure) provide a means to efficiently adapt systems to changing business needs.

5 Examples

In the preceding sections we discussed the basic concepts of using reduced business process languages as a means to represent the business needs of an application in the form of a domain-specific language. We are now ready to illustrate these concepts in some small examples. The examples are simplified versions of project experiences we made in several industrial cooperations.

5.1 A Simple Language

A rather simple case for specifying business needs for an information system exists in the context of a Content Management System (CMS). CMS are used for managing the flow of documents within an organization, often in the context of web publishing.

Individual documents are usually treated as unrelated. Thus, synchronization mechanisms, etc. are not necessary. This implies, that the document flow can be specified from “a per document” perspective, i.e., if we specify what happens to an individual document, we have specified the overall business processes in such a setting. Basically, we can describe the needs relevant to customizing a specific CMS by the states that documents can assume and the different transitions that are possible. This corresponds to elements in the base class and in the control class of primitives. One approach to representing the necessary information are Statecharts. Thus, we can represent the business needs relevant to customizing the individual systems by describing the state-transitions of documents. Figure 3 gives an example of such a description.

This shows that a subset of expressiveness, as simple as the two basic classes combined and can be actually relevant for describing adaptation-relevant business needs for a system. Thus, in this case a Statechart diagram can be used as a Domain-Specific Language for representing customization-relevant business needs. An example is shown in Figure 3. Actually, a content management company we are working with now aims for this approach to represent the relevant customization needs.

5.2 A Medium Complex Language

The next example stems from the domain of IT-management systems (IMS). If we assume a system which is responsible for managing the IT-infrastructure of an organization, we will need partially the same functionality as in the CMS-example, for example, in order tracking. In addition, we will need to execute for example installation procedures. This might imply that we need to deal with errors or with long transactions (e.g., an installation may actually need several minutes to hours).

Thus, we need a more powerful description language to represent the specific aspects of an organization. We are currently still analyzing this case in order to identify which kind of expressiveness is required exactly and whether there is a restricted – and thus more usable – technique for representing the correlated needs. Aspects that can be used in this domain for defining a simplified representation language:

- Choreography does not seem to be required.
- There will only be one longterm-transaction at a time per request.
- There will be no other actions within a request.

Thus, it might be possible to define a language that does not represent transactions at all. This would then lead to a simple flow language like activity diagrams with an added expressiveness for handling asynchronous events. Due to the more complex situation, a larger range of expressiveness will be needed to capture the business needs of the IMS-customization in a domain-specific language.

5.3 A Complex Language

The next example stems from the domain of Enterprise Resource Planning (ERP). Here, we are working with a company that develops complete ERP systems for warehouses [13].

In this case, all kinds of adaptations are needed for individual customers: whether it relates to how to handle parallel activities, how to handle exceptions, how to handle the choreography of different flows, etc. This is the case because individual activities are usually highly related in this scenario – and customer requirements differ widely.

For example, when a delivery happens, the resulting goods must be unpacked and distributed to fill the needs of different customers. As new packages are ready, trucks are needed, etc. The specific interrelation of individual business processes is then highly customer-specific.

This corresponds to a typical situation where the full range of business process primitives will be needed. Thus, in this case we used a very common and widespread business process notation, namely EPC (Event-driven Process Chains) [14] to specify the requirements with regards to adaptation. EPCs have been developed at the IWi (Institut für Wirtschaftsinformatik Saarbrücken) and are a key component of the ARIS-methodology [15] (Architecture of Integrated Information System). Due the fact that EPCs are a key notational element of the ARIS toolset, developed by IDS-SCHEER, they are very common in industry with regards to specifying business processes. In practice, we also used a decision-based approach to model product line development. However, the obvious difficulty being that it does not directly support the full range of precision required as a basis for adaptations.

6 Conclusions

In this paper, we asked the question “how to represent business needs” from the point of view of an organization that aims at building adaptable systems. This enables us to make our question more precise and take up the issue of how to represent business needs that lead to different systems in a domain. A notion which is of course associated with the key business needs.

What we found were several techniques which were drawn from the wider range of the product line area. We found the approach of domain-specific languages in combination with business process modeling a particularly fruitful approach. A structuring of business process expressiveness provides the primitives that are relevant to the definition of such adapted domain-specific languages.

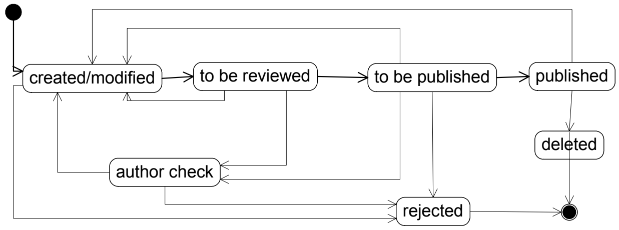


Figure 3 Example Document Flow

Acknowledgements

The work presented in this paper has been partially supported KMU Research Lab funded by Rheinland-Pfalz and the European Fund for Regional Development (EFRE).

We would also like to thank our industrial partners for many valuable discussions and our cooperative work. Their issues and problems provided the basis for our work.

References

1. T. Andrews, F. Curbera et al. Business Process Execution Language for Web Services, 2003.
2. Assaf Arkin. Business Process Modeling Language, BPMI.org, 2002.
3. Stephen A. White. Business Process Modeling Notation, BPMI.org, 2003.
4. IBM, WebSphere Business Integration Modeler Advanced Edition, http://www-306.ibm.com/software/websphere/, 2004.
5. Oracle. Oracle BPEL Process Manager, http://www.oracle.com/technology/products/ias/bpel/index.html, 2004.
6. Jan Bosch. *Design and Use of Software Architectures*, Addison-Wesley, 2000.
7. Klaus Schmid and Isabel John. *A customizable approach to full lifecycle variability management.* Science of Computer Programming. Vol. 53, No. 3, pp. 259-284, 2004.
8. Software Productivity Consortium Services Corporation. *Reuse-Driven Software Processes Guidebook*., Technical Report SPC-92019-CMC, 1993.
9. Steven J. Bleistein, Aybüke Aurum, Karl Cox, and Pradeep K. Ray. *Strategy-Oriented Alignment in Requirements Engineering: Linking Business Strategy to Requirements of e-Business Systems using the SOARE Approach.* Journal of Research and Practice in Information Technology. Vol. 36, No. 4, 2004.
10. Michael Lowry, Jeffrey van Baalen. META-*AMPHION: Synthesis of Efficient Domain-Specific Program Synthesis Systems*. Automated Software Engineering Journal, Vol. 4, No. 2, pp. 199-241, 1997.
11. N. Gupta and L. Jagadeesan and E. Koutsofios and D. Weiss. *Auditdraw: Generating Audits the FAST Way*. Proceedings of the 3rd IEEE International Symposium on Requirements Engineering, 1997.
12. Paul Clements and Linda Northrop. *Software Product Lines*. Addison-Wesley, 2001.
13. Klaus Schmid, Ulrike Becker-Kornstaedt, Peter Knauber and Florian Bernauer. *Introducing a Software Modeling Concept in a Medium-Sized Compaany.* Proceedings of the 22nd Internatioal Conference on Software Engineering (ICSE'22), pp. 558-567, 2000.
14. Keller, G., M. Nüttgens, and A.-W. Scheer. *Semantische Prozessmodellierung auf der Grundlage "Ereignisgesteuerter Prozessketten (EPK)".* Veröffentlichungen des Instituts für Wirtschaftsinformatik, 1992. 89
15. Scheer, A.-W., ARIS - *Vom Geschäftsprozess zum Anwendungssystem*. 3 ed. 1998, Saarbrücken: Springer. 182.

Assessing the Usability of VORD for Web Applications Requirements Engineering - An Industrial Case Study -

Lana S. Al-Salem
The Arab Academy for Banking and Financial Sciences, Amman, Jordan
lalsalem@cti-grp.com

Ala Abu Samaha
Computer Information Systems Department, Amman University, Amman, Jordan
ala_samaha@yahoo.com

Abstract

The Viewpoint-Oriented Requirements Definition method (VORD) as a means of eliciting and formulating requirements has never been applied to Web business applications (WebApps). VORD method is based on assumptions that are partially valid for the Requirements Engineering of e-Commerce, e-Business, e-Banking and e-Government applications. This paper justifies why VORD is chosen for Web Requirements Engineering and evaluates the usability of VORD to elicit and formulate Web application requirements in an industrial case study. The paper includes a discussion of the business strategy impacting the requirements gathering for WebApps. The paper concludes by discussing adapting and extending VORD to suite WebApps.

Keywords

Viewpoint-Oriented Requirements Definition, VORD, Web Applications, Requirements Engineering, Web Requirements Engineering

1. Introduction

Web business applications are a kind of Web information systems (WIS). Such applications tend to be used to integrate and streamline an organisation's business processes across organisational (customers, Agents, suppliers, others) and across geographical borders. Such systems are often vital to the daily operations of the organisation. Hence, any deficiency or system failure can wreak havoc on its business. Therefore, RE is a vital part of the development process of such applications. Yet little attention has been paid to the RE for Web applications (WebApps) compared to other areas, i.e. system modelling, design and coding [6, 16]. There is a pressing need for the RE approaches and techniques that take into account the multiplicity of user profiles and the various stakeholders involved, eliciting overall functionality and the business environment of the WebApp and specifying technical and non-technical requirements [6, 15, 17, 18, 21, 22]. More important is aligning requirements with business strategy [2-4] and meeting business needs. RE can be defined as "the process of discovering that purpose [the purpose for which the software was intended] by identifying stakeholders and their needs and documenting them in a form that is amenable to analysis, communication, and subsequent implementation" [20]. Requirements analysis will remain a vital phase for the development of any application, answering the most fundamental question "what is the system intended for?"[19].

In a previous work [1], WebApps' unique facets were identified in comparison with traditional systems: multidisciplinary development team; state-of-the-art technology; diverse and volatile requirement; vast and unknown end users; multiple stakeholders; short development life cycle; essential quality requirements; heavy content; integration with backend databases and third party applications; adaptable architecture; visibility; and most importantly the WebApps' relevance and direct effect on business.

VORD [14] is considered in solving the problems and issues of WebApps particularities. VORD is applied in an industrial WebApp project; the intent was to test VORD's usability to elicit requirements for WebApps. At the beginning of the project, we didn't take the business strategy into consideration at all. We were going direct to the requirements, and then from our early discussions some questions were formulated that turned out to be of strategic concern such as "what kind of products and services the client wanted to offer?", “what is the client's pricing strategy?”, “what kind of partner network do the client want to formulate?". We could not start with the requirements until we had a clear idea of the strategy issues.

The case study relates to theory-testing approach, of which aim is to examine whether or not reality corresponds to a certain theory, model, method or framework. The paper follows the framework created by Järvinen [10, 11] and builds new theoretical insights from the case study.

The rest of this paper is organised as follows: Section 2 presents an overview of VORD and it justifies the reason behind choosing VORD for eliciting WebApps' requirements. Sections 3 provides an overview of WWT case study. Section 4 illustrates the observations made during the case study. Section 5 presents a general discussion which is related to some issues encountered during the project. Section 6, describes how VORD can be extended to elicit Web Requirements to meet the special characteristics of WebApps and their development process. Finally, the paper concludes with remarks and future work on the Web Requirements Engineering (WRE) method based on VORD.

2. VORD for Web Applications Requirements Engineering

Kotonya and Sommerville [13, 14] proposed VORD as a software RE approach to organise both the elicitation process and the requirements themselves using viewpoints [24]. A key strength in viewpoint-oriented analysis is that it covers the RE process from initial requirements discovery to detailed system modelling [14]. A service-oriented model is adopted for viewpoints; the system delivers services to viewpoints and the viewpoints pass control information and associated parameters to the system. Viewpoints map to classes of system end-users or to other systems interfaced to it [14]. VORD concentrates on three iterative steps, namely [14]: (1) viewpoint identification and structuring, (2) viewpoint documentation, and (3) viewpoint requirements analysis and specification.

2.1 Why VORD can be used for Web Applications Requirements Engineering

VORD is chosen as a candidate WRE method to meet several Web requirements needs. Meeting stakeholders’ needs acts as the most important factor in a project's success; this conviction is the fundamental reason why VORD has been chosen and why there is an attempt to adapt it to fit the WebApps’ facets.

The main reasons for choosing VORD as the reference model for Web Requirement Engineering (WRE) are:

- VORD is a process model designed for highly interactive systems where requirements are mapped to services provided by the system [23].
- VORD aids in the identification of stakeholders and provides separation of concerns [5, 13, 26].
- VORD provides a fairly complete structure for the requirements specification document [14].
- VORD enhances traceability by the explicit association of requirements with the viewpoints from which they are derived [5, 13, 26].
- In VORD, the union of the sets of all the viewpoints’ requirements is likely to be more complete than if the viewpoints have not been identified, and it is more likely that the needs of a diverse set of stakeholders are satisfied [13, 27].
- VORD provides a framework where viewpoints, services, non-functional requirements, and event scenarios can be integrated [13].
- VORD structures non-functional requirements around viewpoints and services. Each service may have associated non-functional requirements; the same service, however, may have different non-functional requirements in different viewpoints.
- VORD recognises that requirements are built gradually over long periods of time and continue to evolve throughout the component's life cycle [14].

It is worth noting that Kotonya [13] has used VORD for a WebApp to demonstrate VORD in a practical experience. However, the special features of WebApps were not taken into consideration. The emphasis was on functional and non-functional requirements. The subtle differences in the nature and life-cycle of Web-based software systems and the way in which they are developed and maintained [6], were all but ignored in Kotonya’s example [13].

3. VORD illustrated by an industrial case study

To assess the usability of VORD in WRE, this section reports parts of a study of requirements for a WebApp for the Management of Entertainment and Sports Events. Some details are omitted from the case study in order to provide clear exposition of the method. The paper focuses on specific parts which will provide enough detail to illustrate the steps taken to formulate the requirements.

3.1 Company background

The case study undertaken by this paper is based on an industrial project for the development of a WebApp targeting the European market. WWT will be used to refer to the WebApp for the Management of Entertainment and Sports Events (the full name of the company is not revealed for confidentiality purposes). The initiator's vision was to have a "state-of-the-art" E-Ticketing application. A paperless Ticket environment, "Tickets of the Future"; CODIQUANT® Cards are to replace the traditional paper ticket. CODIQUANT® Cards are a kind of smart cards; you can write and read from these cards by using a special device that are connected to a PC just like a keyboard where you can transfer data from your PC to the CODIQUANT® Cards through air frequency. WWT's strategy is to be the leaders in introducing such a technology. An E-Ticketing application fully automates the business processes chain, starting with promoters who set up events, venues who are responsible for defining the seating layout and areas, and ending with the actual online sale of tickets to customers. In addition, the WWT itself will administer the system through the WebApp. The WWT application is split into two business models: Business-to-Business (B2B) and Business-to-Consumer (B2C).

The envisioned business cycle for the e-commerce side was as follows: The customer will be able to choose from a large variety of different events categorised according to type, geographical area, etc. WWT will present 3 ways for choosing and booking a seat through the WebApp, ticket outlets (box office, or call centres). Through the WebApp the customer simply clicks on the desired event/ time and date and moves straight to the Venue layout to reserve a seat. When the customer confirms reservation, he/she needs to login to the WebApp, book the seat(s) for an event, pay, and transfer the booking to his/her CODIQUANT® Card.

At the Venue entrance, which is a registered partner and part of the WWT network, the touch free chip on the customer's WWT ticket ensures a swift and easy admission through Intel-gates. These gates are also connected to the internet were before any event it will download all booking details.

The system's capacity to cover the German market alone was initially estimated as follows: Number of Promoters 5,000; Number of Venues 30,000; Number of Customers 1,000,000; Number of Events/ Year 100,000; and Event Size (Attendees) 1 – 200,000. WWT was intended to be extended to cover the rest of Europe starting with Switzerland and Germany.

The objectives for WWT customers are to purchase and receive tickets online 24x7x365; to eliminate the inconvenience of picking up and handling paper tickets; to have a portal presenting all sort of events, to name a few: concerts, exhibitions, cinemas, museums and sport events; and to present detailed content such as directions, parking, hotels, restaurants, and additional offers.

The objectives for WWT Promoters are to receive ticket bookings immediately through online payment, thus eliminating the risk on income lost from insolvent ticket outlets; to create a new marketing channel; to provide added services to both customers and third parties; to minimize tickets counterfeiting; and to take off some of the work load by letting the external organisations setup their events directly into the system

The objectives for WWT venues are to minimize staff; to minimize duplicate or counterfeit tickets; and to increase sales.

The objectives for WWT ticket outlets are to eliminate bank guarantees usually required by Promoters, and to reduce telephone calls cost by booking directly through the internet.

Enhancing customer relationship is a common objective for promoters, venues and WWT; by capturing customer profile, preferences and interests, a direct personal marketing will be created.

The profit for WWT will be achieved by collecting a small charge percentage included in the ticket price paid by the customer, as well as advertisements, and registration fees from venues and promoters.

3.2 Viewpoint identification and structuring

The methods used to identify potential viewpoints with the associated services were: questionnaire, joint application development (JAD), surveying competitors and similar Web sites, and individual semi-structured interviews; as part of the interview process a set of questions were developed specifically to elicit issues of business strategy and vision. The notes from these interviews were recorded and later on analysed by the requirements team. The requirements engineer followed the method of viewpoint identification [14] which involves the following stages:

1. Viewpoint class hierarchies which were not relevant to WWT were excluded.
2. The system stakeholders were considered, i.e. those people who will be affected by the introduction of the system as: Venues, and distribution channels.
3. Viewpoints of three main sub-systems were identified as: a Venue Layout Design Application, a Cinemas Ticketing and Reservation System, and a Payment System.
4. System operators who use the system were identified as: a Web master, a security officer, and an administrator.
5. Indirect viewpoint classes were identified, i.e. legal, and marketing.

Based on this approach, the direct viewpoints developed for WWT are shown in Figure 1 (Promoter, Venue, Distribution Channel, Business Partner, and Customer). Attendee Viewpoints are specialisations of the customer's viewpoint and as such inherit its requirements and attributes. Likewise, the Web master and Marketing & Sales Viewpoints are specialisations of the WWT staff.

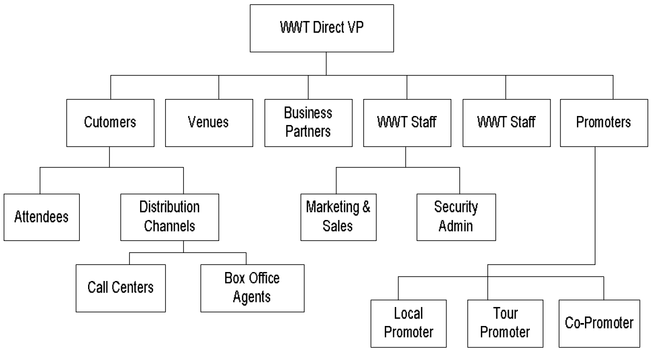


Figure 1. WWT viewpoint hierarchy

A brief description of the major viewpoints is given below:

The Promoter is an organisation responsible for setting up the event and its logistics and organisation in terms of providing the venue, the distribution channels, and the artist/ team/ exhibitor, as well as making sure they all operate in a coordinated manner.

The Venue is the company that owns the facility (building/stadium/grounds, etc) where the event is to take place. The Venue is responsible for defining the seating area of customers, press, and VIPS.

The distribution channels are the Agents that provide the mechanism for the customers (attendees) to select their accommodation areas, complete the reservation transaction (if required for an event), and issue a ticket or pass for the event. The distribution channels are managed by the Promoter on an event-by-event basis. Sub-classes of distribution channels are box offices, reservation Agents, call centres, and the WWT organisation itself.

The customer is the attendee of the event, and makes a booking for an event or has one made by a distribution channel. Special cards are issued for customers that enable them to gain ticketless access to the events. These cards ensure secure data access and verification, and they connect the customers to their personal profiles stored in the system.

4. Observation of VORD usability

Having the VORD method as the basis of requirements elicitation is essential to maintain control. Although a need to adapt and extend the method has been earlier identified, without this initial starting point, there would be no common concept based on which the adaptation can commence. By using the VORD method, the requirements team was able to address a variety of requirements and classify them according to their type in the elicitation step. In addition, the clear steps of VORD guided the RE process: The specification and validation sessions that followed were within a common agreed-on framework which is based on the VORD templates.

Observation 1 - VORD allowed for staying focused and structuring the requirements around the viewpoints.

Viewpoint Name	Customer
Attributes	Customer ID, customer name, gender, birthday, title, address, country, preferred events.
Description	A customer is someone who is interested in buying tickets online and has a profile saved in the system.
Events	-Searching for events -Browsing and selecting events -Logging in -Booking seats -Paying tickets or checking out without payment -Issuing the paid-for tickets -Logging out
Services	-Booking seats for an event -Paying online -Delivering tickets
Sub VPs	Attendees, Distribution Channels

Figure 2: Customers viewpoint template

The Customers viewpoint (VP) had two sub VPs: attendee and distribution channels (Figure 2). The events described in Figure 2 are common for all types of customers. Each sub VP is fully documented later on using a separate VP template. The hierarchy of VPs is reflected on attributes, events and services too. Any unique attribute or event for the sub VP is reflected on its template; same is applied to services where each VP can have its unique constraint or path, i.e. the booking service was intended for the Customers VP but the booking service had a constraint applied to the Attendees VP level. The attendees are not allowed to cancel their booking and get a refund while the distribution channels can.

Observation 2 - VORD was understandable by the clients with minimal explanation, and there was a good interaction in obtaining their ideas and feedback and in verifying the requirements, as VORD offered a common language between the Requirements Engineers and the stakeholders with which both parties were comfortable and familiar.

Observation 3 – The viewpoints notion allowed the client team to focus only on the concerns of their interest. It proved to have a clear cut of separation of concerns. The review and walkthrough phase were separated per viewpoint for each concerned business area.

Observation 4 – VORD provided the requirements team with a framework for formulating very detailed requirements specifications. Viewpoints and their associated services were captured and documented using templates discussed in [23]. The templates were filled iteratively as the Requirement Engineers gained more domain knowledge and captured more requirements. Figure 3 illustrates an example of a service offered to the Customer VP. This template justifies the need for the service; who is using it (VPs); restrictions; and who is going to provide the data for this service. The service details are presented in Figure 4.

Service Name	Booking tickets online
Rational	-To allow customer direct entry to the event without the need to come earlier to pick up the tickets from box offices. -To reduce time & cost for box offices, thus saving money.
Specification	Refer to Spc-B1
Viewpoints	Customers with Sub VPs: Attendees, Distribution Channels.
Non-Functional Requirements	Booking transaction should not exceed 2 minutes.
Provider	-Promoter -Venue

Figure 3: Booking service template

VORD allowed for capturing the details of each service. Figure 4 depicts the details of the "Booking Online" service, the input and output data with their sources, and the pre and post conditions.

Service Specification ID	Spc-B1
Service	Booking
Description	The users will be able to book seats for a certain event through the WWT website.
Inputs	Booking code, booking date, booking done by, customer code, distribution channel code, event code, Venue code, location code, seat number, ticket category, ticket amount, booking download flag, and number of seats.
Source	Event code, Venue code, ticket category, and number of seats are input by the user. Booking code, booking date, booking done by, customer code, distribution channel code, ticket amount, booking download flag, location code, seat number, and gate code are input by the system.
Outputs	The customer booking profile is committed to the database upon the completion of the operation. In addition, the following will be displayed on the screen: booking code, location code, seat number, gate code, ticket price, and total.
Destination	Customer Bookings Profile
Required Fields	Booking code, booking date, booking done by, customer code, distribution channel code, event code, Venue code, location code, seat number, gate code, ticket category, ticket amount, booking download flag, and number of seats.
Pre-condition	The number of bookings should not exceed the number allowed per each transaction, which the Promoter sets for the event. The user must confirm the seats within the time frame specified by the Promoter for that event; if the user exceeds the time frame, the seat will no longer be available. Accordingly, the seats status will change to "Not Sold" again. Provided that the reservation is within the time frame, the seats will be marked as "Temporarily Reserved". Booking can be made by either a distribution channel or an attendee.
Post-condition	The transaction number will be generated automatically for each confirmed booking.
Side-	None

effects

Figure 4: A specification of booking service template

Observation 5 – VORD assisted in creating sequenced services for each class of viewpoint.

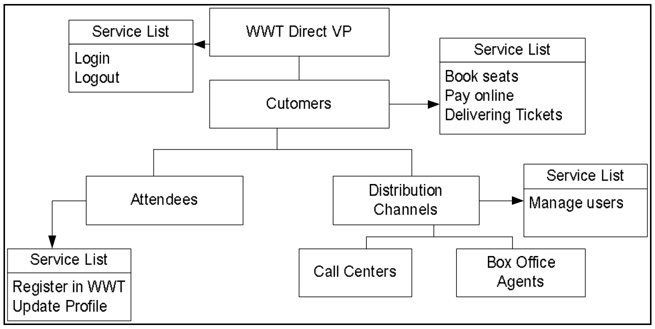


Figure 5: Customer VP structuring

A hierarchy of services is depicted using the VP structure (Figure 5). Services in a higher level will be inherited by the sub VPs.

Observation 6 – The documented VPs exceeded the recommended number of VPs.

Although the number of the direct viewpoints (Figure 1) exceeded the maximum number of VPs suggested in [25], due to the large size of the WebApp under study, it was still easy to classify the requirements according to the VP class.

Observation 7 – VORD lacked the ability to capture the WWT vision and strategy.

Though VORD focuses the RE process on viewpoints services and non-functional requirements, it still lacked capturing explicitly the viewpoints objectives and the organisation's vision and strategy. Many issues were raised that were not related to requirements but to the organisation's strategy; nevertheless the vision and strategy had a direct impact on the requirements.

For example, WWT created a new distribution channel on the e-commerce side, and a new collaboration on the e-business side with Agents, Venues, and Promoters. Questions like: what is the range of products the WWT plans to offer? How often are they going to change the product data and product mix? Who are their targeted customers? How many Venues will be registered with WWT?

Observation 8 – It was not easy to identify the WWT VPs, and to make sure they are relevant.

An extended period of time was spent on identifying and discovering viewpoints (Promoters, Business Partners, Venues, Distribution Channels) as WWT had two business models, B2C and B2B.

Observation 9 – There was resistance to change from the development team.

The development team were mostly familiar with UML and use cases. They doubted that VORD can capture complete requirements as for them it was a new way of collecting and documenting requirements. They were finally convinced when the customer signed the Software Requirement Specifications (SRS) without giving the Requirements Engineers team a hard time. The VORD method is easy to grasp by people. It is in natural language, organised (not a maze), captures details, and concentrates on the services the application will offer.

Observation 10 – VORD lacked a framework for capturing navigation, personalisation or content requirements within the VP details.

The current shape of VORD does not provide a structured elicitation or documentation for navigation, personalisation and content requirements. To overcome the inability to document the navigation requirements, an HTML mock-up of the WWT was prepared.

Observation 11 – VORD lacked the ability to capture the daily business operations and how the WebApp is going to affect them.

During the requirements elicitation process, a lot of questions were raised concerning the way WWT should handle the daily business operations. During the elicitation process, the customer assumed that the business analyst has an answer, where actually most ideas were new. Thus, requirements were sometimes invented rather than elicited; a risky and highly dubious requirements analysis practice that occurs far more often than it should.

Example: How to distribute cards in each country? How to prevent children from buying tickets without their parents' supervision?

Observation 12 – Raising international and legal issues.

International issues and legal issues such as taxation, tariffs, confidentiality and jurisdiction issues relating to users and content, including protection of privacy in addition to illegal and harmful content were raised too during the elicitation process. These issues were new to the client and needed verification from the legal department that also had to search for an answer. How these issues can be documented was not clear.

Observation 13 – Raising marketing issues.

Unlike traditional applications, during the RE phase of the WebApp, stakeholders were wondering about how to attract and sustain customers, and how to encourage more Venues and Promoters to register and display their event within WWT.

5. Discussion

Most of the questions raised by stakeholders at the beginning of the RE phase were more strategic questions than functional, i.e. Who are we going to approach? What ranges of events do we want to offer? What is our pricing strategy? How are we going to retain the customers? How are we going to make the deals with Agents, Promoters & Venues? etc. Such issues had to be finalized first before we even could start thinking of the functionalities the WebApp will offer. This supports Vidgen in his assertion that "WebApps are directly stemmed from and influenced by strategic business vision and goals and they may present new business opportunities" [28].

E-business and e-commerce applications are WebApps that perform business to sell and buy products and/or services on-line. Business interaction means exchange of value, i.e. a product (goods or services) is delivered and payment is made in return [7, 8]. WebApp should directly or indirectly contribute to the value of the customer. Yet, WebApps can also be used for other purposes than direct business interaction. Using WebApps in an organisation must be understood within the organisation's business strategy. The authors claim that a basic fundamental property of WebApps in a business context is its ability to support a business strategy. This is asserted by Bleistein *et al.* who recognise that strategy is critical in requirements analysis of e-business systems [2-4].

Also, as mentioned above, during the requirements elicitation, questions related to the business process and even the business structure were raised like: Which department should be responsible for updating the content? Do we need to create a separate division for the online system or establish a subsidiary company? These questions were directed to the business analysts who questioned their usual role in such WebApps.

WebApps are merging two paradigms: business and Information Systems. On one hand, WebApps are Information Systems where users can perform transactions and operations. As such, the requirements of this aspect of the WebApps are particularly concerned with the traditional requirements functionality of the system. On the other hand, WebApps also support the business and serve organisational strategy, i.e. as a new distribution channel, providing new services or products to customers, etc. Accordingly, WebApps support the organisation's vision and strategy, create a new business model, and change current business processes. From this side, Requirements Engineers should be concerned with decisions about the business vision, strategies and business processes.

E-commerce, e-banking and e-business WebApps are typical examples of the combination of these two paradigms. Blending IS with business poses new challenges for the WRE process. Therefore, requirements alignment with business strategy and anticipating the business processes to be re-engineered from requirements are needed.

Another issue is that WebApps encompass multiple stakeholders and multiple requirements layers (strategic, services, non-functional, content, navigation). There is a challenge to develop a requirement definition approach that would encompass the breadth of requirements and process issues across the organisation and the different levels of stakeholders' requirements within and outside the organisation.

Top management will set the Web business strategy; their input will be directed to the next level of service requirements - what are the services the WebApp will provide that can meet the Web business strategy? Managers of each business unit will evaluate the effect of each service on the business process. Copyrighters and marketing people will set the content for each stakeholder and service provided. End users of WebApps who are normally an external party of the organisation, i.e. customers, suppliers, agents, will set the details of each service.

6. Conclusion and future

The activity underlying the work emphasised the usability of VORD for WRE, primarily due to the focus VORD places on: separation of concern, multi-viewpoints, standardisation and integration of viewpoints, services, non-functional requirements, and event scenarios.

Holck [9] is convinced that the shortcomings of traditional methods should not cause them to be rejected but rather to be enhanced or supplemented with new methods and techniques. Although VORD has a good base to be used for eliciting WebApps requirements, it still cannot be directly applied; it has to be modified and extended to meet the peculiarities of such applications. New enhancements to the method include::

- Need to capture the business strategy and vision of the WebApp; this should be the first step in the process. Blending VORD with Balanced Scorecard [12] could overcome the limitation of VORD in this particular area. Balanced Scorecard is a measurement-based strategic management system, originated by Robert Kaplan and David Norton, which provides a method of aligning business activities to the strategy and monitoring performance of strategic goals over time.

- Need to extend VORD to capture business processes not only automated services.
- Apply new viewpoint taxonomy for WebApps. In addition, the heuristics for identifying Web Viewpoints should be available to assist requirements engineers.
- Utilise scenarios to capture services.
- Create a way to map services to the WebApps Requirements Specifications document.
- Construct a prioritisation system based on the importance of business strategy.
- VORD should cater for new types of Web requirements such as legal, marketing, and privacy issues.
- Capture content, path, user interface, and access for each VP (VP template should be extended and amended to fit these new requirements).

The suggested enhancements are geared towards elicitation and are not primarily intended for requirements modelling or for validation.

7. Acknowledgement

Thanks to Lina Hussein for editing and proof reading the paper.

8. References

[1] L. S. Al-Salem and A. A. Samaha, "Assessing the Usability of VORD Method for Web Applications Requirements Elicitation," presented at International Conference on Internet Technologies and Applications (ITA 05), Wrexham, North Wales, UK, 2005.

[2] S. J. Bleistein, A. Aurum, K. Cox, and P. K. Ray, "Strategy-Oriented Alignment in Requirements Engineering: Linking Business Strategy to Requirements of e-Business Systems using the SOARE Approach," *Journal of Research and Practice in Information Technology*, vol. 36, 2004.

[3] S. J. Bleistein, K. Cox, and J. Verner, "Modeling Business Strategy in E-Business Systems Requirements Engineering," presented at ER Workshops 2004, 2004.

[4] S. J. Bleistein, K. Cox, J. Verner, and K. T. Phalp, "Requirements Engineering for e-Business Advantage," *Requirements Engineering Journal*, 2005.

[5] A. Finkelstein and I. Sommerville, "The Viewpoints FAQ," *Software Engineering Journal*, vol. 11, pp. 2--4, 1996.

[6] A. Ginige and S. Murugesan, "Web Engineering: An Introduction," *IEEE Multimedia*, vol. 8, pp. 14-18, 2001.

[7] J. Gordijn, H. Akkermans, and H. v. Vliet, "Requirements for e-commerce applications are created rather than elicited," presented at Second nordic workshop on software architecture, Sweden, 1999.

[8] J. Gordijn, H. Akkermans, and H. v. Vliet, "Value based requirements creation for electronic commerce applications," presented at 33rd Hawaii International Conference On System Sciences, IEEE, Hawaii, USA, 2000.

[9] J. Holck and T. Clemmensen, "What Makes Web Development Different?," presented at The 24th Information Systems Research Seminar in Scandinavia (IRIS), Ulvik, Norway, 2001.

[10] P. Jarvinen, "On a variety of research output types," presented at IRIS23, Laboratorium for Interaction, University of Trollhättan, Uddevalla, 2000.

[11] P. Jarvinen, "Research Questions Guiding Selection of an Appropriate Research Method," presented at European Conference on Information Systems, Vienna: Vienna University of Economics and Business Administration, 2000.

[12] R. Kaplan and D. Norton, *The Balanced Scorecard: Translating Strategy into Action*. Boston: Harvard Business School Press, 1996.

[13] G. Kotonya, "Practical Experience with Viewpoint-Oriented Requirements Specification," *Requirements Engineering Journal*, vol. 4, pp. 115-133, 1999.

[14] G. Kotonya and I. Sommerville, "Requirements Engineering With Viewpoints," *BCS/IEE Software Engineering Journal*, vol. 11, pp. 5-18, 1996.

[15] D. Lowe, "Web Development Methodologies Help or Hindrance?," *WebNet Journal*, vol. 1, pp. 9 - 10, 1999.

[16] D. Lowe, "Web system requirements: an overview," *Requirements Engineering Journal*, vol. 8, pp. 102–113, 2003.

[17] A. McDonald and R. Welland, "Agile Web Engineering (AWE) Process," Department of Computing Science, University of Glasgow, Scotland, Technical Report TR-2001-98, 2 December 2001 2001.

[18] M. Meldrum and J. Rose, "Activity Based Generation Of Requirements For Web-Based Information Systems: The SSM/ICDT Approach," presented at The 12th European Conference on Information Systems (ECIS 2004), Turku Finland, 2004.

[19] J. Mylopoulos, J. Castro, and M. Kolp, "Tropos: A Framework for Requirements-Driven Software Development," *Lecture Notes in Computer Science. Springer-Verlag*, 2000.

[20] B. Nuseibeh and S. Easterbrook, "Requirements Engineering: A Roadmap," presented at International Conference on Software Engineering (ICSE-2000), Limerick, Ireland, 2000.

[21] O. Pastor, "Current Trends in Web Engineering – The Challenge of New Ideas," Business Briefing: Data Management & Storage Technology 2002.

[22] R. S. Pressman, *Software Engineering: A Practitioner's Approach*, 5th ed: McGraw-Hill, 2000.

[23] I. Sommerville, *Software Engineering*, Fifth Edition ed: Addison Wesley, 1995.

[24] I. Sommerville, *Software Engineering*, Seventh Edition ed: Addison Wesley, 2004.

[25] I. Sommerville and P. Sawyer, "Viewpoints: principles, problems and a practical approach to requirements engineering," *Annual Software Engineering*, vol. 3, pp. 101 - 130, 1997.

[26] I. Sommerville, P. Sawyer, and S. Viller, "PREview: tackling the real concerns of requirements engineering,"

Computing Department, Lancaster University, Lancaster, Technical Report Technical Report CSEG/5/1996, 1996.

[27] I. Sommerville, P. Sawyer, and S. Viller, "Viewpoints for requirements elicitation: a practical approach," presented at IEEE International Conference on Requirements Engineering - ICRE'98, Colorado Springs, Colorado, 1998.

[28] R. Vidgen, D. Avison, J. Wood, and A. Wood-Harper, *Developing Web Information Systems*: Butterworth Heinemann, 2002.

Aligning business process models with specifications using enactable use case tools

John Kanyaru and Keith Phalp
Software Systems Modelling Group
Bournemouth University, UK
{jkanyaru, kphalp}@bournemouth.ac.uk

Abstract

One of the issues in aligning business and IT systems is in mapping business goals and processes through to system specification. This paper presents an approach which uses an enhanced (use case) specification, in order to preserve many of the behavioural features depicted in a process model (in our case using Role Activity Diagrams). The paper argues that such process models often include a 'richness of description', which is lost in moving to a standard specification. Furthermore, that traditional specification approaches typically require a significant formal component to describe such issues.

By providing a simple, mechanism to capture behavioural (dependency) issues we believe that specifications will be able to reflect the business goals (as depicted in the process model), and thus subsequent systems may more closely match this aspect of the business needs.

Furthermore, by providing an enactable version of the specification we enhance stakeholder involvement in the requirements process, and further improve the chances for systems meeting client's requirements.

1. Introduction

Within business process modelling, graphically based models (mainly RADs [1]) are deployed in constructing diagrams of partial processes which are then composed to depict a whole system. The verification of the correctness of such models is sometimes conducted using formal descriptions of the process diagram within an automated environment such as that described in [2], [3], [4] and [5]. Most of these process modelling support tools either model client business processes or the software development process itself [6]. In both cases, benefits are to be found by using the model to highlight real process problems either of clients or developers. However, the modelling of client business processes allows a further opportunity for gain, where the intention is to build a system to provide support for the process being modelled. Although process models inform the requirements process, by providing clarity and understanding at the business modelling stage, the potential of such technology is often lost in the subsequent development phases due to the inclination of these technologies toward formal syntax and semantics. In this paper, we borrow the concept of automated checking of behaviour descriptions from the process modelling community. The objective is to integrate state-based checking of behavioural models through enaction by constructing a light-weight state-machine for use case specification and validation where the need to produce intermediate formal specification is obviated.

The software engineering community has long understood the importance of stakeholder involvement in validation of requirements and specifications [7-9]. Tool support may help to bridge the communication gap between engineers and customers, by providing appropriate models to enhance shared understanding. This paper focuses on providing a mechanism for the transition between business process models to a specification of the supporting software. By deriving behavioural descriptions (with use cases) of the intended software from its associated business process, a seamless transition from the business process to a software specification is provided. Given the inadequacy of use cases to provide a rich picture of process interactions [10], we propose the application of process technology to use cases.

Use cases have gained widespread adoption mainly due to their presentation (with natural language) of system behaviour from the viewpoint of its users. In particular, the use case description details the interaction of external users (actors) with the proposed system. This viewpoint is crucial, especially when validating the adequacy of the specification. However, UML use cases have several shortcomings that curtail their expressiveness in specifying behaviour. Whereas the use of natural language makes use cases easy to construct and understand, it is also a weakness, since natural language specifications can be ambiguous.

The rationale for considering the application of process technology to use case descriptions is twofold. First, despite many years focus on software process, the requirements phase still seems somewhat neglected by practitioners. That is, the single most cited problem for the failure of most software development projects pertains to inadequate requirements determination (e.g. [11]). Second, despite much work on UML use cases, there is little or no focus on enhancing use cases for the explicit specification of dependencies amongst use case interaction steps. That is, whereas use cases describe software behaviour in terms of user-system interactions, there is no explicit way to describe dependencies between constituent events. Moreover, use cases lack sufficient detail [12] and usage guidance ([13, 14]) to be able to support meaningful dialogue between stakeholders.

Thus, despite adopting an interaction-based approach, use case descriptions lack state-based information and do not explicitly include any details about the dependencies of actions (except those at the start and finish of the whole description). [15] suggests that an examination of dependencies in scenarios (and use cases) can be used as a validation mechanism between requirements and specification and would be of use in subsequent design.

In most commonly used process algebras (e.g., [16]), and within business process modelling, it is well appreciated that the flow of logic within processes is not always sequential. That there are points of synchronisation due to parallel threads, processes or events calling each other, or indeed waiting periods involving roles completing tasks in a context-dependent way [1]. Similarly, requirements are not always top-down, transitive entities as often there are a number of actors needed to fulfil a requirement and a number of parallel activities necessary to reach a desired computation result. The UML specification of the use case does not allow for the description of state-based relationships between use case elements. Thus, validation of use case specifications and the described interactions is neither straightforward nor rigorous.

The rest of the paper is organised as follows. Section 2 presents the rationale for moving from business process models to use case specifications. Section 3 discusses an industrial study aimed at demonstrating the movement from RAD models of business processes to use case specifications. The section elaborates on the adoption of a state-based approach to enhance use cases as means to representing interaction issues. Section 4 discusses the advantages of enhancing use cases with state-based information. Section 5 offers some discussion regarding our experiences integrating business technology with use cases and further enhancements arising from the industrial study. Section 6 concludes the paper.

2. Rationale for moving from business process models to use cases

Business process models are representations of activities within a business setting. Thus, despite the rich picture provided by business process models, where software support for certain aspects of the business is required, further specification is necessary.

To ensure the information and knowledge gained from business processes is transferred and represented within the associated specification of software behaviour, it is imperative that the specification notation depicts a rich interaction scheme similar to that of the RAD (or associated description language, e.g.RolEnact). Our focus is on constructing such specifications with textual use cases.

Researchers who have been concerned about the ineffectiveness of use cases to represent stakeholder requirements precisely have advocated the formal specification of use cases with well established languages such as Z (e.g.[17]), Petri Nets (e.g. [18]) or implementations of a process algebra such as CSP (e.g. [19] and [20]). The main argument in these efforts is that a precise statement of what the system is expected to do is important for assuring the consistency and completeness of the specification. We argue however that, ascertaining that a requirements specification is internally consistent, or certified as complete by the requirements engineer does not necessarily guarantee that there is a functional match with the business needs. For example, a specification drafted with the Z language, or LOTOS, might only be understandable by the engineers and not by business stakeholders who directly use the system or pay for it. For this reason, [21] argues for a formalism that enhances communication between engineers and customers. Moreover, [21] postulates that the consideration of requirements interdependencies can help both the domain experts and engineers to clarify hidden issues, both of the domain and expected software.

We argue for an integration of a process modelling flavour of dependency consideration within use cases where constituent events are augmented with state-based information. The state-based information serves to offer a mechanism for explicit consideration of interaction issues amongst use case elements hence ensuring assumptions that can lead to incorrect specification of system behaviour are not made.

There is need to ensure that use case specifications are systematic and unambiguous about dependency issues as a means to validate stakeholder expectations of the specified system early before time and resources are committed to any further development. Additionally, the inclusion of state-based information must be such that the state-information is derived from the business process and that such information does not lead to learning of a different specification language altogether.

3. From RADs to use case specifications: A Case Study

To illustrate and validate our approach, we describe an industrial study involving a company that develops and maintains real time control systems for its clients (companies located in various parts of the UK). The company wanted to establish an on-demand service on server applications to enable easy registration of new customers, addition of new networks for new customers, registration of associated network controllers, and the creation of project roots for these networks and for new customers.

The process of scrutinising new customers, their fixtures, location, and whether it is feasible to offer them support is a preserve of the operations manager, the managing director and the applications manager. These three roles liaise in various ways to authenticate the acquiring and ratification of a new customer. In Figure 1, we consider the generic process depicting interactions between the operations manager (OppsManager), the managing director (MD) and the applications manager (AppsManager) when taking in a new customer:

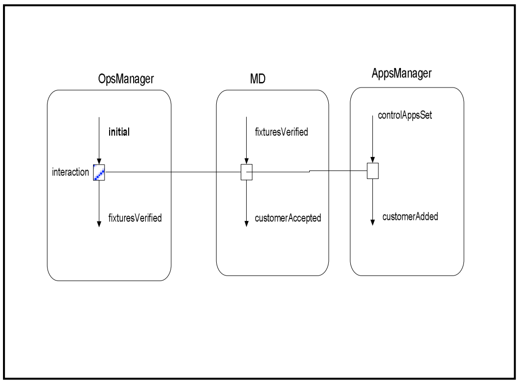


Figure 1: RAD for customer registration process

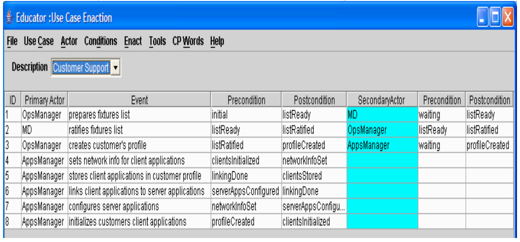
Given the process of registering a new customer for support with the control company's real-time monitoring systems, the task for the authors was to specify the behaviour of a registration process in sufficient detail as articulated by developers (and AppsManager). The specification would then be demonstrated to both the OpsManager and the MD for their scrutiny and ratification. The idea was to specify aspects of the generic process of Figure 1 that can be developed and automated for ease of customer registration.

Our interview with the developers and applications manager yielded the following initial description:

1. Operations manager prepares fixtures list.
2. Managing director ratifies fixtures.
3. Operations manager creates customer's profile.
4. Applications Manager sets up client applications network information.
5. Applications manager stores client applications in customer's profile.
6. Applications manager links client applications to server applications.
7. Applications manager configures server applications.
8. Applications initializes customers client applications.

Figure 2: Registering a new customer

Most authors of use case specifications seldom consider dependencies beyond the typical *include* and *extend* relationships [22]. The presumption normally is that events follow a time-order dependency. That is, event 4 follows from event 3, which follows from event 2, and so on. This was the assumption by the developers when they initially outlined the use case above. Clearly, for simple problems and uncomplicated domains, it is possible to outline the correct sequence of events in a use case like the one above with confidence. Most control systems are however, complex, and the synchronisation of events in a process, or indeed, the synchronisation of processes is usually far from obvious [23]. We deploy our approach and tool support, which requires augmentation of use case events with pre and post conditions to explicitly determine whether the sequence of events in the use case above is correct, and provide a mechanism for revising to correct any flaws. The revised description (states for each event and secondary actors) is shown in Figure 3:



Description: Customer Support

ID	Primary Actor	Event	Precondition	Postcondition	SecondaryActor	Precondition	Postcondition
1	OpsManager	prepares fixtures list	initial	listReady	MD	waiting	listReady
2	MD	ratifies fixtures list	listReady	listRatified	OpsManager	listReady	listRatified
3	OpsManager	creates customer's profile	listRatified	profileCreated	AppsManager	waiting	profileCreated
4	AppsManager	sets network info for client applications	clientsInitialized	networkInfoSet			
5	AppsManager	stores client applications in customer profile	linkingDone	clientsStored			
6	AppsManager	links client applications to server applications	serverAppsConfigured	linkingDone			
7	AppsManager	configures server applications	networkInfoSet	serverAppsConfigu...			
8	AppsManager	initializes customers client applications	profileCreated	clientsInitialized			

Figure 3: State-based use case within Educator

Our state-based approach is supported by enactable functionality within the Educator tool, which means written descriptions can be animated to step-through the description logic.

4. Advantages of the Enhanced Use Case Description

As shown in the RAD of Figure 1, each role has a pre-state and a post-state indicating the states (prior and after) an action is carried out by a role. States determine whether an actor may invoke (or participate in) an event. This means that for an actor to be able to invoke an event, it must be in a state matching that event's precondition, and after the event is successful, the actor changes state to the post-condition of that event. In other words, the textual use case specification is comprised of one central theme, the event, which in turn is accessible to the triggering actor (primary actor) and secondary (passive) actor. A state based use case description need not be written in time-order as it mimics a state-machine whose order of event execution depends on the states of invoked and available events. Thus, default ordering of events is not presumed, as the order of execution is based solely on states. The automated environment, Educator has functionality for including states and their amendment to allow for testing of different combinations of possible behaviours. Stakeholders with differing views on the desirable behaviour can construct different combinations of the description to prototype the implication of different event sequences.

We reproduce the description of Figure 3 to form Figure 4 for the sake of readability (Figure 3 is somewhat small for most readers):

1. **OpManager** prepares fixtures list [initial] listReady] **MD** [waiting] [listReady]
2. **MD** ratifies fixtures[listReady][listRatified] **OpsManager** [listReady][listRatified]
3. **OpsManager** creates customer's profile [listRatified][profileCreated] **AppsManager** [waiting][profileCreated]
4. **AppsManager** sets up client applications network information[clientsInitialized] [networkInfoSet]
5. **AppsManager** stores client applications in customer's profile [linkingDone][clientsStored]
6. **AppsManager** links client applications to server applications [serverAppsConfigured] [linkingDone]
7. **AppsManager** configures server applications [networkInfoSet] [serverAppsConfigured]
8. **AppsManager** initializes customers client applications[profileCreated] [clientsInitialized]

Figure 4: List of events reiterated

(actors are shown in bold print; states are enclosed in square brackets)

The state-based description of Figure 3 (and Figure 4) can be explained as follows: the fixtures list has to be prepared as first step by the OpsManager. The OpsManager is at *initial* state and moves to *listReady* state after completing the list. The essence of this event is to ensure that once the OpsManager has obtained the list of fixtures a customer desires to be monitored and controlled, then the list is encoded in a specific application at the control centre for access by other relevant staff and applications. Given that the MD has to know that the list is ready, there is an interaction between the MD and the OpsManager during the execution of this event. The final authority to accept the customer rests with the MD. The MD then can ratify the list of fixtures as an acceptable quantity and range to support. The essence of states comes out quite clearly after the third event is executed. This is because, after OpsManager creates customer profile", the next available event is not event 4. Instead, the developers and the AppsManager argued that once the customer's profile has been created, the next step must involve the initialisation of the applications required to support that customer's list of fixtures. This step is crucially performed at this point in the description because specific fixtures are associated with specific supporting applications failure to which such a link will not exist. This can lead to incidents of fire or intrusion going undetected on customer sites. Thus, the next event is one whose pre-state is the post state of the previously executed event (that is event 8). This means that the linear dependency scheme implied in the description of Figure 2 is erroneous and misleading. Crucially, these dependencies became obvious during enaction when the state-based information was added to events (see Figure 5 for an example enaction window):

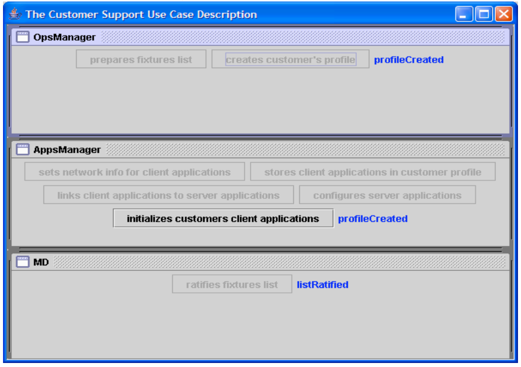

The Customer Support Use Case Description
OpsManager
prepares fixtures list
creates customer's profile
profileCreated
AppsManager
sets network info for client applications
stores client applications in customer profile
links client applications to server applications
configures server applications
initializes customers client applications
profileCreated
MD
ratifies fixtures list
listRatified

Figure 5: Sample Enaction window

By producing enactable use cases derived from parts of business processes, this approach was found to bridge the communication gap between managers who understood high-level process issues, and the software developers who understood low-level concerns. Moreover, the authors were able to easily comprehend the problem domain in terms of the business venture and the inter-play between roles, and also take part in the specification of various software components required to automate parts of the business.

5. Discussion

The prototyping of textual use cases that describe parts of a business process likely to benefit from software support was beneficial in three ways. First, the information forming the use cases was derived directly from the process models, thereby offering a direct mapping of business view to subsequent software behaviour. This enabled informed discussions on aspects of some processes that could not possibly lead to development of software support. Second, for enaction purposes, the use case view was found to be more intuitive than formal process descriptions because the use cases only relied on named states rather than correct and "compilable" programming like constructs of RolEnact. Third, the use of this approach and tool support stretched our initial thoughts regarding the extent to which use case descriptions can be inter-related. For instance, whereas we had envisioned the need to inter-relate distinct use cases so that one description is able to call another, or indeed, the consideration of alternative courses of action, the study indicated several other important relationships. In short, it became imperative that we provide support for descriptions that can be enacted concurrently as some subsystems do perform their computations concurrently. Additionally, it was important to ensure that subsystems which perform similar tasks for different users can be distinguished and chosen as need arise. Given the nature of some applications needing to be executed every so often (at defined frequencies) it was necessary to provide this mechanism too.

Overall, the study has shown that it is feasible to move from business process models to behavioural descriptions of supporting software without the need for any formal specification.

6. Conclusion

In order to meet business needs IT system specification must reflect business goals and process issues. Our experience is that often many important issues are depicted in process models, but that these issues are often lost in moving towards specification. One reason for this is that specifications written as use cases do not offer the mechanisms to depict such issues. One such issue is that of event dependencies

This led to our move to define a systematic scheme for augmenting use cases with state-based information and providing tool support for authoring and enaction. This mechanism also enhances the level of detail in use case specifications, and forces the explicit consideration of dependencies amongst constituent events.

This paper describes an industrial application of this approach – utilising the use case enaction tool. Our experience suggests that the tool scales well to reasonably complex specifications and helps provide a communication medium by which domain experts and requirements engineers can clarify understanding. Furthermore, the tool allows users to reflect the logic of existing process models in the augmented use case specification, and to enhance this understanding by using enaction. By adopting such approaches we believe that specifications will more adequately reflect business needs, both by reflecting business issues and by allowing stakeholder involvement in validation.

7. References

1. Ould, M., *Business Processes: Modelling and Analysis for Reengineering and Improvement*. 1995, Chichester: Wiley.
2. Walters, R., *Checking of models built using a graphically based formal modelling language*. Journal of Systems and Software, 2005. **76**(1).
3. Walters, R.J. *A Graphically Based Language for Constructing, Executing and Analysing Models of Software Systems*. in *26th Annual International Computer Software and Applications Conference*. 2002. Oxford, England.
4. Heimdahl, M.P. and N. Leveson, *Completeness and Consistency Analysis of State-Based Requirements*. IEEE Transactions on Software Engineering, 1996. **22**(6): p. 363–377.
5. Phalp, K., et al., *RolEnact: role-based enactable models of business processes*. Information and software Technology journal, 1998.
6. Finkelstein A, J. Kramer, and B. Nuseibeh, eds. *Software Process Modelling and Technology*. Advanced Software Development Series, ed. R.S.P. Ltd. 1994, Wiley: Somerset, UK.
7. Sutcliffe A and N. Maiden. *Use of Domain Knowledge for Requirements Validation*. in *Proceedings of IFIP WG 8.1 Conference on Information System Development Process*. 1993: Elsevier Science Publishers.
8. Pfleeger, S., *Software engineering: theory and practice*. 2nd edition ed. 2001: Prentice Hall.
9. Leonhardt U, et al. *Decentralised process enactment in a multi-perspective development environment*. in *Proceedings of the 17th international conference on Software engineering*. 1995. Seattle, Washington, United States: ACM Press.
10. Glinz, M. *Problems and Deficiencies of the UML as a Requirements Specification Language*. in *Proceedings of the 10th International Workshop on Software Specification and Design*. 2000. San Diego.
11. Finkelstein A. and J. Dowell, *A Comedy of Errors: the London Ambulance Service case study*. 8th International Workshop on Software Specification & Design IWSSD-8: IEEE CS Press, 1996: p. 2 -- 4.
12. Cockburn, A., *Writing effective Use cases*. 2001: Addison-Wesley.
13. Phalp, K. and K. Cox. *Supporting Communicability with Use Case Guidelines: An Empirical Study*. in *6th International Conference on Empirical Assessment and Evaluation in Software Engineering*. 2002. Keele University, Staffordshire, UK.
14. Maiden N and D. Corrall, *Scenario-driven systems engineering*. 2000, IEE Seminar: London.
15. Sutcliffe, A., *Scenario-Based Requirements Analysis*. Requirements Engineering Journal, 1998. **3**: p. 48-65.
16. Hoare, C.A.R., *Communicating Sequential Processes*. 2004: Prentice Hall.
17. W. Grieskamp and M. Lepper. *Using Use Cases in Executable Z*. in *3rd IEEE International Conference on Formal Engineering Methods*. 2000. York, England.
18. J.B. Jørgensen and C. Bossen. *Executable Use Cases as Links Between Application Domain Requirements and Machine Specifications*. in *3rd International workshop on Scenarios and State Machines (at ICSE04)*. 2004. Edinburgh, Scotland.
19. B. Hnatkowska and Z. Huzar, *Transformation of dynamic aspects of UML models into LOTOS behaviour expressions*. International Journal of Applied Mathematics and Computer Science, 2001. **11**(2): p. 537-556.
20. Harel, D. and R. Marelly, *Capturing and Executing Behavioral Requirements: The Play-In/Play-Out Approach*. 2001, The Weizmann Institute of Science: Rehovot, Israel.
21. Sutcliffe, A. and S. Minocha. *Analysing Socio-Technical System Requirements*. in *4th International Symposium on Requirements Engineering*. 1999. University of Limerick, Ireland.
22. Kulak, D. and E. Guiney, *Use Cases: Requirements in Context*. 2000: Addison-Wesley.
23. Leveson, N.G., et al., *Requirements Specification for Process-Control Systems*. IEEE Transactions on Software Engineering, 1994. **20**(9).

Agent-oriented Requirements Modelling

Liang Xiao and Des Greer
Queen's University Belfast, Belfast, UK
email: { l.xiao, des.greer }@qub.ac.uk

Abstract

We describe how reconstructing requirements specification using agents and UML helps to build a hierarchy of requirements knowledge which can then be transformed into a running agent system. With agents as the division units for organising functional requirements structurally, the relationship between different pieces of the requirements document appears more obvious and, checks for completeness and consistency are easier to perform. The approach has four steps: i) Goal Mapping, where we relate functional requirements to system goals; ii) Structural Modelling, where agents are identified and rules for these agents are specified; iii) Behavioural Modelling, where the business processes are documented in UML and XML, aiming at realising the system goals by participating agents; iv) Implementation, where a running agent system is automatically generated from the transformed requirements document. We illustrate the approach and its tool support using a real life case study. Using the knowledge from this, we discuss and evaluate the approach.

1. Introduction

1.1. The demand for a new requirements/design modelling approach

UML Use Cases can help with requirements capture by providing a structured way to elicit and identify the actors and the interactions they have with the system. UML Class Diagrams describe the structure of a system and Sequence Diagrams describe a sequence of required actions, in terms of messages being passed. These requirements and design models provide a high level view of the system and help to reduce variability in the specification. However, the requirements models, being largely textual descriptions of system functions, are separated from the design models, which lack the capability to describe behavioural semantics exactly [1]. This is a major limitation of traditional software system development. The models rapidly lose their value as soon as coding starts. This loss in value is compounded by the fact that, in practice, changes are often done at the code level only. Thus, the connection between the models and the code fades away as the coding phase progresses [2]. This situation has been recognised in Extreme Programming (XP) [3] which focuses on coding and testing. XP proponents argue that the building of models is an overhead in the eyes of developers. Counter to this, others argue that without these high-level models, developers, especially new team members will get lost in code, being unable to understand what the software is doing and, consequently, maintenance becomes more difficult. Disregarding modelling contradicts with the expected intention that business is the master, rather than the servant, of technology and limits business agility and technology freedom by emphasising the *what* over the *how* [4]. A further argument in favour of models is found in the need to get the requirements as correct as possible and as early as possible, thus avoiding more expensive changes at the design and implementation stages.

In this paper we assume that both requirements and design models are useful and seek to increase their utility by integrating them. We propose that a good structure for the requirements specification is one that facilitates a clear and comprehensible division into modules, models the system diagrammatically, captures the knowledge on system functions in detail and links them to the elements in the design models. In addition, the integrated models should be easily adaptable (or adaptive) and changes easily checked for consistency and sensibleness without regard to a particular implementation environment. Most useful would be the ability to transform the models into actual software products. In this case, the modelling becomes the primary task in software development, and time spent on it will never be wasted.

1.2. MDA, a potential solution?

The value of system models is increased, if they can be converted to executable software. Model Driven Architecture (MDA) [2] [5] is an up and coming area that seeks this goal. In MDA, models are central rather than an overhead in the development process. MDA proposes a

Platform Independent Model (PIM), a highly abstracted model, independent of any implementation technology. This is translated to one or more Platform Specific Models (PSM). One difficulty in this process is that the process of PIM→PSM→code starts from the design products rather than requirements models. Consequently it requires highly creative work [2] to build a PIM from narrative requirements documents. This results in a high cost of requirements change due to the need of highly skilled professional engineers for the process.

Recognising that UML alone is not able to capture some semantics in its diagrams [6], a combination of UML and OCL [5] is used in MDA. However, OCL constraints are static and are external 'add-ons' to UML [7] and used in the design stages rather than the requirements stages.

1.3. Agent-oriented modelling

Adopting the objectives of MDA, we propose a modelling paradigm starting from constructing requirements models that are re-useable throughout the software development phases. Such models, in the first place, capture the overall system structure with individual functional requirements captured in the model elements. The requirements models are built from the transformation of the original requirements document in a systematic way, and they in turn can be transformed into the final software products. Supporting tools that can recognise this structured requirements and the semantics of its division units are provided to assist the transformation. The approach is agent-oriented, goal-guided, and rule-based. Systems are documented in UML and XML. We illustrate the approach using a real case study concerning the management of a rail track network.

2. Background

2.1. Business Rules – the missing piece in UML

According to the Object Management Group [5], business rules are "declarations of policy or conditions that must be satisfied". In [6], it is stated that they capture functional requirements, the decisions, guidelines and controls behind the functionality, and are the true essence of functional requirements. Also, as long as inconsistency and ambiguity are identified, either in rules or by rules, they are proved to be useful for capturing and resolving conflicts, both in the requirements level [8] and in the design level [9]. However, to date, rather than made explicit as part of the requirements document, their importance is ignored and often embedded directly into the final software product. This not only leads to the misinterpretation by developers with their own assumptions, but also provides no way to trace the rules in the code back to the requirements.

Although UML, accompanied with OCL [4] [5] [7], is able to specify constraint rules on its model elements, these rules are restricted to the design. To properly derive business rule requirements as business assets and let them remain valuable to stakeholders, they should be elicited and validated in collaboration with business customers during requirements analysis. For example, use cases are at the centre of the UML and can have business rules linked to them. Adding business rules to use cases provides better structured requirements than the traditional use case narratives. The use of business rules can compensate for the lack of capability of existing UML diagrams for capturing behavioural semantics.

It is suggested in [6] that business rules have the form of terms, facts, factor clauses and action clauses, expressed in natural language using a tailored taxonomy. According to this suggestion, if business rules are wrong, or subject to frequent change according to business needs, then there will be a lot of maintenance work. Thus, better still is the case where business rules are present in a structured natural language format and also executable by the system. Finding a way to make these rules executable would contribute significantly to the solution of transforming functional requirements to the final product.

2.2. Agents – execution engine of business rules

Business rules exist only to support business goals [6]. Therefore, the validity of a rule can be checked for mapping to a business goal. Since business rules are constraints about what must or must not be the case [4] and capture functional requirements, these rule statements not only constrain but also guide the realisation of business goals. On the other hand, groups of cooperative software agents interact with each other towards common goals [10]. Thus *agents* and *business rules* complement each other in the implementation: the former is the actor, the latter guides it and dictates the role that it plays. They work together to achieve business goals. On the Requirements Engineering (RE) level, an agent is the container and a business rule is the knowledge to fill the container. The mapping from requirements to implementation for both the agent and the business rule brings the traceability to the original requirements.

The RE level concept of agent has been introduced in [11] and the agent is advocated as a guiding concept, similar to how the object has been central in the OO approach. Some agent-oriented RE frameworks have been put forward. One such is Composite System Design [12]. This uses agents as composite components. Global goals of the system are decomposed until they can be assigned to individual agents. “Goals” are replaced by “responsibilities” and assigned to the agents during the design and agents are enabled with capabilities to behave. Similarly, KAOS [13], a goal-directed RE framework requires designers to refine goals until they are reduced to constraints that can be assigned to agents. The Albert II approach [14] expresses functional requirements using formal statements and groups them around agents to define the admissible behaviour of agents. A commonality of these frameworks is that agents are used as a way for requirements organisation. Requirements for the system feature are collected around agents as the basis to direct agents on how to behave.

2.3. Business Goals –putting agents and business rules in business processes

It is suggested that goal-based reasoning is central to requirements engineering and that goals are also abstractions that stakeholders are familiar with and are interested in [15]. The refinement of goals can help to build a comprehensible requirements structure and combined with agent assignment, alternative system solutions can be explored [16]. Many goal-oriented requirements engineering approaches are also agent-oriented, some of which are listed in section 2.2.

2.4. Business Processes – the organisation unit

The UML version 2 has thirteen official diagram types [1]. However, none of these provides entirely suitable constructs for expressing business goals.

Business process is a main/central construct for business modelling and is closely related with the requirements [4]. It follows that business processes are also closely associated with use cases, the main requirements construct in the UML. Nevertheless use cases are not good at expressing goals, while business processes describe how an organisation carries out a set of discrete but related activities to achieve business goals. Each activity can be accomplished by an agent playing a business role. Consequently, with the aid of the collaboration of all these agents, the goal of the business process is achieved. System requirements therefore can be represented by a set of business processes, each process being one that agents interact within to achieve a particular system goal.

3. Solution

3.1. Meta-model for requirements model

Figure 1 shows a meta-model of our Agent-oriented Requirements Modelling approach. In this approach, a *business process* has a *business goal* [4], *agents* collaborate towards *business goals* and *business rules* support business goals. *Agents*, therefore, have the responsibility of achieving the business goals. Using this terminology we can describe, not only what functional requirements should be together, but also how they function sequentially. Agents collaborate by playing *business roles*, which are, in turn, dictated by business rules, which represent the fundamental functional requirements. Agents are the medium that incorporate functional requirements into the business process models, in the form of business rules, connecting agents’ input and output ports. Business processes provide a framework that agents and rules can be put together to make the requirements inter-related and understandable from a high level. Thus, our models are agent-oriented, goal-guided and rule-based business process models.

A business rule in such models is developed as much more complex than a traditional simple constraint. It can have a nested structure, where other rules or functions are invoked by this rule to accomplish a specific task, when it is executed by an agent to fulfil a role. *Business functions* are distinguished from business rules. Business functions stay as internal to their master agents only, invoked by business rules, and invisible to external agents. They are owned by the lower level *business classes*, while conversely business rules stay in a higher level, owned directly by agents. Business classes are managed by the agents. This thereby constitutes a hierarchy. In the hierarchy, one piece of functional requirements becomes either a class method, or an agent rule. We will show our requirements modelling approach based on this meta-model in the following sections.

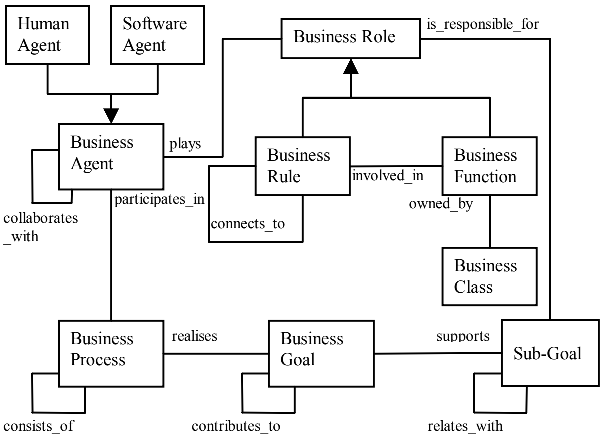


Figure 1. Meta-model of the requirements modelling approach

3.2. Case study

Underpinning the approach is a traditional functional requirements specification. These are usually documented textually, often in a form-based fashion using natural language. One actual requirements document, a national railway track system specification, has been investigated and the appropriateness of our agent-oriented requirements modelling approach assessed. The system is mainly responsible for the running of the railway on a daily basis, monitoring train running with regard to incidents and ensuring the safety of the train service by conveying issues to relevant parties for resolution. Being a very complex system, the document has more than 250 pages and contains a large number of function descriptions in a unified form as typified in Table 1. Relationship between function tables is not immediately obvious and this makes it difficult to maintain consistency.

Table 1. The original requirements documented in functional requirements tables

Accept Late Addition

Description	To handle a late request for a train journey.
Cause	Receipt of a request for a train journey directly from a **Train Operator** or from the driver entering the production function's area. ***The request is provided in the form of a combination of relevant train details, locations and desired timings***.
Assumption	The crew is competent for the route requested.
Information Used	Relevant locations.
Outputs	A new train journey, to **Train Operator** and others.
Required Effect	A new train journey is created from the request, and validated (**PF.TR.B-ValidateTrainPlan**). If the train journey is acceptable then it is distributed to all interested parties; otherwise the request is rejected or renegotiated. Having been accepted, the new train journey is known to the Production Function.
Impact of Unavailability	The unavailability of this function would render the Production Function unable to respond at short notice to customer requests for additional train paths.
Identifier	PF.TR.B-AcceptLateAddition

Validate Train Plan

Description	To validate a train journey.
Sub-Reqt of	**PF.TR.B-AcceptLateAddition**
Information Used	Train journey and relevant train details, sectional running times, locations and track restrictions.
Required Effect	The Production Function checks that the train journey: • complies with the Rules of the Route and, where relevant, the Rules of the Plan; • does not conflict with other train journeys and planned possessions; and • neither introduces unacceptable disruptions to this or other services nor increase the sensitivity of the Train Plan to disruption beyond "an acceptable level". The Production Function must also balance the needs of all its customers.
Identifier	PF.TR.B-ValidateTrainPlan

Two fundamental models are involved. Firstly, a *structural model*, with the UML Class Diagram as its counterpart in the OO world, is developed for structural relationship modelling. Secondly, a *behavioural model*, with the UML Sequence Diagram as its counterpart in the OO world, is proposed for behavioural interaction modelling. Agents are used to model the conceptual domain units, business rules are used to model the behaviours, and message passing is used to model the interactions.

3.3. Step one: goal mapping

Our approach assumes that the main goals of the system under development are explicitly stated in the user requirements document. We believe that organising functional requirements in terms of their goals can build a hierarchal requirements structure, where those at the bottom support those at the top. Therefore, we use a goal-decomposition technique, similar to [17], to expose all relevant functional requirements tables. Top level requirements are those most valued by the business people and reflect the final business goals. Subordinate goals can be derived using our goal decomposition technique down to the lowest level goals, which map to individual functional requirements. This is necessary because we need to model the business processes in terms of all participating functions in order to later implement those processes. One important business goal documented in the user requirements for the rail track system is to "deliver train journeys safely". Whereas this goal may have been used as the basis to construct multiple functional requirements tables, the link from individual functional requirements may not have been maintained in the requirements document. Figure 2 demonstrates the decomposition process for the goal "deliver train journeys safely".

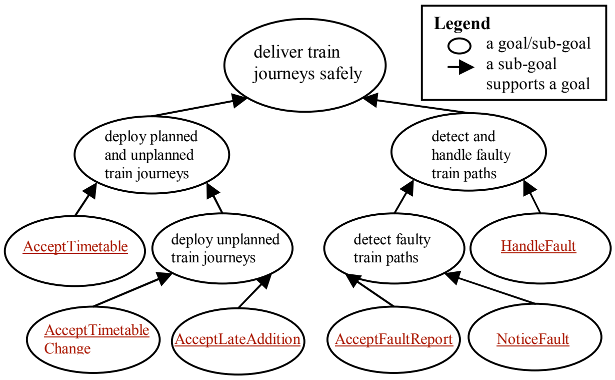

Legend
a goal/sub-goal
a sub-goal supports a goal
deliver train journeys safely
deploy planned and unplanned train journeys
detect and handle faulty train paths
AcceptTimetable
deploy unplanned train journeys
detect faulty train paths
HandleFault
AcceptTimetable Change
AcceptLateAddition
AcceptFaultReport
NoticeFault

Figure 2. Goal decomposition graph for the case study

This business goal, in the first place, can be decomposed into "deploy planned and unplanned train journeys" and "detect and handle faulty train paths". The former sub-goal can then be decomposed according to the nature of the train journeys and the latter into the detection and handling of faults. These sub-goals can be further decomposed into smaller sub-goals and considered just like ordinary goals in the subsequent decomposition process. Finally, when the business goal is decomposed into the smallest granularity, the process terminates and we find that all the leaf nodes, underlined and highlighted in the figure, are presented as functional requirements tables. Among these is "AcceptLateAddition" as documented in Table 1. We map one table for the functional requirements to a sub-goal in a leaf node or, alternatively, a combination of such tables to a goal/sub-goal in an intermediate node. Therefore such graphs can help the deduction of intermediate goals, provide a way to organise functional requirements, and check the completeness and validity of the original user requirements. For example, only with all the leaf nodes existing as requirements tables in the document and with the top nodes being fully supported by the bottom nodes can the business goals guaranteed to be represented in the requirements.

Note that a business process can be delegated to a sub-goal at the graph end, such as "AcceptLateAddition", and also to an intermediate goal, as a relatively complex case, such as "detect and handle faulty train paths", if its sub-goals are inter-related. We will only discuss the first case because of limited space.

3.4. Step two: structural modelling, including agent identification and rule transformation

Agents can be actors or information systems, identified in the knowledge domain, aiming at accomplishing the business goals by participating in business processes. To realise the goal "AcceptLateAddition", an independent goal identified in Step One, a business process will be delegated, the detail of which will be presented in the next section. Before that the involved agents should be identified.

In our case study, "AcceptLateAddition" (see Table 1), "AcceptTimetable" and "AcceptTimetableChange" belong to a single "Train Running" business domain. (This is indicated by the shared prefix "TR.B" field in their specification table identifier.) Consequently, they belong to the same agent, which we name "TrainRunning". Note that it is a coincidence that the common goal of, or the one supported by the three, is the same - "deploy planned and unplanned train journeys". In many cases one goal is realised by multiple agents, as a result of cross functional division interactions [17]. Notice that the concept of agent should be distinguished between the requirements modelling and the implementation. Agents in this phase are used to organise the requirements and when implemented as software units they are responsible to meet their corresponding requirements.

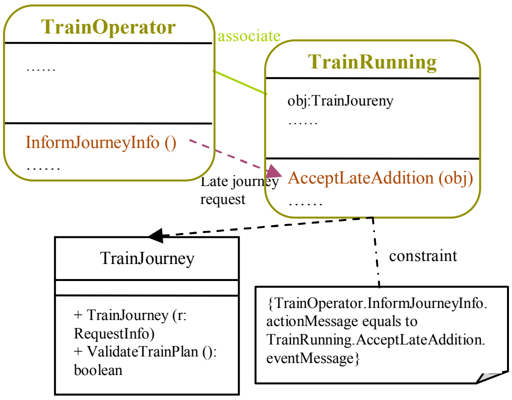

TrainOperator
……
InformJourneyInfo ()
……
associate
TrainRunning
obj:TrainJoureny
……
AcceptLateAddition (obj)
……
Late journey request
TrainJourney
+ TrainJourney (r: RequestInfo)
+ ValidateTrainPlan (): boolean
constraint
{TrainOperator.InformJourneyInfo. actionMessage equals to TrainRunning.AcceptLateAddition. eventMessage}

Figure 3. Agent Diagram for case study

The static structure of the system is modelled using the Agent Diagrams, an example of which is shown in Figure 3. The identified agents represent conceptual domains and are used to organise functional requirements tables. This diagram has the UML Class Diagram as its counterpart in OO models, but in our approach, the agent is regarded as the first class citizen rather than the class. In an Agent Diagram, each box that represents an agent is divided into three compartments, just like a class box in a UML Class Diagram. Despite this similarity, they have different content, the compartments in the Agent Diagram being: the name of the agent, the classes managed by the agent along with their instantiation, and rules that govern the functions of the agent.

As the meta-model in Figure 1 shows, to accomplish a sub-goal, an agent in fact plays a role, and that role is responsible for the sub-goal. A role can be either a rule or a function. In most cases, it would be a rule, which may connect to other rules and have functions involved. For the case study requirement “AcceptLateAddition”, a single functional requirements table represents a sub-goal and its owner agent “TrainRunning” is responsible to realise it by playing a role. We can call this role “AcceptLateAddition”, the same name as the goal. Also in the case where the role becomes a rule the same name given to it. In contrast would be the case that multiple roles are played to contribute to a higher level goal. The simple goal in our case is to make “late addition” accepted, while the single role/rule involved is defined in the functional requirements table telling the agent how to do it. “ValidateTrainPlan” is involved to assist the rule “AcceptLateAddition” to realise the goal, by receiving some information from the rule, and returning some results back to it. In the case, train journey information is received and a boolean value is returned. Although both are presented as functional requirements table in Table 1, they are distinguished, because “ValidateTrainPlan” has “AcceptLateAddition” in its “Sub-Reqt of ” section.

The requirements from the case study captured by the Agent Diagram in Figure 3 are: the agent “TrainRunning” and “TrainOperator” are the only agents involved for the business goal. The two recognised agents represent the “train running planning” domain and the “train operating company” domain respectively. “TrainOperator” has a rule of “InformJourneyInfo” that will construct a “TrainJourney” object, package it into a “Late journey request” message and send the message to the “TrainRunning”. This may reflect increasing demand of train journeys during special dates and events in real life. To respond to such requests, “TrainRunning” will plan additional journeys using the rule of “AcceptLateAddition”. The required goal of deploying additional train journey is documented in the functional requirements table, which also describes how the goal is achieved by using such a procedure.

During processing of the rule “AcceptLateAddition”, a “TrainJourney” object will be constructed according to the journey request information encoded in the message. The constructed object should pass a validation check before being put into use. Therefore it is a necessity that a business class “TrainJourney” is managed by the “TrainRunning” agent and the class has a constructor method and a validation method to be invoked by the agent. “ValidateTrainPlan”, presented in Table 1 as one of the functional requirements tables, represents that validation method of the business class. It is involved in the rule “AcceptLateAddition” to assist its function rather than work as an independent rule.

As we can see, such an Agent Diagram can be used to structure the requirements specification with agents, rules, classes and messages and provide details of their relationship. Agents are higher level entities having rules that govern their behaviours. Business classes with functions are in a lower level, used by the agents.

A rule essentially captures a functional requirement in that it lays out action that should be taken, on receipt of an event (normally modelled as a message), if certain conditions are satisfied. This is analogous to the traditional components of function input, function use context, function process and function output. From this basic structure which is shared by all rules, we split each rule into several related compositional parts and each captures one aspect of the requirements for a function.

Figure 4 illustrates the rule specification template, formulated according to functional requirements tables documented in Table 1. A “Cause” section is used to make the rule “event”; its sections of “Information Used” and “Required Effect” are used to make the rule “processing”; its “Required Effect” and “Outputs” are used to make the rule “condition” and “action”. Thus events cause agents to execute rules, if certain conditions are satisfied, some actions are triggered which in turn include generated events for other agents. We also conceive “belief” as an integral part of the rule structure. “Belief” is a collection of knowledge that an agent can learn from the messages received from other agents.

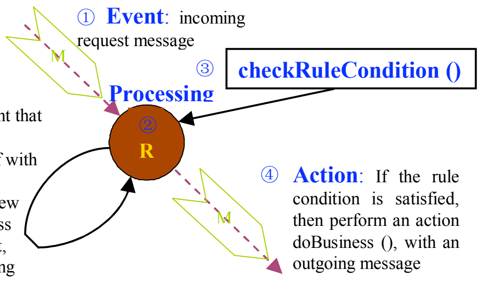

① Event: incoming request message
M
③ checkRuleCondition ()
Processing
⑤ Belief: the agent that executes the rule updates its belief with the information received in the new message (business company interest, customer shopping habits, etc.)
②
R
M
④ Action: If the rule condition is satisfied, then perform an action doBusiness (), with an outgoing message

Figure 4. Rule template (with only one pair of {condition, action} considered)

Thus the rule “AcceptLateAddition”, transformed from the functional requirements table in Table 1, would be specified as:

1. Receive a late journey request message from TrainOperator.
2. Construct a TrainJourney object using the information contained in the message.
3. If the created object can pass the ValidateTrainPlan () evaluation method.
4. Then send a message with the created TrainJourney to TrainOperator and other relevant agents (the alternative condition is omitted here as a simplification), and
5. Add the belief that the TrainOperator has made such a request at this moment.

3.5. Step three: behavioural modelling, including business process specification

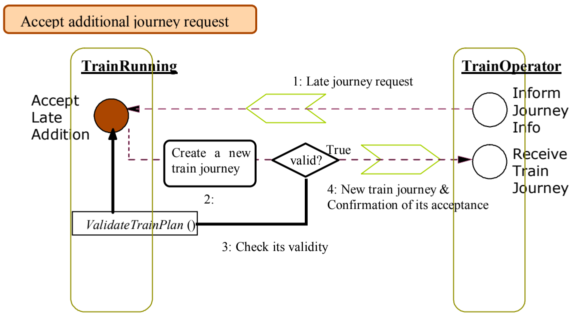
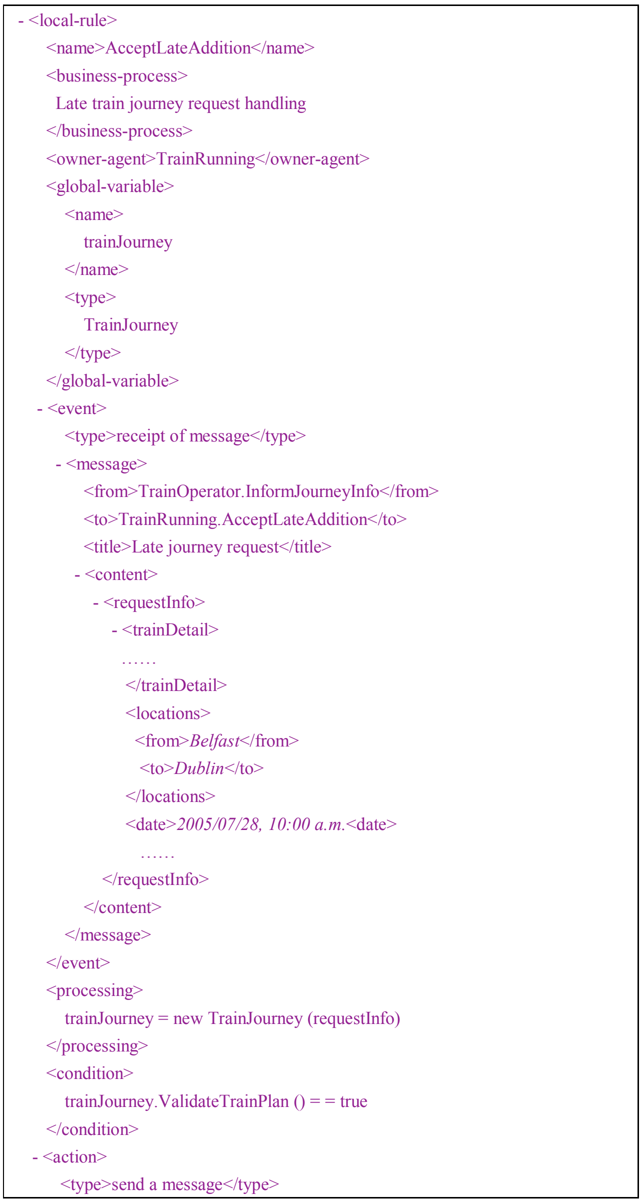


Figure 5. Agent Communication Diagram describing a Business Process Rule (with only the default condition considered)

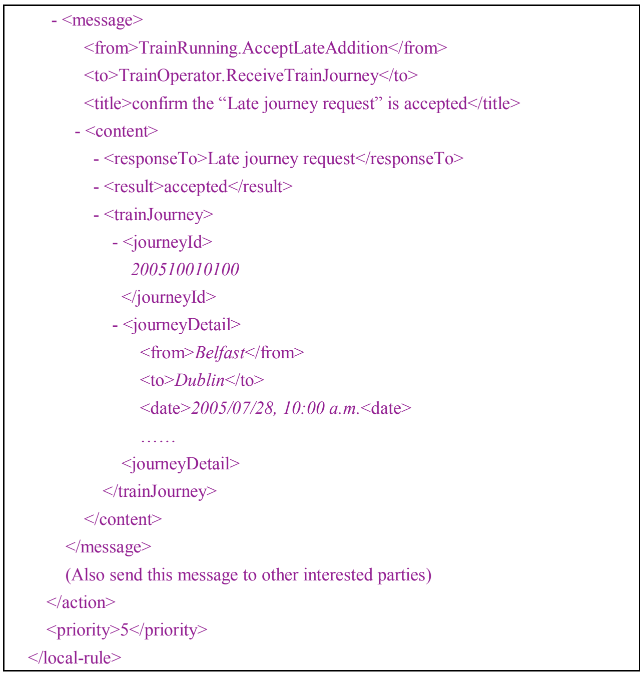
The agent/rule and class/method identification is completed when the captured requirements are documented in the Agent Diagrams. Agent Diagrams are capable of capturing the structure of related agents and rules, as reflected in the requirements. Like UML Class Diagrams, the Agent Diagrams focus exclusively on structure and ignore behavioural details. Therefore, additional modelling methods are required to represent behaviour [1]. For OO systems, the Sequence Diagram is a commonly used means to capture the behaviour of a scenario, showing the participant objects and the messages that are passed between these objects [1]. Similarly, we need to organise associated agents and their rules to show how their behaviours can finally achieve the goals belonging to the business processes. Thus, we group agents by business goals/processes and each agent can appear in multiple groups using different rules for different purposes. Agents and their rules are organised according to the business processes they participate in. Requirements specification therefore can be divided according to business processes. For each business process, rules that dictate agent behaviours can be documented.

In order to ensure that behavioural models of the requirements are understandable for computers and can facilitate future automation, we introduce the Agent Communication Diagrams as a means to visualise agent behaviour alongside individual rule definitions. Higher level Business Process Rules (BPR) can be formed to specify business processes. They are the division units for organising the specification, well-defined in UML. One such BPR is composed of a collection of lower level ordinary rules, each defined in XML. Tools for the transformation of these models into implementation are developed and described in section 3.6.

A sample Agent Communication Diagram is given in Figure 5. Each BPR defined in the diagram specifies the reaction of all involved agents for their common goal. Related reaction rules are grouped in BPR. Transformed from the traditional requirements specification, each composite rule is composed of four essential parts as discussed earlier. These components are described using four XML tags: <event>, <processing>, <condition>, and <action>. They reflect respectively four steps an agent take to execute a rule in the Agent Communication Diagram, numbered from 1 to 4 (a "belief" component would be added in the future for additional agent intelligence). The XML definition of the rule for this case is shown in Figure 6.



```
- <local-rule>
    <name>AcceptLateAddition</name>
    <business-process>
      Late train journey request handling
    </business-process>
    <owner-agent>TrainRunning</owner-agent>
    <global-variable>
      <name>
        trainJourney
      </name>
      <type>
        TrainJourney
      </type>
    </global-variable>
  - <event>
      <type>receipt of message</type>
    - <message>
        <from>TrainOperator.InformJourneyInfo</from>
        <to>TrainRunning.AcceptLateAddition</to>
        <title>Late journey request</title>
      - <content>
        - <requestInfo>
          - <trainDetail>
              ……
            </trainDetail>
            <locations>
              <from>Belfast</from>
               <to>Dublin</to>
            </locations>
            <date>2005/07/28, 10:00 a.m.<date>
              ……
          </requestInfo>
        </content>
      </message>
    </event>
    <processing>
      trainJourney = new TrainJourney (requestInfo)
    </processing>
    <condition>
      trainJourney.ValidateTrainPlan () = = true
    </condition>
  - <action>
      <type>send a message</type>
```



```
- <message>
    <from>TrainRunning.AcceptLateAddition</from>
    <to>TrainOperator.ReceiveTrainJourney</to>
    <title>confirm the “Late journey request” is accepted</title>
  - <content>
    - <responseTo>Late journey request</responseTo>
    - <result>accepted</result>
    - <trainJourney>
      - <journeyId>
          200510010100
        </journeyId>
      - <journeyDetail>
          <from>Belfast</from>
          <to>Dublin</to>
          <date>2005/07/28, 10:00 a.m.<date>
          ……
        <journeyDetail>
      </trainJourney>
    </content>
  </message>
  (Also send this message to other interested parties)
</action>
<priority>5</priority>
</local-rule>
```

Figure 6. XML definition for the rule “AcceptLateAddition”

Agent Communication Diagrams are good at showing collaboration among agents, while rules are good at precise definition of agent behaviours, which is not possible in UML Sequence or other Diagrams [1].

The definition of rule “AcceptLateAddition” in XML (Figure 6) is based on the rule specification given in section 3.4 and is executable by computers. In the case of rule “Accept Late Addition”, suppose all previous rules managed by the agent “TrainRunning” are not applicable and it’s time to check the applicability of this rule. The agent parses the message just received and finds out that the message has come from an agent with the name “TrainOperator”. The agent knows its rule “Accept Late Addition” is defined to deal with the message received from that agent, because according to the rule definition, the content of XML element <event>/<message>/<from> matches with that agent in the name. Also the message content has the same structure as specified in the rule (in rule definition shown in Figure 6, the content between tags in italic is possible message content, where the rule specifies it will only accept events with this kind of structure). Then, by invoking business classes, managed by the agent, a business object of “trainJourney” can be constructed (the name of which is declared and shared as a global variable), and its validity can be evaluated. The “trainJourney” object can be built from the content of the <event>/<message>/<content> structure of the received message. Its validity can be checked using the “ValidateTrainPlan ()” method of the business class. If the “trainJourney” is valid then the condition for executing the rule is satisfied. A corresponding message will then be structured using the created “trainJourney” object (the name of which refers to the global variable) and sent to all interested agents, including “TrainOperator”, again as it is specified in <action>/<message>/<to>. Finally the “TrainRunning” agent adds the knowledge that the “TrainOperator” agent has sent a late train journey request at this time to its own beliefs. When enough such information is collected, business reports can be built for analysis for the later use. In the whole rule execution process, if the event does not match or the condition is not satisfied, the next candidate rule with the highest priority will be tested for applicability and executed in the same way.

After the behavioural models are built, the requirements are documented in a set of Business Process Rules, each delegated for a high level business goal. The details of business rules that constitute the business processes are further defined in XML, representing system functionalities that realise the goals. With the Agent Communication Diagrams and XML-based rules documented, any later change to the traditional specification will be mapped to XML element content, according to the mapping from the four main sections of functional requirements to the four main XML elements, and the corresponding UML model structures. The advantage of this is that the maintenance of the requirements on UML diagrams and XML definitions is much easier than the text-based descriptions and the UML/XML-based format is less likely to bring ambiguousness, incompleteness and inconsistency. These changes will be further transformed to the implementation so that maintenance of code is avoided. The details of this are in the next section.

To illustrate one of the benefits the approach brings, suppose there is a change that requires the function production being available to a different entity, as described in the <Required Effect> section of the traditional specification. This means that an agent needs to send its actionMessage to a different agent, hence the matching XML tag <action>/<message>/<to> of the function’s corresponding rule will change. This also requires us to check the validity of the association between this agent and its previously associated agent. This check compares the XML tag <event>/<message>/<from> of its associated rule and the <Cause> section of that rule which matches the original functional requirements. Moreover, the change can be visualised in our UML diagrams, the direction of a message switching from the changed rule to a different agent. This example demonstrates how the consistency of the requirements can be checked at any time.

3.6. Step four: implementation

```
thisAgent. addBehaviour (Rule thisRule) {
      thisBehaviour.setPriority (thisRule.getPriority ());
      TrainJourney trainJourney;
      Message m = thisAgent.receiveMessage ();
      while (m != null){
         Agent fromAgent = m.getSenderAgent ();
         if (fromAgent.equals (thisRule.getEvent ().getMessage ().getFromAgent ()))
         {
            /* the rule is applicable to the received message */
            RequestInfo requestInfo = (RequestInfo) m. getContentObject ();
            trainJourney = new TrainJourney (requestInfo);
            if (trainJourney.ValidateTrainPlan ()) {
               /* the condition of the rule is satisfied */
               Message m2 = new Message ();
               m2.setContentObject (trainJourney);
               Agent toAgent = thisRule.getAction ().getMessage ().getToAgent ();
               m2.addReceiverAgent (toAgent);
               thisAgent.send (m2);
               /* update this agent's beliefs */
               thisAgent.addBelief (System.getCurrentTime (), fromAgent, m);
            }
         }
       m = thisAgent.receiveMessage ();
      }
}
```

Figure 7. Pseudo code for the case study, transformed from the requirements models

Agent-oriented systems can be generated from the models obtained in step three. It should be noted that, although agent-oriented implementation is a straightforward and also the recommended one, it is not restricted to that, due to the fact that the requirements models built are neutral requirements documentation. In fact they have no assumptions on the technology used to implement them.

Nevertheless, implementing each conceptual agent in our models as a software agent is natural. One running unit should be used to represent the domain and several such units collaborate in an information passing fashion to achieve a common goal, with each contributing its own knowledge or knowledge processing capability. A tool has been developed which generates agent systems running on the JADE platform [18]. Each generated agent represents a corresponding agent in the Agent Diagram.

According to the steps that agents follow for the rule execution processes, there will be an agent behaviour generated for each rule of the agent. Thus a functional requirement maps to an agent behaviour in this approach. These behaviours can function simultaneously so that agents are multi-threaded. A shared module "Rule" is used by all behaviours with the ability to access the XML definition of rules and assemble corresponding objects. Methods of getPriority(), getEvent(), and getAction() are provided in the "Rule". On receiving an incoming event, an agent reacts by performing one of the defined behaviours. The one with the highest priority is retrieved to check its applicability. It will be executed if it passes the check; otherwise the same procedure will be carried out to check the next one, and so on, until one of the available behaviours is performed. For each single behaviour, first of all, the tool generates a "setPriority()" statement, then retrieves all global variables used in the rule and declares these.

After that the tool generates an "if" statement to evaluate if the received message matches with the expected incoming message, as specified in the rule. If this is the case, the received message is processed and some business objects are assembled at this stage (they are usually the declared global variables). A check is then made to test whether the rule condition is satisfied. Finally the tool generates an instantiation statement for a new message, a set content statement to encode some business objects declared as the global variables to the message, a send message statement to send it, and an add belief statement to update the agent's beliefs .

Figure 7 shows the pseudo code for an agent behaviour generated by the tool (Figure 8). Only minor human effort has been put to adjust the code. This sample is for the rule "Accept Late Addition", agent "TrainRunning".

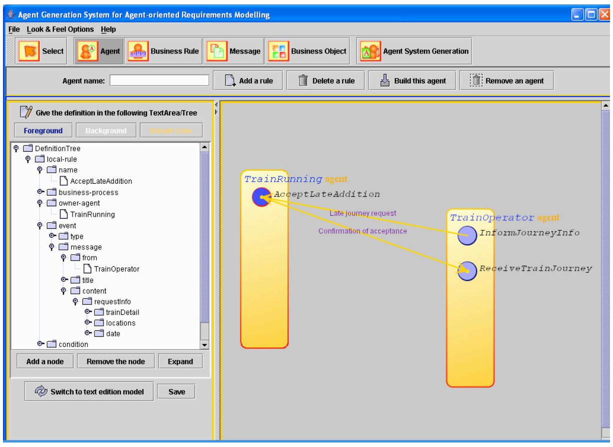


Figure 8. Screenshot from supporting tool

The overall result is, one piece of the functional requirements in Table 1 is modelled as an event-based rule "AcceptLateAddition", and then an agent behaviour as shown in Figure 7. Another piece is modelled as an internal business method "ValidateTrainPlan ()", for the "TrainJourney" business class. The later implementation element is invoked to assist the former implementation element in the above agent behaviour code, reflected in the original requirements as the later is a "Sub-Reqt of" the former. Thus functional requirements have been modelled properly in two different levels: rules and methods, reflecting their relationship in the original requirements.

4. Evaluation and Conclusion

Extra notations are defined in our modelling approach to complement UML in capturing structural and behavioural semantics as well as internal and external collaboration. As opposed to traditional requirements models, the agent-oriented modelling approaches as proposed have an advantage over the plain UML models in that they are not a development overhead. Executable systems are actually automatically generated from our models. This is due to the fact that accompanying XML rule definitions are used to explain function details for each UML element and to describe their behaviours. These requirements can not be well documented in graphical UML diagrams. With this approach, there will be no excuse for skimping on requirements/design and accurate translation from the diagrams to the running systems is possible. Maintenance of the models is actually maintenance of the running systems.

As the requirements model itself is also intended as a design model, the requirements and design phases are in effect merged. Because of this, our approach should reduce the cost of maintenance. When requirements change from the baseline system specification, the only manual changes needed are in the UML diagrams and the XML. These are much less painful to modify than the code. They can be updated using the same steps given for the requirements model transformation wherever there are changes. The implementation would be automatically re-generated by tools. Hence, we do not need to look for places in the code to reflect every change in the requirements. This helps us to avoid the risk of touching code frequently during the maintenance and guides us to an effective way to change the system. Business classes used by rules, however, may need to be maintained manually. Since they are independent simple components, each representing one aspect of the business domain, they are relatively stable. The interaction and logical combination of these is at the agent level and can be re-configured frequently and easily, supported by methods and tools.

Our investigations using real requirements specifications show promise in providing a method for constructing requirements specifications using agents and ultimately generating running agent systems which precisely conform to their requirements models. At the moment, they have limited autonomy and intelligence after the transformation, truthfully reflecting the requirements demand. However, the integral design of agent beliefs leaves the possibility of adding advanced autonomous features to agents. For example, agents may resolve requirements conflicts or may reject some requirements from a judgement based on their experience. This add-on will be developed as a future enhancement to strengthen the approach.

5. References

[1] Fowler, M., "UML Distilled: A Brief Guide to the Standard Object Modeling Language, 3rd Edition", Addison-Wesley, 2004.

[2] Kleppe, A., Warmer, J. & Bast, W., "MDA Explained: The Model Driven Architecture: Practice and Promise", Addison-Wesley, 2003.

[3] Beck, K., "Extreme Programming Explained: Embrace Change", Addison-Wesley, Boston, 2000.

[4] Morgan, T., "Business Rules and Information Systems", Addison-Wesley, 2002.

[5] Object Management Group, Inc., 250 First Ave. Suite 100, Needham, MA 02494, USA.

[6] Ambler, W. S. & Constantine, L. L., "The Unified Process Inception Phase: Best Practices in Implementing the UP", R&D, 2000.

[7] Warmer, J. & Kleppe, A., "The Object Constraint Language: Getting Your Models Ready for MDA, Second Edition", Addison-Wesley, 2003.

[8] Rosca, D. & Wild, C., Towards a flexible deployment of business rules, Expert Systems with Applications, Vol 23, Issue 4, 385-394, 2002.

[9] Liu, W., Easterbrook, M. S. & Mylopoulos, J., Rule-based detection of inconsistency in UML models, Proc. Workshop on Consistency Problems in UML-Based Software Development, 5th Int. Conference on the Unified Modeling Language, Dresden, Germany, 2002.

[10] Jennings, N. R., Sycara, K. & Wooldridge, M., A Roadmap of Agent Research and Development, Autonomous Agents and Multi-Agent Systems, 1 (1998) 7-38.

[11] Yu, E., Why Agent-Oriented Requirements Engineering, Proc. 3rd Int. Workshop on Requirements Engineering: Foundations for Software Quality, Barcelona, 1997.

[12] Feather, M. S., Composite System Design, ICSE-16 Workshop on Research Issues in the Intersection Between Software Engineering and Artificial Intelligence, Int. Conference on Software Engineering, Sorrento, Italy, 1994.

[13] Dardenne, A., Lamsweerde, A. & Fickas, S., Goal-directed Requirements Acquisition, Science of Computer Programming, Vol 20, Issue 1-2, 3-50, 1993.

[14] Heymans, P. & Dubois, E., Scenario-Based Techniques for Supporting the Elaboration and the Validation of Formal Requirements, Requirements Engineering, Springer, 3, 202-218, 1998.

[15] Lamsweerde, A., Requirements engineering in the year 00: A research perspective, Proc. 22nd Int. Conference on Software Engineering, 5-19, ACM Press, 2000.

[16] Lamsweerde, A., Goal-Oriented Requirements Engineering: A Guided Tour, Proc. 5th IEEE Int. Symposium on Requirements Engineering, Toronto, IEEE, 249-263, 2001.

[17] Kavakli, V. & Loucopoulos, P., Goal-Driven Business Process Analysis - Application in Electricity Deregulation, Information Systems, Vol 24, No 3, 187-207, 1999.

[18] Jade platform, http://sharon.cselt.it/projects/jade/

Value-Based Business-IT Alignment in Networked Constellations of Enterprises

Roel Wieringa
Department of Computer Science
University of Twente
The Netherlands
roelw@cs.utwente.nl

Jaap Gordijn
Department of Computer Science
Free University, Amsterdam
The Netherlands
gordijn@cs.vu.nl

Pascal van Eck
Department of Computer Science
University of Twente
The Netherlands
vaneck@cs.utwente.nl

Abstract

Business-ICT alignment is the problem of matching ICT-services with the requirements of the business. In businesses of any significant size, business-ICT alignment is a hard problem, which is currently not solved completely. With the advent of networked constellations *of enterprises, the problem gets a new dimension, because in such a network, there is not a single point of authority for making decisions about ICT support to solve conflicts in requirements these various enterprises may have. Network constellations exist when different businesses decide to cooperate by means of ICT networks, but they also exist in large corporations, which often consist of nearly independent business units, and thus have no single point of authority anymore. In this position paper we discuss the need for several solution techniques to address the problem of business-ICT alignment in networked constellations. Such techniques include:*

- *RE techniques to describe networked value constellations requesting and offering ICT services as* economic value. *These techniques should allow reasoning about the matching of business needs with available ICT services in the constellation.*
- *RE techniques to design a networked ICT architecture that supports ICT services required by the business, taking the value offered by those services, and the costs incurred by the architecture, into account.*
- *Models of decision processes about ICT services and their architecture, and maturity models of those processes.*

The techniques and methods will be developed and validated using case studies and action research.

Paper type: Research position paper

1. Introduction

Business-ICT alignment is the problem of matching ICT services with the requirements of the business. In businesses of any significant size, business-ICT alignment is a hard problem, which is currently not solved completely.

Additionally, most businesses can not be viewed anymore as a single enterprises with precisely one point of authority of decision taking on ICT support for business need satisfaction. Rather, businesses form *networked value constellations* [17] to satisfy complex customer-needs. Well-known examples are Cisco Systems and Dell, but many other constellations exist in practice. By a networked value constellation we mean a network of profit-and-loss-responsible business units, or of independent companies. Networks exist when different businesses decide to cooperate by means of ICT networks, but they also exist in large corporations, that often consist of nearly independent business units. For example, large companies may acquire other companies that must remain profitable; or they may restructure themselves into a number of cooperating business units that are all profit-and-loss responsible. Businesses may outsource some or even most of their activities. In yet other scenarios, companies may join a value chain or start a cooperation with a number of other companies to implement an e-commerce idea.

Networked value constellations place strict requirements on ICT support, because it is ICT that enables and allows the creation of such a constellation in the first place. Without properly functioning ICT, there can be no networked value constellation.

Networked business-ICT alignment has the characteristic feature that there is no single point of decision taking regarding ICT. In practice, many enterprises are involved, with different and, in many cases, conflicting interests. Because economic value —monetary value— is a well known means to make trade-offs between enterprises with conflicting interests, we propose to deal with the alignment problem of networked constellations using a value engineering viewpoint. Value-oriented techniques need to be investigated by which one can design networks of services and implement these in a network of business processes and systems. In our approach we view a networked value constellation as a set of enterprises exchanging object of value with each other. Here the objects are ICT services that satisfy a business need. In order to facilitate automated reasoning on alignment, we need to conceptualize and formalize such constellations from a customer (business) perspective as well as from a ICT-supplier perspective.

Secondly, the design of ICT architectures for networked value constellations must be done in such a way that expenses related to the architecture become apparent, and can be used in the value engineering viewpoint for assessment of economic sustainability of the chosen architecture.

Finally, to reach a certain level of alignment in a networked value constellation, it is required that processes to do so are in place at the participating enterprises. Obviously, such processes are often not executed by enterprises yet. To arrive at enterprises that have the capabilities to align their business needs properly with offered ICT services in a network, a Capability Maturity Model (CMM) is needed, identifying the minimum set of core capabilities to reach a certain alignment level.

In this research position paper we analyze these research problems and sketch the solution approach that we have embarked upon. We sketch the research framework in section 2 and research questions in section 3. In section 4 we compare our approach with other approaches, and with the current needs of industry. Section 5 concludes the paper with a discussion of the current state of the research.

2. Research Framework

To structure the problem and explain the research questions, we use the research framework shown in figure 1. First we structure a business constellation into a number of service provision layers. From the bottom up, these layers are as follows:

- The *physical infrastructure,* consisting of buildings, computers, cables, wireless access points, radio waves, printers, etc.
- The *software infrastructure,* consisting of operating systems, middleware, network software, database management systems, office software, etc. We define infrastructure (physical and software) as a utility service, required to be present and functioning for all users when and where they need it. Software infrastructure is rapidly growing in functionality; for example, the telephone system is nowadays integrated with the software infrastructure.
- *Business systems,* consisting of software applications and information systems acquired and used for the service of particular business processes and particular users. In contrast to infrastructure, business system design is driven by the needs of particular users, particular business processes, and particular business domains, not by the needs of all possible users, all possible processes and all business domains.
- The *business constellation,* consisting of processes, organizational roles and units that perform value adding activities and exchange physical objects and services of economic value.
- The *business constellation environment,* consisting of other business actors, customers, suppliers and other stakeholders.

We have motivated the suitability of these layers for architecture research elsewhere [23]. Cross-cutting these layers are several important aspects, including the following.

- *Services.* These are useful activities performed by entities at the various layers.
- *Value.* Services are useful, by definition, when they produce economic value for some actors.
- *Semantics.* The services we are interested in are ICT services, and these consist of storing and manipulating data, that have a semantics.
- *Communication channels.* ICT services are delivered by transmitting data across channels connecting actors.
- *Process.* At all levels in the hierarchy, services are delivered by sequences of interactions ordered in time, called processes.
- *Quality.* Service delivery has a certain quality, such as usability, efficiency, etc.

We have shown the relevance of these aspects, except the value aspect, in earlier research in software and systems design frameworks [20, 21, 22]. We added the value aspect

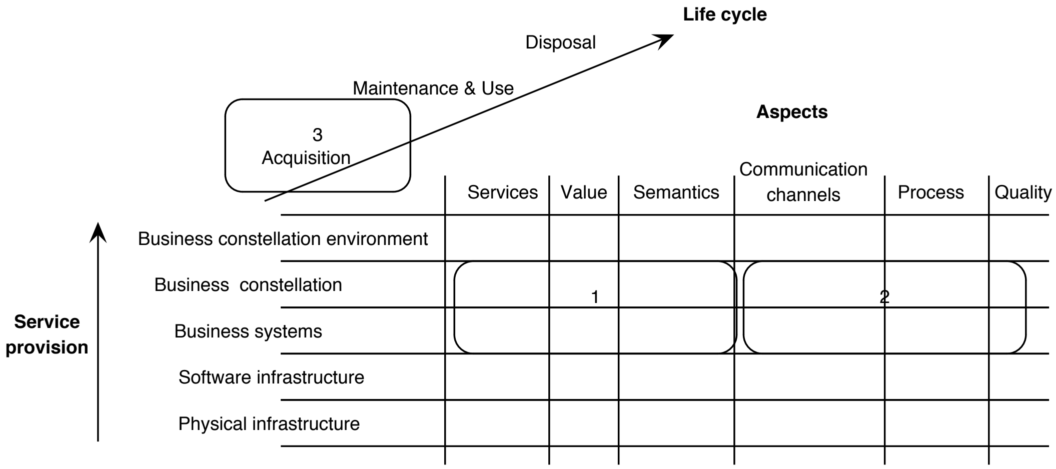


Figure 1. Research framework.

to address business-ICT alignment from an economic value point of view.

Orthogonally to these two dimensions, there is a *life cycle dimension,* which indicates that entities at each of these layers have a life cycle starting with acquisition and ending with disposal. During their life, entities have properties as shown in our framework: They provide services that should be of value and that should have semantics, etc.

3 Research questions

We can explain our three major research questions in terms of our framework. The first question concerns value-based ICT service specification (area 1 in figure 1), the second concerns the realization of these services by networked business processes and business systems (area 2), and the third concerns the architecture processes by which these specification and design activities can be realized (area 3).

Figure 2 explains the relationship between the three areas in terms of a business network (service consumer and service provider) and life cycle phase. Arrow A represents decisions made by the provider and consumer about what services will be offered by whom. The key working hypothesis is that we regard arrow A as *commercial service provisioning*, both in the case of cooperating independent companies, but also within one company. Arrow A corresponds to area 1. The vertical arrows B through E represent the realization of services in business processes and systems, and their influence on the service model. This is area 2. And where areas 1 and 2 study design techniques, area 3 studies the design processes involved in this life cycle phase. Arrow F in the figure represents IT service management, and is out of the scope for this research.

More in detail, the three areas contain the following research questions.

1. **Value-oriented requirements engineering (RE).** Here our research goal is to specify ICT services from a business value perspective. We will do this by building upon previous research by Gordijn and Akkermans [6], in which the e^3*-value* method for designing a network of value activities and value exchanges was developed. We also developed a supplier-oriented service provisioning ontology, which has been used, as an extension to e^3*-value*, by the electricity and entertainment industries to define bundles of services to be offered by cooperating electricity companies to consumers [2]. What still needs to be done is to design a service ontology from a *customer* (i.e. business) point of view, and to specialize the supplier and consumer-oriented ontologies to the ICT service provisioning domain. Additionally, we need to develop techniques for matching ICT-requirements, expressed cf. the earlier mentioned customer-side service ontology, with ICT services to be offered by suppliers. We plan to address these issues by the following research questions:

   (a) Which ontologically founded concepts are needed to conceptualize ICT services, both from a consumer and a supplier perspective, such that preferably automated matching of consumer's ICT needs and supplier's ICT services is feasible? Additionally, the ICT services ontology should properly relate to the e^3*-value* ontology.

   (b) How can we match supplier-oriented and consumer-oriented ICT service specifications? We need to consider ways to compose supplier

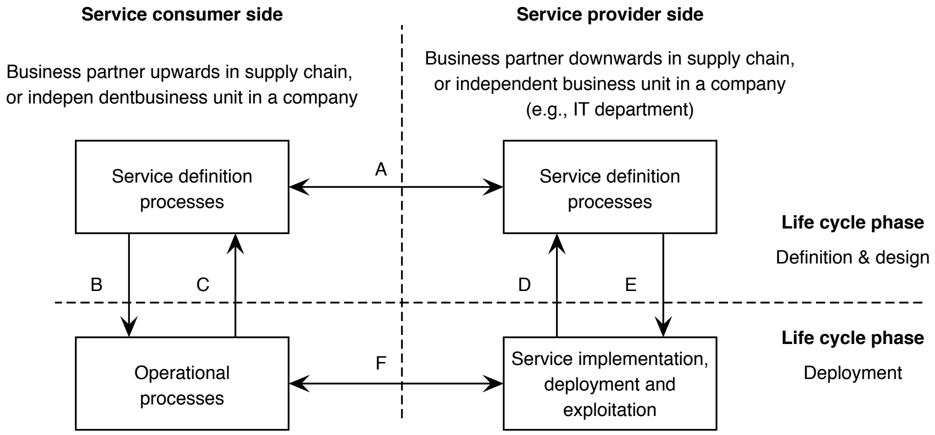

Service consumer side
Service provider side
Business partner upwards in supply chain, or indepen dentbusiness unit in a company
Business partner downwards in supply chain, or independent business unit in a company (e.g., IT department)
Service definition processes
A
Service definition processes
Life cycle phase
Definition & design
B
C
D
E
Life cycle phase
Deployment
Operational processes
F
Service implementation, deployment and exploitation

Figure 2. A networked business.

services into bundles that are valuable from a consumer perspective and profitable for all concerned. We intend to deliver software support for solving the matching and composition problem.

(c) How can we estimate the economic value delivered by a service? The e^3-*value* approach and supporting software tool already have facilities for economic value analysis of services. We want to extend and specialize them for the ICT services domain.

We will investigate these questions by using our previous work on service specification and value engineering [2, 3, 6, 7], and by using theories from investment analysis [9] and software engineering economics [4, 19]. We will validate our results jointly with our business partners by means of action research.

2. **Business-ICT architecture design.** In this area we need to investigate how to implement services in a networked business. In terms of our framework, this requires a definition of the business systems (applications and information systems), their external behavior, communication and quality attributes so that they support the desired business services, using as many existing systems as possible. This leads to the following questions.

(a) How can existing systems be configured so that the desired services are delivered at the required quality of service? We need to link configuration decisions to desired services. Furthermore, we will investigate how to rank the relevant architectures on their support of different required services, and how to make value-based decisions among them. We will validate these techniques in simulated case studies and action research.

(b) How can we design a network of business systems to provide the services as identified in area 1? Classical methods such as Information Engineering [10, 15] design modular systems by means of CRUD analysis but in a networked context this is not sufficient, as ownership is not taken into account. Modular networks involve decisions about different kinds of ownership (of data, of processes, of systems) each with different cost and revenue structures, communication requirements, and access restrictions. We will investigate the use of value-based techniques to make these decisions in practice by means of case studies, design new techniques and validate them in simulated case studies and action research.

(c) How does ICT-architecture influence the value network? We showed earlier that this influence exists [24]. For example, a decision to outsource ICT services requires enterprises are to be added to the value network; and this may in turn require adding an additional enterprise that assists in outsourcing, introducing additional value exchanges.

(d) All previous three research questions touch in one way or another on the question when a model of business systems and business processes (the rightmost columns of figure 1) is “correct” with respect to a model of value network (the leftmost three columns of figure 1). The value network expresses business requirements to be satisfied by an architecture of systems and processes on the right. The research question is what the appropriate correctness notion is, and how we can provide support for proving a correctness relation between the value model and architecture model.

We will investigate questions (b), (c) and (d) by means of action research and simulated case studies, i.e. we will propose techniques, and then validate them in simulations and in consultancy projects.

3. **Architecture maturity model.** Business-ICT alignment can be reached and done at various levels of maturity. There have been some proposals for architecture alignment maturity models [18], but these are oriented to single businesses and do not incorporate the value viewpoint. In this area, we study architecture processes in networked businesses and develop a maturity model for this that incorporates the value viewpoint.

   (a) Which decision processes take place in networked businesses when allocating services to a distributed ICT architecture? How can we use value-based specification and allocation techniques in these processes?

   (b) What is the relationship between these processes and known maturity models such as CMMI, the IT Service CMM and the REAIMS maturity model [11, 12, 16]?

   (c) How can maturity levels for architecture management be defined? What process areas are needed at each level?

   Except for the question how to use value-based decision techniques, these questions are empirical, not normative, and we will investigate them by means of case study research. The normative question how to use value-based specification and allocation techniques in these processes will be studied by simulated case studies, i.e. by showing how these techniques could have been used in the cases that we study. With our business partners we will identify user organizations where we can study the structure of architecture design processes.

Note that the research methods mentioned above are empirical: Very briefly, case study research is the analysis of projects performed by others [25], and action research is the analysis of projects in which the researcher participated [14]. We will also use *simulated* case studies, in which we will explore what would have happened if our techniques would have been used in a case studied by us.

4 Comparison with related work

The combination of value engineering with service-oriented requirements engineering and architecture design is, to our knowledge, new and currently not investigated elsewhere. This approach leads to interesting new insights in requirements engineering that we need to explore further, for example concerning the use of problem frames at the business level [24].

As observed before, our research effort is about business-ICT alignment for networked businesses, and is not limited to alignment in a single enterprise. Classical methods like Information Engineering [10, 15] analyze functions, processes and semantics domains in one business to then design information systems using modularity arguments (i.e. CRUD analysis). In this research effort, we take a network point of view and extend these techniques with value-based techniques to design and implement value networks.

Value-based software engineering extends software project management with techniques that relate decisions to their impact of budgets and business objectives [5, 8]. We do not study project management (although we will look at the architecture process) and we will focus on ICT service provision for networked business.

Asundi used techniques from investment theory in decisions about the mix of architecture styles to be used to support a given set of quality attributes [1], but this does not relate architecture to service requirements in a networked business, as we do.

The RAISA project (`http://www.ifi.uib.no/projects/raisa/`) investigates architecture alignment in a model-driven framework [13]. Although RAISA does allow inclusion of the network view, the focus on networked business integration and the commercial value of architecture decisions, that is at the heart of VITAL, seems to be absent from RAISA.

5 Discussion and conclusions

Current businesses face an architecture integration problem caused by the presence of legacy systems, vestiges of island automatization, acquisitions and mergers of other companies, and the increasing importance of value chain automization and of business networks. These developments facilitate outsourcing of non-core business activities and, increasingly, of ICT development activities. In some cases outsourcing takes the form of offshoring to low-wage countries. This trend is currently very clearly observable. All these developments require a well-integrated and business-aligned ICT architecture. Our research aims to deliver techniques to align business perspectives of various enterprises with ICT-architecture integration and outsourcing decisions, operationalize this by means of validated techniques for integrated business process and information system architecture design, and facilitate implementation of these techniques by means of an architecture process maturity model.

The research described in this position paper will be done in the coming four years in cooperation with about 10 consultancy firms and ICT service providers, who will act as a sounding board and as a source of industrial case studies. More information can be found at http://www.vitalproject.org/. We are actively seeking cooperation with other researchers in this area.

References

[1] J. Asundi, R. Kazman, and M. Klein. Using economic considerations to choose among architecture design alternatives. Technical Report CMU/SEI-2001-TR-035, Software Engineering Institute, 2001.

[2] Z. Baida, J. Gordijn, A. Z. Morch, H. Sæle, and H. Akkermans. Ontology based analysis of e-service bundles for networked enterprises. In *Proceedings of the 17th Bled International e-Commerce Conference*, page to appear, 2004.

[3] Z. Baida, J. Gordijn, H. Sæle, A. Z. Morch, and H. Akkermans. Energy services: A case study in real-world service configuration. In A. Persson and J. Stirna, editors, *Advanced Information Systems Engineering. 16th International Conference, CAiSE 2004*, volume 3084 of *Lecture Notes in Computer Science*, pages 36–50. Springer-Verlag, 2004.

[4] B. Boehm. *Software Engineering Economics*. Prentice Hall, 1981.

[5] B. Boehm and L. Huang. Value-based software engineering: A case study. *Computer*, pages 33–41, March 2003.

[6] J. Gordijn and J. Akkermans. Value-based requirements engineering: Exploring innovative e-commerce ideas. *Requirements Engineering journal*, 8(2):114–134, 2003.

[7] J. Gordijn, J. Schildwacht, V. Kartseva, R. Wieringa, and H. Akkermans. A domain-specific cross-organizational requirements engineering method. In *12th IEEE International Requirements Engineering Conference*, sept 6–10, Kyoto 2004.

[8] W. Harrison. Economic-driven software engineering research. `http://www.grabpage.org/~edser/`.

[9] C. T. Horngren and G. Foster. *Cost Accounting: A Managerial Emphasis, sixth edition*. Prentice-Hall, Englewood Cliffs, NJ, 1987.

[10] J. Martin. *Information Engineering*. Prentice-Hall, 1989. Three volumes.

[11] F. Niessink. *Perspectives on Improving Software Maintenance*. PhD thesis, Vrije Universiteit Amsterdam, Division of Mathematics and Computer Science, 2000. `http://www.serc.nl/people/niessink/publications/PhD.Niessink.pdf`.

[12] F. Niessink. Perspectives on improving software maintenance. In *Proceedings of the International Conference on Software Maintenance*, pages 553–556, 2001. `http://www.serc.nl/people/niessink/publications/ICSM01.Niessink.pdf`.

[13] A. Opdahl. Model-supported alignment of information systems architecture. In K. Kangas, editor, *Business Strategies for Information Technology Management*. Idea Group Publishing, 2003.

[14] C. Robson. *Real World Research*. Blackwell, 2002. Second edition.

[15] W. v. d. Sanden and B. Sturm. *Informatiearchitectuur: De infrastructurele benadering*. Panfox Holding, 1997.

[16] I. Sommervile and P. Sawyer. *Requirements Engineering: A Good Practice Guide*. Wiley, 1997.

[17] D. Tapscott, D. Ticoll, and A. Lowy. *Digital Capital - Harnessing the Power of Business Webs*. Nicholas Brealy Publishing, London, UK, 2000.

[18] H. van der Zee, P. Laagland, and B. Hafkenscheid. *Architectuur als managementinstrument: Multi Client Study*. Ten Hagen Stam, 2000. In Dutch.

[19] C. Verhoef. Quantitative IT Portfolio Management. *Science of Computer Programming*, 45(1):1–96, 2002. Available via: `www.cs.vu.nl/~x/ipm/ipm.pdf`.

[20] R. Wieringa. *Requirements Engineering: Frameworks for Understanding*. Wiley, 1996.

[21] R. Wieringa. A survey of structured and object-oriented software specification methods and techniques. *ACM Computing Surveys*, 30(4):459–527, December 1998.

[22] R. Wieringa. *Design Methods for Reactive Systems: Yourdon, Statemate and the UML*. Morgan Kaufmann, 2003.

[23] R. Wieringa, H. Blanken, M. Fokkinga, and P. Grefen. Aligning application architecture to the business context. In *Conference on Advanced Information System Engineering (CAiSE 03)*, pages 209–225. Springer, 2003. LNCS 2681.

[24] R. Wieringa, J. Gordijn, and P. v. Eck. Value framing: A prelude to software problem framing. In *1st International Workshop on Advances and Applications of Problem Frames (IWAAPF)*, 2004.

[25] R. Yin. *Case Study research: Design and Methods*. Sage Publications, 2003. Third Edition.

Designing Data Warehouses: From Business Requirement Analysis to Multidimensional Modeling

Jose-Norberto Mazón[1], Juan Trujillo[1], Manuel Serrano[2], and Mario Piattini[2]
[1]Dept. of Software and Computing Systems
University of Alicante
Apto. Correos 99. E-03080
{jnmazon,jtrujillo}@dlsi.ua.es
[2]Alarcos Research Group
University of Castilla – La Mancha
Paseo de la Universidad, 4; 13071 Ciudad Real
{Manuel.Serrano,Mario.Piattini}@uclm.es

Abstract

Most of the data warehouse projects still fail because the final data warehouse does not properly meet business goals. Designers start a data warehouse project with the conceptual or logical modeling of the multidimensional schema, and unfortunately, not much attention has been paid on the requirement analysis phase. However, this phase is very important, because it can include the understanding of the business context in which the data warehouse is pretended to work. This is a crucial issue, since the aim of data warehouses is to provide enough information in a suitable way to improve decision making and accomplish with business goals. In this paper, we propose an approach to take into account business context and their business goals in data warehouse requirement analysis phase. First of all, we adapt i notation to model business environment and goals for data warehouses requirement analysis. Then, from i* models, a multidimensional model which satisfies business goals is obtained. To avoid an arbitrary use of our approach, we provide a set of guidelines to correctly specify i* diagrams and transform them into a multidimensional model. Finally, we apply our approach to a case study to show its benefit.*

1. Introduction

Even though a decade later, data warehouses still pay a central role in current decision support systems, since they are oriented to provide adequate information to improve the decision making process [4]. Nowadays, it is widely accepted that the basis for designing the data warehouse repository is the multidimensional modeling [4,5,6,8]. Conceptual models have been provided to be able to represent main properties of the multidimensional modeling that satisfy final user requirements [5,8]. Nevertheless, even though we use conceptual models, many data warehouse projects still fail because traditionally not much attention has been paid on the requirement analysis phase. Therefore, the final data warehouse may not reflect organization needs and may not deliver the expected support of the decision making process [3,12]. This process is crucial in organizations, since making better decisions allows them to improve business processes by achieving business goals. Moreover, several studies have shown that more than 80% of data warehouse projects fail to meet business goals [12]. Often, business goals are ignored as a result of poor communication between IT and business professionals during requirement analysis. Therefore, it is obvious that an effort is needed to develop data warehouses within a business context by incorporating explicit understanding of the business into data warehouse requirement analysis using some organizational modeling technique [19]. Then designers will be able to develop data warehouses that provide organizations with the necessary information to fulfill their business goals. A summary of main benefits that organizations pretend to achieve with the use of a data warehouse can be viewed in figure 1. These benefits can be achieved if the data warehouse is understood within its business environment.

Thus, we present an approach for including business issues (i.e. business goals) in data warehouse requirement analysis and then, transform requirements in a multidimensional model that helps to fulfill business goals. Since the i* technique provides understanding of the organizational environment and goals in requirement analysis phase [19], we adapt i* diagrams to requirement analysis in data warehouses. We also structure business goals that data warehouse helps to achieve into strategic, decision and information goals. This allows developers to have a better understanding of business and, then users can communicate better their ideas. Finally, from these i* diagrams, we obtain a multidimensional conceptual model which provides organization with the adequate information to fulfill business goals. This model is designed using our UML (Unified Modeling Language) profile [7,8,14]. Furthermore, we provide, based on our experience in designing real world data warehouses, a set of design guidelines to correctly specify these i* diagrams and transform them into the corresponding multidimensional model.

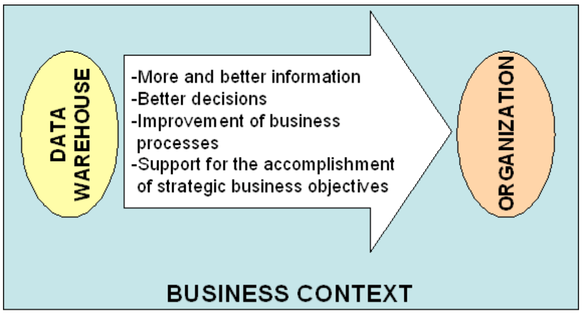

DATA WAREHOUSE
-More and better information
-Better decisions
-Improvement of business processes
-Support for the accomplishment of strategic business objectives
ORGANIZATION
BUSINESS CONTEXT

Figure 1. Some benefits derived from developing the data warehouse within a business context (adapted from [1]).

The rest of this paper is structured as follows. Section 2 presents the most relevant related work for requirement analysis in data warehouses. Our approach for requirement analysis and its guidelines for properly specify the i* and multidimensional models are presented in section 3. We show the benefit of our approach in section 4 by means of a little case study. Finally, in Section 5 we present our conclusions and sketch some future works.

2. Related Work

In this section, we will make a brief description of the most relevant approaches for requirement analysis in data warehouses. We want to point out that, most of approaches have a main drawback: they are not part of a global methodology in which we can directly obtain, from the requirements, the corresponding conceptual multidimensional schema that provides adequate information to fulfill business goals.

Böhnlein et al. [2] derive a data warehouse from business process models. They point out that it is relevant to focus on goals and strategies of the company for an efficient decision making, since this information cannot be only extracted by analyzing operational data sources. Therefore, it is crucial to situate the data warehouse within a business context and analyze this context. However, as Winter and Strauch [17,18] point out, only a detailed business process analysis is not feasible because decision processes consist of tasks which are often unique and unstructured, and decision makers often refuse to disclose their process in detail. Thus, they present a methodology based on determining information requirements which data warehouse users need in decision processes, and matching information requirements with actual information supply (operational sources), because these seem to be more stable, more concrete, and better accessible. Nevertheless, they do not use any notation to represent business goals and understand the business context in which the data warehouse works.

Schiefer et al. [12] present a method, easyREMOTEDWH (easy Requirements Modeling Technique for Data Warehouses), which considers data warehouse requirements from different stakeholders perspectives, according to several levels of abstraction. They include an interesting business point of view to represent business objectives and needs. Unfortunately, they do not present a notation or guidelines to properly specify requirements.

Prakash et al. [11] also propose a requirement elicitation process for data warehouses grouping requirements in several levels of abstraction. Their process consists of identifying information that support decision making via information scenarios. The process starts with the determination of the goals of an organization. Secondly, the decision making needs are specified, and finally, the information needed to cover these decisions is identified. In this process, they use a Goal-Decision-Information (GDI) diagram. Although they show how to obtain the GDI diagram and the information scenarios, the relationships between information scenarios and requirements are not properly specified. Moreover, they only represent interaction between decision makers and the data warehouse, and how to obtain organization needs are not considered. Finally, they lack in using guidelines to specify data warehouse requirements.

Paim et al. [10] present the DWARF (Data Warehouse Requirements deFinition) technique. They adapt traditional requirements engineering process (even capturing non-functional requirements, such as performance or accessibility, using the NFR framework) to propose a methodological approach for requirements definition and management of data warehouses. However, they focus in technical issues (e.g. how to access data), but they do not explicitly obtain business goals.

Therefore, from the above-presented approaches, we can summarize the following: (i) they pay little attention in providing mechanism to understand the business context in which the data warehouse will be deployed, or (ii) they do not provide a clear set of guidelines for data warehouse requirement analysis, or (iii) they are not part of a global methodology in which we can directly obtain the corresponding conceptual multidimensional schema that allows us to fulfill business goals.

3. From Business Requirements to Multidimensional Conceptual Schema

Since a data warehouse provides organization with information to support the decision making process in order to achieve business goals, a data warehouse requirement analysis approach should not only deal with technical details (e.g. data warehouse architecture or access to data), but also business issues should be taken into account in early phases of data warehouse development. This allows designers to understand the business environment and develop a data warehouse which meets the real needs of organizations. Therefore, better decisions will be taken and business goals will be achieved.

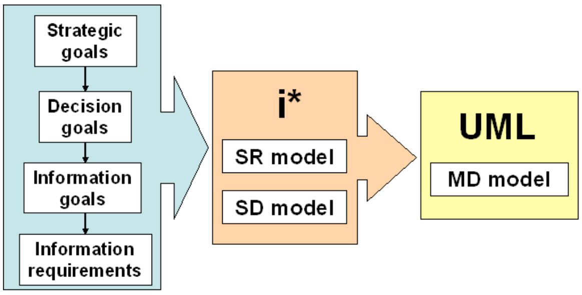


Figure 2. Overview of our approach for requirement analysis in data warehouses.

In our approach (see figure 2), we focus on defining goals that organization must achieve and relationships among stakeholders needed to fulfill them. These goals are the main objectives the organization wants to achieve by implementing a data warehouse and can be classified in strategic, decision, and information goals. From these business goals, we derive information requirements as information provided by the data warehouse to achieve business goals. Our point of view about business goals and information requirements is explained in subsections 3.1 and 3.2. These business goals and information requirements must be represented in an organizational model together with users of the data warehouse and the necessary relationships between the data warehouse and its users needed to achieve business goals. In subsection 3.3, we explain how to use i* technique [19,20] to model these business issues. In subsection 3.4, we explain how to transform information requirements into a multidimensional model [15] which provides the needed information (facts, dimensions, levels of aggregation...) to achieve business goals.

3.1. Business Goals for Data Warehouses

Business goals describe the objectives that organization pretends to achieve through the cooperation of actors in the environment (i.e. executives, managers, data warehouse, etc.). In fact, the foundation of the data warehouse requirements is describing goals of stakeholders, since they often express their information needs in general expectations of the data warehouse to improve their business [12]. Business goals that a data warehouse helps to achieve are considered to form a hierarchy of nested goals depending on the level of abstraction (it is represented in figure 2):

- Strategic goals represent the highest level of abstraction. They are main objectives of the business process. They are thought as changes from a current situation into a better one. For example: “increase sales”, “increase customers”, “decrease cost”, etc. Their fulfillment causes an immediate benefit for the organization.
- Decision goals represent the medium level of abstraction. They try to answer the question: “how can a strategic goal be achieved?”. They are objectives which need the ability of forming judgement about something and taking a determination in order to be achieved. For example: “determining some kind of promotion” or “open new stores”. Their fulfillment only causes a benefit for the organization if it helps to reach strategic goals, since decision goals only take place within the context of a strategic goal.

- Information goals represent the lowest level of abstraction. They try to answer the question: "how can decision goals be achieved in terms of information required?". They are objectives based on determining what kind of information must be required to help to fulfill a decision goal. For example: "analyzing customer purchases" or "examine stocks". Their fulfillment helps to achieve decision goals and they only happen within the context of a decision goal.

In order to define a goal hierarchy [16] two questions must be asked to data warehouse users, "how" and "why". The former, discovering goals by refinement (top-down strategy), it is used for refining goals into subgoals. It consists on asking "how" questions about goals already identified: "how can this goal be satisfied?". However, the latter, discovering goals by abstraction (bottom-up strategy), it is used for introducing more abstract goals. It consists on asking "why" questions about other goals: "why is this goal useful?". In this paper, we focus on a top-down strategy.

3.2. Information Requirements for Data Warehouses

Data warehouse requirements must be considered in terms of information provided by the data warehouse to support the decision making in order to achieve business goals. These information requirements can be obtained from the information goals above-described. They are related to interesting measures of business processes (contained in facts) and the context for analyzing these measures (dimensions and their hierarchies).

3.3. Requirement Analysis for Data Warehouses

A requirement analysis phase for an information system must deal with analyzing, understanding, and modeling business context in which it works [20]. Regarding data warehouses, the aim of this phase is to represent users of the data warehouse, business goals of the organization in which data warehouse is integrated, and relationships between the data warehouse and its users in order to achieve business goals. Thus, business is taken into account in early stages of the development of the data warehouse by means of representing how data warehouse helps to achieve business goals. This phase is crucial in the data warehouse development, since usually stakeholders do not know how to describe information requirements and the final data warehouse may not reflect business needs. Therefore, this requirement analysis phase allows developers to situate the data warehouse within its business context and relate it to business goals. From these goals, designers can more easily obtain what the data warehouse users need to do with the data warehouse system to achieve business goals (i.e. information requirements). This approach is much more powerful than asking users what they want the system to do [3].

In our approach, we adapt i* technique [19] to model data warehouses within their organizational environments, since this technique allows representing actors, their dependencies, and structuring those business goals that organization pretends to achieve.
This technique consists of two models: strategic dependency (SD) model to describe the dependency relationships among various actors in an organizational context, and the strategic rationale (SR) model, used to describe actor interests and concerns and how they might be addressed. The central concept in i* models is the intentional actors, since organizational actors (i.e. data warehouse users) have intentional properties as they depend each other for goals to be achieved, tasks to be performed and resources to be furnished [20]. Due to the lack of space we refer reader to [19] for a further explanation of i*.

3.3.1. Using i* for Data Warehouses

The foundation of the SD model in data warehouses is that the organization depends on the data warehouse to obtain proper information to achieve its goals. So, business goals and information that data warehouse must provide to achieve business goals must be addressed in the SD model. On the other hand, data warehouses depend on information sources (internal and external) to populate facts and dimensions. So, information sources and their dependencies with the data warehouse must be also included in the SD model, then information supply and information requirement will be explicitly linked. Therefore, users (i.e. decision makers), the data warehouse under construction, and information sources are the main actors. We want to point out that, in this paper, we focus on modeling dependencies between users and the data warehouse and a future research will be done to achieve relationships between the actual data warehouse and the operational sources in order to complete data warehouse schema with data from operational sources.

In summary, in a SD model for a data warehouse, we can distinguish two kinds of dependencies: goal dependencies (users depend on the data warehouse to achieve their goals) and resource dependencies (information needed by user is provided by the data warehouse).

Although SD models describe business environments and dependencies between the users and the data warehouse, only external relationships among actors are showed. However, intentional constructs within each actor stay hidden [20]. On the other hand, the SR model provides a more detailed level of modeling internal intentional relationships of each actor. Intentional elements (goals, tasks, resources and softgoals) and their relationships (means-end and task-decomposition) are represented. Regarding data warehouses, we are interested in represent goals, tasks and resources as intentional elements. Following, we explain how to build the SD and the SR models for data warehouses.

3.3.2. Building the Strategic Dependency Model

Several guidelines to build the SD model for data warehouses are given. These guidelines are based on representing actors and dependencies between them.

Guideline 1. Discover business actors. These actors are decision makers (e.g. managers, top executives...). The data warehouse under construction is also considered as an actor. We have to represent these actors in a SD model.

Guideline 2. Determine strategic goals of organization from decision makers. These goals must be represented by means of goal dependencies between every actor and the data warehouse.

Guideline 3. Information required by decision makers is represented as a resource dependency between each actor and the data warehouse, since this information is provided by the data warehouse.

3.3.3. Building the Strategic Rationale Model

Guidelines to build SR model for data warehouses are given. These guidelines are based on representing internal intentional elements and relationships. Here, we also specify dependencies between actors with a more level of detail.

Guideline 4. For each actor who is a decision maker, intentional elements are obtained (in this case, goals and tasks). Several guidelines are given for obtaining and representing them.

Guideline 4.1. Refine main strategic goals (obtained in guideline 2), following a top-down strategy in order to obtain possible strategic subgoals. We have to keep on refining until obtaining decision goals. Strategic and decision goals are represented as goals. Relationships between them are represented as means-end links, since these links are used to describe how goals are achieved.

Guideline 4.2. Refine decision goals (obtaining subgoals) until obtaining information goals (top-down strategy). Each of these goals is represented as a goal. Relationships between them are represented as means-end links.

Guideline 4.3. Each information goal previously obtained is related to the analysis of some measure used to achieve that goal. This analysis describes an information requirement and it is represented as a task. Decision makers carry out this task in order to obtain information from the data warehouse to achieve required information goals.

Guideline 5. For the data warehouse actor, every task and resource (and their relationships) needed in order to provide adequate information (according to the previous guideline) is represented.

Guideline 5.1. For each resource dependency according to guideline 3, providing the adequate information is the objective for data warehouse actor. Then a goal is required to provide such information.

Guideline 5.2. Measures according to guideline 4.3, must be represented. These measures are represented as resources. However, if they are derived measures, then they are presented as tasks in order to calculate them. Both, resources and tasks, are linked to main goal with a means-end relationship. Measures needed to calculate derived attributes are represented as resources (linked by means of a decomposition link to its corresponding task).

Guideline 5.3. Analysis of each measure must be provided within a context. This is represented as tasks. Every task is related to main goal with a means-ends relationship. Within context of analysis, there are several levels of aggregation to analyze measures. These levels of aggregation are represented as resources. These resources are linked to every task that represents the context of analysis by means of a decomposition link.

3.4. From i* Model to Multidimensional Model

We must be sure that each of the tasks and resources reflected in the SD model for data warehouse actor must be addressed by a multidimensional model. This model must be useful to fulfill business goals.

In this paper we follow our UML profile for the conceptual design of data warehouses following the multidimensional paradigm [7,8,15]. The most important feature of this paradigm is dividing data into facts (composed of measures) and dimensions; to provide data on a suitable level of granularity, hierarchies are defined on the dimensions. This profile is defined by a set of stereotypes and tagged values to elegantly represent these main multidimensional properties at the conceptual level using a UML class diagram (see table 1). Due to lack of space we refer reader to [7,8,15] for a further explanation.

Table 1. Main stereotypes of the UML profile.

Stereotype	Description	Icon
Fact class	Represent facts consisting of measures	▦
Dimension class	Represent dimensions consisting of dimension attributes and hierarchy levels	z y x
Base class	Represent dimension hierarchy levels	B

Following, we describe several guidelines to specify a multidimensional class diagram by using our UML profile [7,8,15] for the multidimensional modeling at the conceptual level. This conceptual multidimensional schema is defined from i* models by identifying fact and dimension classes with their corresponding base classes (i.e. classification hierarchies) from the SD model. Identifying attributes within fact and dimension classes should be completed from operational sources.

The following guidelines are used to define a multidimensional class diagram from a SD model:

Guideline 6. Create a fact class for each main goal in the data warehouse actor. For each resource representing a measure we create an attribute. For each task representing a derived measure we create a derived attribute.

Guideline 7. Resources that represent the context of analysis become dimension classes.

Guideline 8. Levels of aggregation (i.e. base classes) are also specified from resources which represent the context of analysis. We want to point out that these base classes have not any attribute, due to the fact that these attributes stay in the operational sources. In this paper, modeling operational sources is not still considered, so enriching levels of hierarchies with attributes from operational sources will be considered in a future research. However, this model can be used as a prototype in order to know if business goals can be achieved by information requirements.

4. Case Study

The aim of this section is to exemplify the usage of our requirement analysis approach. We have selected a case study presented in Chapters 2 and 3 of Kimball's book [6] to show how we can obtain requirements within a business context.

Kimball's retail case study presents a brief description of the retail business which embraces both retail sales and inventory. This retail business is composed of several grocery stores spread over several regions. In each store several products are sold. At the grocery store, management is concerned with the logistics of ordering, stocking, and selling products while maximizing profit. The profit ultimately comes, among other things, attracting as many customers as possible in a highly competitive pricing environment. Thus, some of the most significant management decisions have to do with pricing and promotions used to increase the number of customer, since they include temporary price reductions in a grocery store. One of the most important tasks of managers is to determine whether a promotion is effective or not. Therefore, retail sales business process deals with analyzing what quantity of products are selling in which stores on what days under what promotional conditions.

In this case study, Kimball deals with several kinds of inventory models of a store. We are interested in the inventory snapshot example, where the inventory levels are measured every day and are placed in separate records in the database. Main management objective is making decisions to optimize inventory levels in order to decrease inventory costs. These decisions are related to make sure the right product is in the right store at the right time to minimize out-of-stocks (where the products is not available on the shelf to be sold) and reduce overall inventory carrying costs. So, the inventory management needs the ability to analyze daily quantity-on-hand inventory levels by product and store. Inventory manager is also concerned with measure the velocity of inventory movement (how the product is moving through the store) to know the benefits of sales. So, manager needs GMROI (Gross Margin Return Of Inventory). This is a derived measure which is calculated using the next formula:

$$GMROI = \frac{total\_quantity\_sold \times (value\_at\_latest\_price - value\_at\_cost)}{daily\_average\_quantity\_on\_hand \times value\_at\_latest\_selling\_price}$$

4.1. Strategic Dependency Model

In this subsection we apply our guidelines to obtain SR model represented in figure 3. We want to point out that operational sources are considered an actor due to we plan to extend our approach in a next future to consider them, but they are not still supported. However operational sources are very important in data warehouse development [17,18], so at least, we want to show them.

Guideline 1. Possible actors are: "marketing manager", and "inventory manager". "Data warehouse" must be also aggregated as an actor.

Guideline 2. "Increase number of customers" is a strategic goal for "marketing manager" and "decrease inventory costs" is a strategic goal for "inventory manager". Each of these strategic goals is represented as a goal dependency from actor to data warehouse as we can see in figure 3.

Guideline 3. Information needed for each actor to accomplish strategic goals is represented as a resource dependency from actor to data warehouse. In this case study "information about sales" and "information about inventory" are drawn in the SD model (see figure 3).

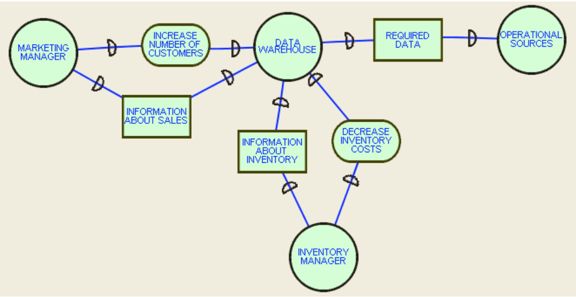


Figure 3. Strategic Dependency Model.

4.2. Strategic Rationale Model

Following we apply our guidelines to obtain SD model for each actor represented in the SD model. The SR models can be viewed in figures 4, 5, and 6.

Guideline 4. Obtain different kinds of goals and tasks of each actor who is a decision maker. We have to build one SR model for each actor ("marketing manager", "inventory manager").

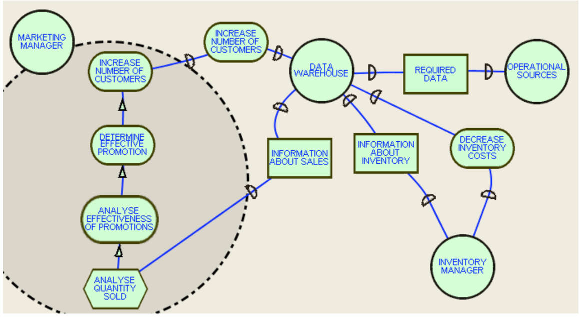


Figure 4. Strategic Rationale Model for Marketing Manager.

Guideline 4.1. Decision goals are obtained by following a top-down approximation for refining strategic goals obtained in guideline 2:

- How can we increase customers? Determining an effective promotion which allow customers to know new products.
- How can we decrease inventory costs? Taking measures to optimize inventory management.

Thus, for strategic goal "increase number of customers", a decision goal is discovered: "determine effective promotion" (see figure 4). Regarding strategic goal "decrease inventory costs", one decision goal is elicited: "optimize inventory management" (see figure 5).

Each decision goal is drawn as a goal and a means-end link is drawn between a decision and a strategic goal. So, "increase number of customers" goal is linked to "determine effective promotion", and "decrease inventory costs" is linked to "optimize inventory management" (see figures 4 and 5).

Guideline 4.2. Information goals are obtained by following a top-down approximation for refining decision goals:

- How can we determine an effective promotion? Analyzing previous promotions.
- How can we optimize inventory management? Analyzing inventory levels and analyzing inventory movements.

Thus, for decision goal "determine effective promotion", an information goal is discovered: "analyze effectiveness of promotions". For decision goal "optimize inventory management", two information goals are specified: "examine inventory levels" and "study inventory movements". These are represented as goals. A means-ends link is drawn between a decision goal and an information goal (see figures 4 and 5). So, “determine effective promotion” goal is linked to “analyze effectiveness of promotions”. Decision goal “optimize inventory management” is linked to two information goals: “examine inventory levels” and “study inventory movements”.

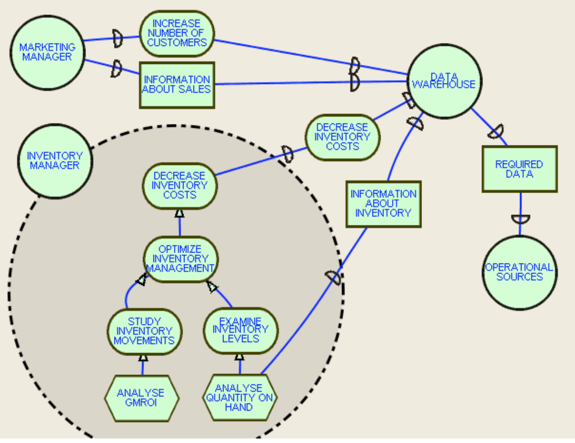


Figure 5. Strategic Rationale Model for Inventory Manager.

Guideline 4.3. Analyzing a promotion is analyzing what quantity of products is selling in which stores, on what days, and under what promotional conditions. Then, an important measure to take into account is quantity of product sold, so one task is represented: “analyze quantity sold” (see figure 4). In order to examine inventory levels, the retailer needs to analyze daily levels of available product in the store (quantity-on-hand) for making sure the right product is in the right store at the right time. Then, measuring quantity of product in the store for certain periods of time is needed. Thus, a task called “analyze quantity on hand” must be represented (see figure 5). For “study inventory movements” we need to analyze GMROI (as we describe above), so a task called “analyze GMROI” is created (see figure 5). A means-ends link must be represented between each task and its decision goal.

Guideline 5. Obtain goals, tasks, and resources of data warehouse actor and represent them in a SR model (see figure 6).

Guideline 5.1. Main goals are related to provide required information. In this case study, “provide information about sales” and “provide information about inventory” are represented as goals in the SR model (see figure 6).

Guideline 5.2. Measures according to guideline 4.3, must be represented. These measures are described in figure 6.

Guideline 5.3. Information must be provided according to a context of analysis. This context is represented as task related to main goal with means-end links (see figure 6). For “provide information about sales” we have “provide information by product”, “provide information by date”, “provide information by promotion”, and “provide information by store”. For “provide information about inventory” we have “provide information by product”, “provide information by date”, and “provide information by store”. Resources are needed to aggregate information depending of certain levels of hierarchy according to each dimension of analysis. Here we represent as resources: “product”, “date”, “promotion”, and “store”. Levels of aggregation are included in each resource.

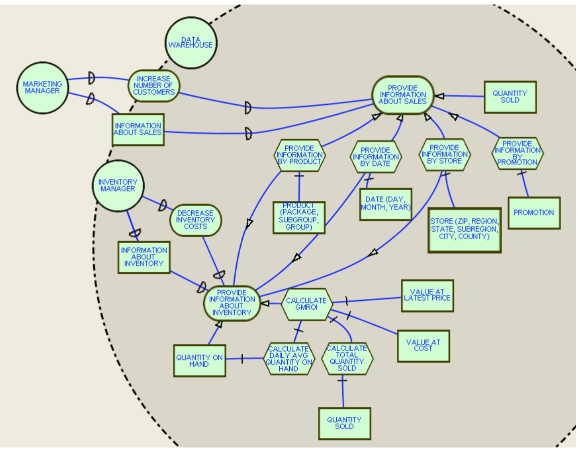

DATA WAREHOUSE
MARKETING MANAGER
INCREASE NUMBER OF CUSTOMERS
INFORMATION ABOUT SALES
PROVIDE INFORMATION ABOUT SALES
QUANTITY SOLD
PROVIDE INFORMATION BY PRODUCT
PROVIDE INFORMATION BY DATE
PROVIDE INFORMATION BY STORE
PROVIDE INFORMATION BY PROMOTION
INVENTORY MANAGER
DECREASE INVENTORY COSTS
PRODUCT (PACKAGE, SUBGROUP, GROUP)
DATE (DAY, MONTH, YEAR)
STORE (ZIP, REGION, STATE, SUBREGION, CITY, COUNTY)
PROMOTION
INFORMATION ABOUT INVENTORY
PROVIDE INFORMATION ABOUT INVENTORY
CALCULATE GMROI
VALUE AT LATEST PRICE
VALUE AT COST
QUANTITY ON HAND
CALCULATE DAILY AVG QUANTITY ON HAND
CALCULATE TOTAL QUANTITY SOLD
QUANTITY SOLD

Figure 6. Strategic Rationale Model for Data Warehouse.

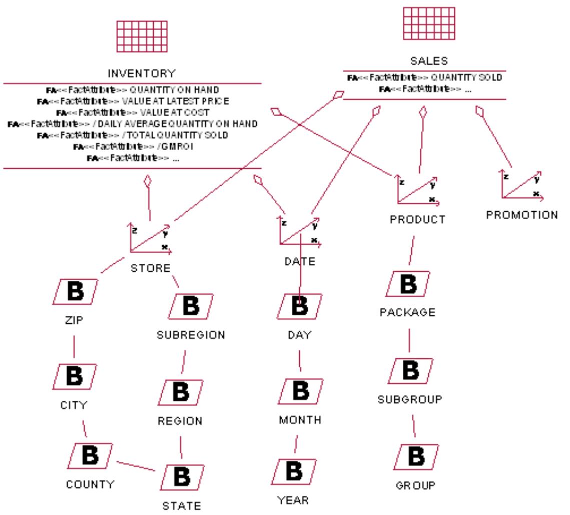
4.3. Multidimensional Model

In this subsection, our guidelines are applied to obtain a multidimensional model from the previously defined i* models (see figure 7).

Guideline 6. Two fact classes are created: from “provide information about sales” we obtain the fact class “sales”, and from “provide information about inventory” we obtain the fact class “inventory”. Then, attributes are included in each fact class: “sales” fact class has the attribute “quantity sold”, and “inventory” fact class has the following attributes: “quantity-on-hand”, “total quantity sold” (derived), “value at latest price”, “value at cost”, “daily average quantity-on-hand” (derived), and “GMROI” (derived).

Guideline 7. Dimension classes are elicited from resources which represent the context of analysis. In this case, we have to create the following dimension classes linked to "sales" fact class: "store", "date", "product", and "promotion". We have also have to link "store", "date", and "product" to "inventory" fact class.

Guideline 8. Specify levels of aggregation by means of resources which represent the context of analysis. Levels of aggregation are defined with base classes. They are all represented in figure 7.



INVENTORY
FA<<FactAttribute>> QUANTITY ON HAND
FA<<FactAttribute>> VALUE AT LATEST PRICE
FA<<FactAttribute>> VALUE AT COST
FA<<FactAttribute>> / DAILY AVERAGE QUANTITY ON HAND
FA<<FactAttribute>> / TOTAL QUANTITY SOLD
FA<<FactAttribute>> / GMROI
FA<<FactAttribute>> ...
SALES
FA<<FactAttribute>> QUANTITY SOLD
FA<<FactAttribute>> ...
STORE
DATE
PRODUCT
PROMOTION
B
ZIP
B
SUBREGION
B
DAY
B
PACKAGE
B
CITY
B
REGION
B
MONTH
B
SUBGROUP
B
COUNTY
B
STATE
B
YEAR
B
GROUP

Figure 7. Multidimensional class diagram created from the i* models.

Once we have the multidimensional schema, if we go back to the first main information requirement above-describe: "analyzing what quantity of products are selling in which stores on what days under what promotional conditions", we can easily see that this requirement can be answered by navigating the obtained multidimensional schema of figure 7. In concrete, the measure specified in the fact class comes from the resource "quantity sold", and the dimension classes and base classes come from resources "product", "date", "promotion", and "store" (see figure 6). For an overview of what multidimensional elements are created to fulfill each goal, see table 2.

5. Conclusion and Future Work

In this paper, we have presented a requirement analysis approach for understanding the data warehouse within its business context. Business goals of organizations must be understood by designers to develop a data warehouse which properly support expected decision making and allow organizations to derive business value. In order to understand business environment, we represent actors and business goals which are pretended to be fulfilled with information provided by a data warehouse using i* technique. First of all, these actors, business goals, information and their relationships are modeled in a SD model. Secondly, we have structured data warehouse goals into strategic, decision and information goals. We have built a SR model to show these goals for each actor. In this SR model, we also represent tasks and resources that actors need to fulfill goals, so we properly represent the information requirements needed to achieve all goals. Finally, we have shown how to build a multidimensional schema from these i* models by following our own approach for the conceptual design of data warehouses with UML. This multidimensional schema provides required information for fulfilling business goals. Moreover, we have also provided, based on our experience in designing real world data warehouses, a set of guidelines to correctly specify i* models, thereby avoiding an arbitrary use of them.

Immediate planned future work involves formalizing and organizing proposed guidelines into a process model. It is also planned to add quality measures to these i* models to provide more objective indicators of quality. Then, these measures must be both formally and empirically validated. On the other hand, a future interesting experiment is focused on analyzing the understandability of these diagrams with stakeholders in real world data warehouse projects (like in [13]). This experiment will allow us to validate our approach. Furthermore, we plan to add softgoals in order to gather security and quality constraints [9]. Furthermore, we also consider adding the specification of operational data sources to this approach (by adding them as actors in SR and SD models). This allows us to enrich levels of hierarchies (i.e. base classes) with attributes. Further future works refer to provide an overall methodology for data warehouse design starting from the requirement analysis phase.

6. Acknowledgements

This work has been partially supported by the METASIGN project (TIN2004-00779) from the Spanish Ministry of Education and Science, by the DADASMECA project (GV05/220) from the Valencia Government, and by the MESSENGER (PCC-03-003-1) and DIMENSIONS (PBC-05-012-2) projects from the Regional Science and Technology Ministry of Castilla-La Mancha (Spain).

Table 2. Goals and their corresponding multidimensional elements which allow their achievement.

Goals	**Multidimensional elements**			
	Fact class	***Fact attributes (measures)***	***Dimension classes***	***Base classes (levels of aggregation)***
Increase number of customers	Sales	Quantity sold	Store, date, product, promotion	Store: ZIP, city, county, subregion, region, state. Date: day, month, year. Product: package, subgroup, group.
Decrease inventory costs	Inventory	Quantity on hand, quantity sold, value at latest price, value at cost, daily average quantity on hand, GMROI	Store, date, product	Store: ZIP, city, county, subregion, region, state. Date: day, month, year. Product: package, subgroup, group.

References

[1] Agosta, L. The Essential guide to Data Warehousing. Prentice Hall, New Jersey, 2000.

[2] Böhnlein M., Ulbrich-vom Ende, A. (2000). Business Process Oriented Development of Data Warehouse Structures. In: Proceedings of Data Warehousing 2000, Physica Verlag.

[3] Bruckner R., List B., Schiefer J. (2001). Developing requirements for data warehouse systems with use cases. In Proc. 7th Americas Conf. on Information Systems (AMCIS'01), Boston, USA (2001). 329-335.

[4] Inmon, W. Building the Data Warehouse (2nd Edition). New York. Wiley & Sons.

[5] Jarke, M., Lenzerini, M., Vassiliou, Y. and Vassiliadis, P. Fundamentals of Data Ware-houses, Ed. Springer. (2000).

[6] Kimball R., Ross M. (2002) The Data Warehouse Toolkit, second edition, John Wiley & Sons. (2002).

[7] Luján-Mora S., Trujillo J., Song I-Y. (2002). Extending UML for Multidimensional Modeling. 5th International Conference on the Unified Modeling Language (UML 2002), LNCS 2460, 290-304. (2002).

[8] Luján-Mora S., Trujillo J., Song I-Y. (2002). Multidimensional modeling with UML package diagrams. 21st Intl. Conference on Conceptual Modelling (ER2002). Volume 2503 of LNCS. pp. 199-213. Springer-Verlag (2002).

[9] Paim F.R.S., Castro J.B. (2002b). Enhancing data warehouse design with the NFR Frame-work. 5th Workshop on Requirements Engineering (WER2002), Valencia, Spain. pp 40-57.

[10] Paim F.R.S., Castro J.B. (2003). DWARF: An Approach for Requirements Definition and Management of Data Warehouse Systems. 11th IEEE International Requirements Engineering Conference (RE'03). Monterey Bay, California, USA, 2003.

[11] Prakash N., Singh Y., Gosain A. (2004). Information scenarios for data warehouse re-quirements elicitation. 23th Intl. Conference on Conceptual Modelling (ER2004). Volume 3288 of LNCS. pp. 205-216. Springer-Verlag (2004).

[12] Schiefer J., List B., Bruckner R. (2002). A holistic approach for managing requirements of data warehouse systems. Proc. Americas Conf. on Information Systems, 2002.

[13] Siau K., Lee L. Are use case and class diagrams complementary in requirements analysis? An experimental study on use case and class diagrams in UML. Requirements Eng (2004) 9:229-237.

[14] Trujillo J., Luján-Mora S., Song I-Y. (2003). Applying UML For Designing Multidimensional Databases And OLAP Applications. Advanced Topics in Database Research, Vol. 2 2003: 13-36.

[15] Trujillo J., Palomar M., Gómez J., Song I-Y. Designing data warehouses with OO conceptual models. IEEE Computer, special issue on Data Warehouses 34 (2001) 66-75

[16] Van Lamsweerde A. (2000). Requirements engineering in the year 00: a research perspective. Invited paper for ICSE'2000 – 22nd International Conference on Software Engineering, Limerick, Ireland. ACM Press, New York, June 2000.

[17] Winter R., Strauch B. A method for demand-driven Information Requirements Analysis in Data Warehousing Projects. Proceedings of the 36th Hawaii International Conference on Sys-tem Sciences, 2003.

[18] Winter R., Strauch B. Information Requirements Engineering for Data Warehouse Sys-tems. ACM Sysmposium on Applied Computing (SAC'04) Nicosia, Cyprus.

[19] Yu, E. Modelling Strategic Relationships for Process Reenginering, Ph.D. thesis. University of Toronto, 1995.

[20] Yu. E. "Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering", Proc. 3rd Int. Symp. on Requirements Engineering, Annapolis, 1997, pp. 226-235.

Aligning Work Processes and the Adviser Portal Bank System

Jens Bæk Jørgensen and Kristian Bisgaard Lassen
Department of Computer Science, University of Aarhus
IT-parken, Aabogade 34, DK-8200 Aarhus N, Denmark
E-mail: jbj@daimi.au.dk, krell@daimi.au.dk

Abstract

The Adviser Portal (AP) is a new IT system for 15 Danish banks. The main goal of AP is to increase the efficiency and quality of bank advisers' work. Requirements engineering for AP includes describing new work processes that must be supported by AP using a combination of: (1) prose and informal drawings; (2) formal models; (3) graphical animation. This representation helps users and systems analysts to align new work processes and AP via early experiments in a prototyping fashion. The contribution of this paper is to present and reflect upon the analysis and description of one specific, important work process.

Topics: New requirements engineering approaches to meeting business needs; capturing and modelling business needs; from business processes to requirements.

Paper type: Full research paper (industry case)

1 Introduction

Bankdata is a Danish company that is currently developing a new system called the *Adviser Portal (AP)*. AP has been bought by 15 Danish banks and will be used by thousands of bank advisers in hundreds of bank branches. The scope of AP is to support advising private customers and small businesses. The total development effort is 15 developers in three years. The first version is planned for delivery in September 2005.

Three banks are actively involved in the development. They are test sites for pre-release versions of AP and they provide users, who participate in requirements engineering workshops, together with analysts from Bankdata. The workshops are about two related issues: (1) work processes in the banks; (2) the AP system itself. The success of AP requires alignment of (1) and (2): AP must give genuine support to advisers' work processes.

This paper is about AP's support for work processes regarding advising customers asking for loans. This is a crucial issue because, on average, an adviser in Bankdata's customer banks uses about half of her working day on tasks related to handling loan enquiries. The contribution of the paper is to present and reflect upon how the specific work process *blanc loan advise* is analysed and described, aiming at aligning it with AP.

A *blanc loan* is a simple type of loan, which can be granted without requiring the customer to provide any security. This is in contrast to, e.g., mortgage credits and car loans. Blanc loans are typically used for consumption purposes like travels, weddings, and gifts. They present a relatively high risk for the banks and have a correspondingly high interest rate.

We, the authors of this paper, have been involved in the AP project for the last half year. We have provided general consultancy advice and we have used the project as a test case for our research.

The cooperation between Bankdata and ourselves was initiated by Bankdata, who asked our institution for consultancy advice regarding (quoting from Bankdata's enquiry) "workflow and usability". A one-day meeting with nine representatives from Bankdata, including managers, analysts, architects, and programmers, was held. Here, we presented the requirement engineering technique *Executable Use Cases (EUCs)*, based on the papers [9, 11]. Bankdata got interested in testing EUCs in the AP project.

This has resulted in creation of an EUC describing the blanc loan advise work process using a combination of: (1) prose and informal drawings; (2) formal models; (3) graphical animation. The EUC has been used as a help to align the new work process with AP. In general, an EUC has similarities with a traditional high-fidelity prototype of an IT system, but an EUC also uses workflow modelling to explicitly represent the considered work processes.

The paper is structured as follows: Section 2 gives some background about AP, both the system and the development project, and about the blanc loan advise work process. Section 3 gives a brief introduction to EUCs. Section 4 presents the blanc loan advise EUC. Section 5 describes the setting for the work that is the basis for this paper. Section 6 discusses lessons learned. Section 7 is about related work. Section 8 draws the conclusions and points to future work.

2 AP and Blanc Loan Advise

The main goal of AP is to increase the efficiency and quality of bank advisers' work. Currently, prior to the deployment of AP, the advisers in Bankdata's customer banks often need information, which is scattered in many places: in different IT systems, on paper sheets in binders or in piles on a desk, on post-it notes, or even only in the mind of the adviser.

This hampers both efficiency and quality; it is time-consuming to search for information, and an adviser may, e.g., sometimes forget to call a customer when she has promised to do so. The scattering of information makes it difficult for an adviser to get an overview, both of her own current and future tasks, and of the information pertaining to a particular task. Moreover, it makes it difficult for the bank, as an organisation, to coordinate, distribute, and plan work. To address these difficulties, AP will provide a *task list* for each adviser.

Analysing and designing the task list is the responsibility of the task list working group, which consists of five users from the customer banks and four analysts from Bankdata. The group must identify which tasks to include in the task list and which information to associate with each task; they must also design the structure of the list, including hierarchical organisation of tasks and dependency between tasks.

In addition, the group must analyse and describe issues concerning the advisers' use of the task list, e.g.: (T1) designing support for concurrent tasks, recognising that advisers often do many things at the same time; (T2) designing support for suspension and resumption of tasks, recognising that advisers are frequently interrupted and suddenly need to shift to another task, e.g., when the phone rings; (T3) designing transfer of tasks between the lists of different employees, e.g., when an adviser, who leaves for vacation, transfers her tasks to a colleague.

The task list is a focal point of this paper. The blanc loan advise EUC that we will present explicitly addresses issues (T1)-(T3) above for the considered work process, i.e., for tasks related to blanc loan advise.

Analysing and designing the blanc loan advise work process, and other credit related tasks and work processes and their support by AP, is the responsibility of the credit working group, which consists of four users and five analysts.

Examples of issues to be dealt with by the credit working group are: (C1) writing a dictionary with designations [7], which fix the meaning of key terms like *grant*, *prior approval*, and *credit assessment.* (C2) identifying the tasks involved in the blanc loan advise work process and describing in which sequences the tasks can be performed; (C3) identifying the information that is needed by an adviser in the blanc loan advise work process. As we will see later, the blanc loan advise EUC has been a help to address (C1)-(C3).

From a technological perspective, AP is a system integration project: A main aim of AP is to create a consistent and coherent picture of information that reside in many different IT systems. This is an enterprise application integration problem that will be solved using the IBM WebSphere platform. The technological challenges involved in the system integration are outside the scope of this paper. Our focus is on aligning the blanc loan advise work process and AP. This involves workflow modelling, which is a central ingredient of the EUC requirements engineering technique we have used.

3 Executable Use Cases (EUCs)

An *Executable Use Case (EUC)* [9, 11] supports specification, validation, and elicitation of requirements. EUCs spur communication between stakeholders and can be used to narrow the gap between informal ideas about requirements and the formalisation that eventually emerges when a system is implemented.

An EUC consists of three tiers. Each tier represents the considered work processes that must be supported by a new system. The tiers use different representations: Tier 1 (the *informal tier*) is an informal description; tier 2 (the *formal tier*) is a formal, executable model; tier 3 (the *animation tier*) is a graphical animation of tier 2, which uses only concepts and terminology that are familiar to and understandable for the future users of the new system. Tier 3 has the potential to offer significant advantages as a means of communication [4].

The three tiers of an EUC should be created and executed in an iterative fashion. The first version of tier 1 is based on domain analysis, and the first version of tiers 2 and 3, respectively, is based on the tier immediately below.

EUCs have notable similarities with traditional high-fidelity prototypes of IT systems; this comparison is made in more detail in [1]. In [10], we describe how an EUC can be used to link and ensure consistency between, in the sense of Jackson [7], user-level requirements and technical software specifications.

An EUC can have a broader scope than a traditional UML-style use case [2]. The latter is a description of a sequence of interactions between external actors and a system that happens at the interface of the system. An EUC can go further into the environment of the system and also describe potentially relevant behaviour in the environment that does not happen at the interface. Moreover, an EUC does not necessarily fully specify which parts of the considered work processes will remain manual, which will be supported by the new system, and which will be entirely automated by the new system. An EUC can be similar to, in the sense of Lauesen [13], a task description.

4 Blanc Loan Advise EUC

The blanc loan advise EUC both has a broad scope and a task description-like nature; we now describe each of the three tiers in turn.

4.1 Informal Tier

The blanc loan advise work process begins when a customer comes to his adviser and asks for a loan. The customer mentions an amount and a purpose, e.g., that he wants to borrow 75,000 Danish Kroner (equivalent of 10,000 Euros) to finance a journey around the world.

An informal flow diagram outlining the work process is shown in Figure 1.

The adviser's first action is to check the customer's data using the customer overview and the credit overview, which will be provided by AP; these actions are represented by the boxes named `Customer overview` and `Credit overview`. Sometimes, the checks result in immediate refusal, e.g., if the adviser sees that the customer has a bad credit history.

If the adviser chooses to process the customer's enquiry further, there are three main tasks. They are represented in the informal flow diagram by the boxes named `Advising / simulation`, `Decision point`, and `Production`.

`Advising / simulation` represents the situation, where the customer and the adviser meet and negotiate the conditions for the loan. In the jargon used in the banks, *simulation* takes place, which means that the adviser does some calculations and she suggests various values for monthly payment, interest rate, and loan period to the customer.

It has not yet been decided to which extent simulation should be supported by AP. It is possible that simulation will be fully integrated in AP; it is also possible that advisers will be allowed to do simulation in a standard spreadsheet or using a pocket calculator, after AP is deployed. The `Advising / simulation` task can finish at any time; this often happens when the customer and the adviser have reached an agreement.

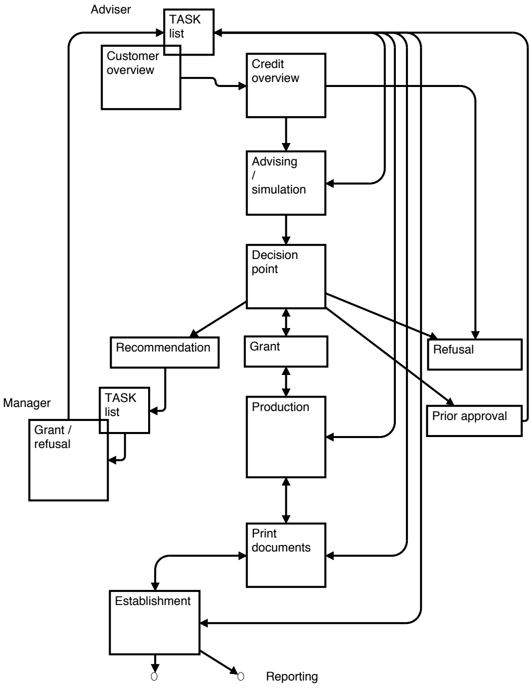


Figure 1. Informal flow diagram (translated into English) of informal tier.

`Decision point` represents that the adviser makes a decision; the adviser chooses to give the customer either: (1) a *grant*, which is a definitive yes; (2) a *refusal*, which is a definitive no; (3) a *prior approval*, which is a conditional yes, typically given in situations, where some information is missing before the loan can be finally established; or (4) a *recommendation*, which is a maybe, but where the adviser makes a recommendation to her nearest manager and asks for his approval.

The work process now branches according to the decision that was made. For the decisions *recommendation* and *prior approval*, there are backwards loops in the flow diagram, indicating further processing and another, later passage of `Decision point` to replace the maybe or the conditional yes with a definitive yes or a definitive no. For the decision *grant*, the work process continues with `Production`. That involves finishing some information gathering, e.g., getting the number of the account on which the loan amount is to be deposited. After that, some documents are printed and the loan is finally established. For the decision *refusal*, the work process terminates right after `Decision point`.

This description of the blanc loan advise work process gives an overview, but it is merely an outline of a few scenarios, or a few combination of tasks. In particular, the description does not explicitly capture concurrent tasks, suspension and resumption of tasks, and transfer of tasks between different employees. These are issues (T1)-(T3) of Section 2, which are crucial to consider in order to align the work process and the system. The formal tier of the EUC, which is outlined below, describes these three issues. More generally, the formal tier describes many more scenarios than the informal tier.

4.2 Formal Tier

The formal tier is created in *Coloured Petri Nets (CPN)*. In general, there are a number of possible choices of modelling languages to be used at the formal tier; please refer to [9] for a discussion.

We have chosen CPN because we have experience with this language and because CPN is appropriate for EUCs: CPN is well-suited for modelling of workflows or work processes [21]. CPN is mature and well-proven to describe the behaviour of large systems with characteristics like concurrency, resource sharing, and synchronisation. For a brief primer on CPN, please refer to [11], for more details to [12], and for full coverage to [8].

CPN is supported by *CPN Tools* [25], which has been used to create and execute the blanc loan advise CPN model; CPN Tools has a graphical part and includes the programming language Standard ML [15].

We will not describe the blanc loan advise CPN model in detail here. However, to give an impression of the model, Figure 2 shows one module. The full model consists of ten modules, organised in a hierarchy.

The shown module, which is the top-level of the hierarchically structured model, roughly describes the same behaviour as the informal flow diagram of Figure 1. In comparison with the informal flow diagram, however, the full CPN model has a number of useful properties: It is formal, executable, and more detailed than the flow diagram. We will discuss consequences of these properties in Section 6.

The CPN model can be compared with a board game and execution of the model with playing a token game on a playing board. The model defines the rules for when and how tokens are allowed to be moved.

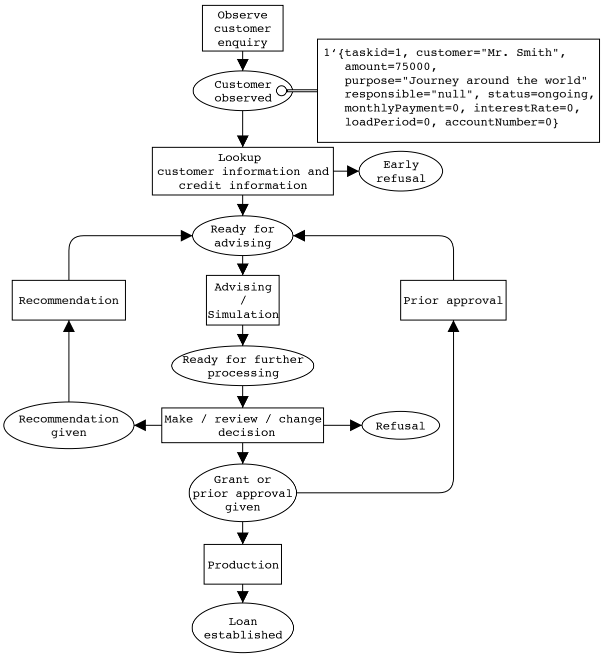

Observe customer enquiry
Customer observed
1‘{taskid=1, customer="Mr. Smith", amount=75000, purpose="Journey around the world" responsible="null", status=ongoing, monthlyPayment=0, interestRate=0, loadPeriod=0, accountNumber=0}
Lookup customer information and credit information
Early refusal
Ready for advising
Recommendation
Advising / Simulation
Prior approval
Ready for further processing
Recommendation given
Make / review / change decision
Refusal
Grant or prior approval given
Production
Loan established

Figure 2. Part of formal tier: module of CPN model (translated into English).

The tokens represent blanc loan enquiries. Each token is a record with ten fields: `taskid`, `customer`, `amount`, `purpose`, `responsible`, `status`, `monthly payment`, `interest rate`, `loan period`, and `account number`. The CPN model describes in detail when and how the attributes values can be changed. There is one token in Figure 2, modelling that customer Mr. Smith has just entered the bank because he wants to borrow 75,000 Danish Kroner for a journey around the world.

Generally, in the real world, a blanc loan advise work process is initiated when a customer comes to the bank and asks for a loan. In the CPN model, this is reflected by the creation of a new token in which the four fields `taskid`, `customer`, `amount`, and `purpose` are set initially. The `taskid` field is a modelling technical means to discriminate between different enquiries; each token in the model is created with a unique and immutable `taskid`. The three latter fields are set to hold the identity of the enquiring customer, the desired amount, and the purpose of the loan.

The `responsible` field is used to record the identity of the bank employee, who is currently responsible for handling the considered enquiry. The value in this field is changed when the enquiry (in the form of a task) is transferred from one employee to another. Thus, by inspecting the `responsible` fields of all tokens in the entire CPN model, the task lists that will be provided by AP can be constructed — or more precisely, the tasks on the lists of the involved employees pertaining to the ongoing blanc loan enquiries.

The `status` field is initially set to `ongoing` and will ultimately be set to either `established`, for a successful enquiry, or to `refused`, for an unsuccessful enquiry. There are various paths from start to finish of an enquiry. An example is: (1) After a meeting with the customer, an adviser changes the status from `ongoing` to `recommended`; (2) the adviser transfers the task to his manager; (3) the manager changes the status from `recommended` to `granted`; (4) the manager transfers the task back to the adviser; (5) the adviser gathers all the needed information, prints the necessary documents, deposits the loan amount on the designated account, and finally sets the status to `established`.

The remaining fields, `monthly payment`, `interest rate`, `loan period`, and `account number`, are set during execution of the model and reflect the adviser's information gathering and agreements with the customer.

4.3 Animation Tier

Figure 3 shows a snapshot of the animation tier; it is created with the help of Magee et al's SceneBeans animation framework [14].

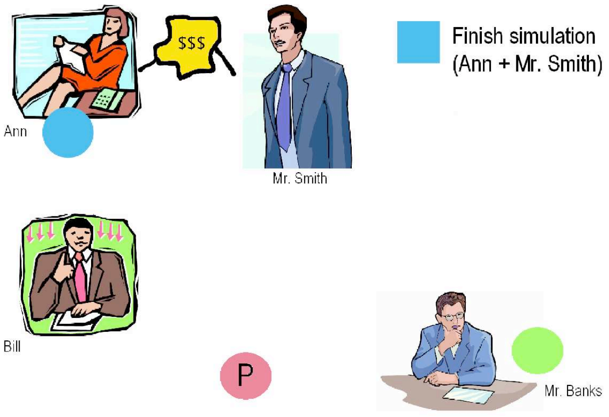


Figure 3. Snapshot of animation tier (translated into English).

The animation tier is consistent with the CPN model of the formal tier. At any time, the graphical animation represents the current state of the CPN model and mimics the token game that is played, when the CPN model is executed. Technically, the link between the CPN model and the animation tier is that the CPN model calls drawing functions when it executes. The CPN model thus causes graphical objects like customer icons and task icons to be created, moved, deleted, etc. in the animation.

Figure 3 mimics a situation in a bank in which there are two advisers, Ann and Bill, their manager Mr. Banks, and one customer, Mr. Smith. The circles represent blanc loan enquiries. i.e., they carry the same information as the ten-field record tokens in the CPN model. The circles can be clicked to display the information currently attached to an enquiry. A circle is close to the icon of the bank employee, Ann, Bill, or Mr. Banks, who is currently working on the task, and has the task on her or his task list. Some circles correspond to suspended tasks. These circles are marked with `P` (in bank jargon, these tasks are *parked*) and are put in a special area in the animation. In Figure 3, there is one suspended task. If the corresponding circle is clicked, it will move close to the icon of the adviser currently responsible for the task.

Ann is advising Mr. Smith about a new blanc loan; Ann is simulating and she has Mr. Smith's loan enquiry on her task list. Bill is currently not working on any tasks within our scope; there are no circles close to him. Mr. Banks is working on the approval of an enquiry. When he is finished, he will transfer further processing to either Ann or Bill.

The square named `Finish simulation (Ann + Mr. Smith)` can be clicked by the animation user. This is often done to mimic that the data attached to Mr. Smith's loan enquiry is satisfactory for both Mr. Smith and the bank as represented by Ann. If the square is clicked, the animation user will be prompted to make a decision on behalf of Ann. A dialog box will be shown in which the status of the loan enquiry must be set to either `granted`, `prior approval`, `recommended`, or `refused`. The consequence of the decision will be reflected in the graphical animation; examples are: If the decision is a refusal, Mr. Smith will move away from the bank, and the text "Refused" will be displayed. If the decision is a recommendation, the circle will move from the Ann icon to the Mr. Banks icon, representing the transfer of the task from Ann's task list to Mr. Banks' task list.

5 Setting

We now briefly describe the setting for the work done on creation and use of the blanc loan advise EUC that we presented above; it has been created in a number of iterations.

The first version of the informal tier in the form of prose descriptions supplemented with an informal flow diagram is based on extensive domain analysis, carried out primarily by the credit working group, but also by the task list working group. The informal tier was written and drawn by an experienced Bankdata analyst.

The first versions of the formal tier, i.e., the CPN model, were created by us. An early version was demonstrated and discussed with Bankdata analysts. Their comments were used to produce an improved and extended CPN model that was presented at a workshop in the credit working group. Based on comments from the users, Bankdata analysts and ourselves worked together to modify and extend the CPN model.

The insights gained through construction and use of different versions of the formal tier had an impact on the informal tier. The informal tier was changed five times and a new version released to the participants of the credit working group, during the half year of the AP project that this paper reports about.

When it was judged that the CPN model was in good accordance with the blanc loan advise work process, we started to design and implement the animation tier. Again, the first version was created by us. It was presented for Bankdata, and afterwards, both the formal tier and the animation tier were further developed in cooperation between Bankdata analysts and us. When the animation tier was considered sufficiently mature, it was presented and subject for discussion at a workshop in the credit working group.

6 Lessons Learned

The blanc loan advise EUC is a supplement to the documentation that Bankdata usually creates for requirements engineering; the usual documentation corresponds to the informal tier of the EUC.

Thus, the EUC is the tangible result of the work described in this paper. Additional results of our work are the lessons learned through creation and use of the EUC. We discuss five key lessons below. The first three lessons are directly related to the three issues (C1)-(C3) that was listed in Section 2 as responsibilities of the credit working group.

6.1 EUCs Support Precise Designations

A designation [7] establishes the meaning of a basic term pertaining to a development project, often a term describing something in the environment of use of a new system. The designations pertaining to the blanc loan advise work process have improved during this project. This is, of course, highly due to the general improvement in understanding that inevitably results from stakeholder discussions, e.g., at workshops in the credit working group. However, in addition, we have also experienced that the EUC has helped.

Via the formal tier, the EUC technique insists on making formal descriptions, which has contributed to more precise designations (in general, formalisation is not guaranteed to yield precision; for discussions on differences between formality and precision, see, e.g., Jackson [7] or Wieringa [23]).

At one of the workshops, it became apparent that the difference between a prior approval and a recommendation was neither entirely clear nor agreed between the workshop participants. The discussion was based on an earlier version of the informal flow diagram of Figure 1. Here, there was room allowing different workshop participants to associate different meanings with the terms. The EUC contributed to clarification; it describes in detail how what was called a prior approval is handled and how what was called a recommendation is handled, in the blanc loan advise work process.

An essential difference between the informal flow diagram and the EUC is that the latter, because of the formal tier, represents the blanc loan advise work process in an explicit and refutable way. Therefore, creating and executing the EUC have triggered discussions about what things mean and, consequently, have lead to more precise designations.

6.2 EUCs Support Precise Workflow Descriptions

The informal tier of the EUC is, indeed, an informal description of the blanc loan advise work process. The informal description has in many situations worked well as means of communication, but we have also experienced that the informal description has not always been sufficiently precise. The EUC as a whole has been an alleviation. As above, a main reason has been that the EUC insists on making formal descriptions.

As an example, at one of the workshops, it turned out that some participants had contradicting interpretations of the meaning of a split in terms of a branching arrow in the informal flow diagram (Figure 1). In an older version of the flow diagram, the branch from the decision point was indicated by a diamond symbol, whose meaning was not clear. There were also some double arcs, which caused some confusion. The possible interpretations had quite severe impact on how AP was intended to support the blanc loan advise work process. As an example, one interpretation would allow an adviser to establish a loan while a manager concurrently handles the recommendation of that loan; a second interpretation would insist that the establishment should always wait until after the manager had given a grant.

One of the authors of this paper participated in that workshop. A version of the CPN model was brought on a laptop. The author listened to the discussions and incorporated what the author thought was a sensible interpretation of the branch into the formal CPN model. This seemingly helped to reach clarification and agreement. The CPN model was changed during the workshop in a rapid prototyping-like fashion; it took a few minutes to make the change. With the formal representation of the CPN model, there was no longer room for different interpretations of the troublesome split. The one and only interpretation was determined by the formal semantics of CPN.

However, changing the CPN model, or the animation tier, is inherently more time-consuming than changing the informal tier, e.g., just by drawing an additional arrow in an informal flow diagram or changing the text in a prose description. Our EUCs cannot dynamically accommodate change as, e.g., is possible in Harel and Marelly's Play-in/out approach [6]. One of the users has commented that he sees this as the main drawback of EUCs.

6.3 EUCs Support Detailed Workflow Descriptions

The informal tier of the EUC is an abstract description of the blanc loan advise work process. Abstract descriptions are useful in requirements engineering, in particular at early stages, but as a project progresses, more details are beneficial. A detailed description can be closer to the real world subject matter being described than a more abstract description. Thus, stakeholder discussions based on detailed descriptions are likely to bring up more issues than discussions based on more abstract descriptions. The EUC has contributed to that. It has provided a more detailed description of the blanc loan advise work process than catered for by the informal tier only.

In particular, the EUC has given a basis for considering issues (T1)-(T3) of Section 2: the handling of concurrent tasks, suspension and resumption of tasks, and transfer of tasks between different employees. The informal tier alone has not allowed us to address these issues. The CPN modelling language's support for describing concurrency, resource sharing, and synchronisation has been useful.

The EUC has, e.g., catalysed discussions about whether AP should always for each adviser have one and only one current task, or if AP should support that an advisers works concurrently on a number of tasks, e.g., with one task per window in the GUI, and where a number of windows are allowed to be open at the same time. This is a design decision that is likely to highly influence whether AP is properly aligned to advisers' work processes, because advisers, indeed, often do many things at the same time. Moreover, the EUC explicitly describes when it is allowed to suspend a task and when it is allowed to transfer a task to a colleague. These are also important issues to reach agreement about in the design of AP.

The tokens representing blanc loan enquiries in the CPN model and their counterparts in the animation tier contain many details. The tokens are records with ten fields; the record type serves as a first identification of the information that is needed in the blanc loan work advise process (cf. (C3) of Section 2).

Moreover, the detailed information has catalysed discussion of a number of relevant issues, e.g., regarding AP's flexibility — because AP will be used by 15 different banks, flexibility is a high priority and the three different banks involved in the development continually make compromises and agreements.

Examples of flexibility issues that have been discussed based on the EUC are: (1) How much of a loan enquiry must be filled in before an adviser can send a recommendation to his manager? Is it required that the interest rate is completely fixed, or can it be an interval? (2) How strict does a bank require that its advisers follow written-down rules, regulations, and policies? This is important for some banks, but other banks are more relaxed. It can be common practice that an adviser exceeds the formal limits for when she must ask her manager for approval. Say, she is only allowed to grant loans up to 50,000 Danish Kroner, and she is now talking to a customer who wants to borrow 75,000. If she knows the customer well and knows her manager well, she may choose to grant the loan immediately, knowing that her manager would have approved it anyway. AP should support both the rigid and the flexible way of working, e.g., AP should not prevent the adviser in the flexible bank to proceed, but AP should prevent the adviser in the rigid bank from breaking the rules.

6.4 EUCs Support Keeping Users Properly Focused

The users have typically contributed efficiently and constructively in workshops when the subject has been their work processes. In contrast, when the subject has been the AP system itself, some users have sometimes seemed more distracted and unfocused.

Bankdata is aware of this problem and often use concrete representations, e.g., drawings and screen snapshots, as a means to get users involved in more technical discussions about requirements. This is typically effective. However, a well-recognised drawback of concrete representations is that users sometimes tend to over-focus on small details rather than paying proper attention to important overall issues [20] like the work processes to be supported.

We have seen instances of this problem. An example occurred at a workshop in the task list working group. A discussion which should be about overall issues regarding alignment of work processes and AP, unintentionally shifted to use much time and energy on minor GUI-related issues, like whether a list should be in the right side or the left side of the screen.

In some situations, the blanc loan advise EUC has helped users to keep a proper focus. The EUC explicitly describes a new work process and downplays GUI-related issues. As such, the EUC is a concrete representation that is less likely to mislead users into prematurely thinking about the GUI of a system.

Another contributing factor to occasional user distraction may be that the AP system is sometimes talked about in abstract and not always clearly defined terms like *cases*, *tasks*, and *actions* (a distinction is attempted to be made between cases, tasks, and actions; a case consists of a number of tasks, and a task consists of a number of actions). In contrast, when the subject is the users' own work processes, they are discussed in domain-specific, concrete terms like *loans*, *grants*, and *recommendations*. Moreover, the users are highly motivated, because they are the leading experts and have a genuine interest in the subject.

The EUC constitutes a concrete, tangible representation, also of abstract concepts; it explicitly provides examples of abstract concepts, like the circles in the animation tier of the blanc loan advise EUC, which represent (abstract) tasks.

6.5 EUCs Support Involvement of Users at Appropriate Times

As in many large software development projects, cooperation between users and software analysts/developers is crucial for AP, but also a challenge [17, 19]. In particular, the two parties need to communicate effectively. If not, there is a risk that project time is wasted due to misunderstandings.

A contributor to effective communication in the AP project is that the involved Bankdata analysts are quite experienced and have considerable domain knowledge; a number of the analysts have previously worked as bank advisers.

In addition, via its very structure, an EUC can contribute to effective communication, because it can support involvement of users at appropriate times. The informal tier of the EUC has been created in close cooperation between users and analysts. In contrast, the users have not been actively involved in the creation of the formal tier and the animation tier. The users have merely been sources providing feedback, allowing Bankdata analysts and the authors of this paper to produce improved versions of tiers 2 and 3.

In general, to ensure efficient use of time, we believe that users should not be involved much, if at all, in formal tier; it is not an adequate basis for discussions with users. Users should be actively involved in the creation of the informal tier and as providers of feedback to the animation tier.

To investigate this conjecture further, we presented a version of the formal tier to the users at a workshop in the credit working group. Even though the CPN model was relatively mature, it got a mixed reception. Some users were seemingly able to understand the model, follow executions, and see that it was an illustration of a future work process. On the other hand, other users did not seem to appreciate the formal model and its execution very much. When asked directly about her opinion about boxes-circles-arrows diagrams like CPN models, one of the users politely answered that this is not the way she thinks — she prefers to see screen snapshots and prototypes.

Thus, presenting the formal tier to users seemingly did not support very efficient communication. On the other hand, when the animation tier was presented, the users were immediately able to see that it represented a future bank work process and could serve as a sound basis for discussions.

7 Related Work

We now briefly discuss some examples of related work on aligning work processes and IT systems.

Our EUC technique has similarities with Wieringa's technique for using workflow modelling for requirements engineering — going from the business goal tree to identification of activities to be supported by the IT system [23]. Wieringa uses statecharts [5] to describe behaviour.

Daneva discusses how to adapt business processes and enterprise resource planning (ERP) systems [3]. AP shares a number of characteristics with ERP systems. We believe that EUCs sometimes can help to comply with some of the lessons of [3], e.g., with lesson 7: *Performing systematically requirements validation and verification is critical.*

Salinesi and Rolland address how to preserve the fit between work processes and systems, when change takes place [18]. This is an important issue, which is not explicitly addressed by the EUC technique.

Potts has in [16] surveyed various software engineering approaches. Potts discusses the varying depths with which the approaches prescribe to analyse and describe the environment of an IT system to be developed. EUCs can be created with varying depths. At one extreme, they can be low-depth descriptions in the form of sequences of interactions between users and a system right at the system's interface, corresponding to traditional UML-style use cases. In this case, an EUC may resemble a usual GUI prototype. However, EUCs may also be used to make deeper descriptions, which include potentially relevant aspects of the environment, including aspects that are not found directly at the interface between user and system. The blanc loan advise EUC is, as we have seen, of this type.

In previous work [9, 11], we have applied EUCs in the health care domain and with specific focus on development of pervasive IT systems [22]. In comparison, the present paper is about requirements engineering in a different domain — in banks — and it is about requirements engineering for a traditional (non-pervasive) desktop based system. Even though the two domains are quite different, they also have a number of similarities. Some of the basic problems at hospitals and in banks are common. Indeed, many issues that are important for a new system like AP are very general and are found in many domains: providing employees an overview of tasks, handling concurrent tasks, interruptions, coordination between different employees, etc. We have demonstrated in this paper how EUCs can be used to analyse and describe such issues.

Another similarity is the categories of users we have worked with in the health care and the banking domain. In both cases, the animation tier of the EUC has proved to be not only nice to have, but essential. A number of nurses and bank advisers we have worked with do not appreciate formal models, but graphical domain-specific animations have proved to be effective means of communication.

8 Conclusions and Future Work

We have described alignment of a new work process with the Adviser Portal (AP) bank system via application of Executable Use Cases (EUCs). We have reported and discussed a number of lessons learned. These lessons are of a nature that make us believe that they generalise to other projects than AP. In continuation of the work described in this paper, we are working in a number of directions.

The top priority in the ongoing work is to relate the EUC descriptions to workflow descriptions inside IBM Process Choreographer. When AP is deployed, some of tasks that are carried out manually in the banks today will be automated by the AP workflow engine, which is IBM Process Choreographer. As we have seen, the blanc loan advise EUC is similar to a task description [13]. It describes what advisers and AP must do together. In this way, the EUC includes both activities of advisers that are not to be supported by AP and activities that are candidates to be supported. In comparison, a workflow in Choreographer is an automation of parts of work processes; to create this workflow assumes that it is known what to automate. Thus, the EUC is a broader description than a workflow description in Choreographer. We are investigating how to use EUCs as basis to discuss and identify which of the current manual tasks in the banks that should be automated by Choreographer inside AP.

We are considering to make it possible to interact with EUCs via the Internet. This will address a practical problem: The users, who participate in the working groups in the AP project, are busy and geographically distributed. It is not always easy to get them together at meetings, at the same time and at the same place. If the users can interact with the EUC remotely, e.g., from their home banks, this may be an alleviation. However, it is yet to be seen how effective that approach is. It is probably a more limited experience than to be together with other stakeholders at a meeting, but may still yield valuable benefits.

We are also currently working on establishing the EUC technique to be used more broadly by Bankdata; this involves arguing the business case for Bankdata management. We are encouraged by the Bankdata analysts we have worked with, who see the EUC technique, or something similar, as useful and promising. In particular, the analysts see EUCs as a help to record and make decisions explicit; in this way, the decisions can be subject for early discussions with and validation by users. Therefore, EUCs have potential to reduce the amount of rework that Bankdata sometimes needs to do in projects, often caused by lack of genuine user validation and, as a consequence, misunderstandings between users and analysts.

A crucial issue in arguing the business case is to consider the cost-effectiveness of EUCs. Tier 1 of the blanc loan advise EUC reflects the requirements engineering activities that Bankdata usually do. These activities have included dozens of meeting in the credit working group and task list working group; most meetings have had about ten participants and have run for one full or two full days. In general, thousands of person hours have been put into the domain analysis that is necessary for tier 1. Tiers 2 and 3 of the EUC has been created in approximately 120 person hours. Thus, creation of the EUC (all three tiers) has had a relatively low additional cost, compared to the cost of usual requirements engineering in Bankdata's projects.

With the lessons learned discussed in Section 6, we have demonstrated that the EUC has given some benefits as well. Ultimately, we hope that we can observe a good alignment between AP and the blanc loan advise work process, when AP is deployed. However, even if this observation is made, it may be difficult to assess the particular benefit of EUCs (or any other requirements engineering technique). On the long and complex path from initial requirements engineering to implementation of a system, there are many factors, which contribute to the quality of the system. To make a reliable evaluation of the impact of one particular factor seems inherently difficult.

Acknowledgements

We thank Bankdata for allowing us to participate in the AP project. We thank the analysts we have worked with, in particular Gert Schmidt Sofussen and Lisbeth Gravesen, who have contributed significantly to the work described in the paper. We thank our research group for help, in particular Michael Westergaard for providing the tool used to make the animation tier and Søren Christensen for fruitful discussions.

References

[1] C. Bossen and J. Jørgensen. Context-descriptive Prototypes and Their Application to Medicine Administration. In *Proc. of Designing Interactive Systems (DIS) 2004*, pages 297–306, Cambridge, Massachusetts, 2004. ACM Press.

[2] A. Cockburn. *Writing Effective Use Cases.* Addison-Wesley, 2000.

[3] M. Daneva. Lessons Learnt from Five Years of Experience in ERP Requirements Engineering. In Wieringa [24], pages 45–54.

[4] A. Gemino. Empirical Comparisons of Animation and Narration in Requirements Validation. *Requirements Engineering*, 9:153–168, 2004.

[5] D. Harel. Statecharts: A Visual Formalism for Complex Systems. *Science of Computer Programming*, 8:231–274, 1987.

[6] D. Harel and R. Marelly. Specifying and Executing Behavioural Requirements: The Play-in/Play-out Approach. *International Journal on Software and Systems Modeling*, 2(2):82–107, 2003.

[7] M. Jackson. *Problem Frames — Analyzing and Structuring Software Development Problems.* Addison-Wesley, 2001.

[8] K. Jensen. *Coloured Petri Nets – Basic Concepts, Analysis Methods and Practical Use. Vol. 1, Basic Concepts.* Monographs in Theoretical Computer Science. An EATCS Series. Springer, 1992.

[9] J. Jørgensen and C. Bossen. Requirements Engineering for a Pervasive Health Care System. In Wieringa [24], pages 55–64.

[10] J. Jørgensen and C. Bossen. Executable Use Cases as Links Between Application Domain Requirements and Machine Specifications. In *Proc. of 3rd International Workshop on Scenarios and State Machines (at ICSE 2004)*, pages 8–13, Edinburgh, Scotland, 2004. IEE.

[11] J. Jørgensen and C. Bossen. Executable Use Cases: Requirements for a Pervasive Health Care System. *IEEE Software*, 21(2):34–41, 2004.

[12] L. Kristensen, S. Christensen, and K. Jensen. The Practitioner's Guide to Coloured Petri Nets. *International Journal on Software Tools for Technology Transfer*, 2(2):98–132, 1998.

[13] S. Lauesen. Task Descriptions as Functional Requirements. *IEEE Software*, 20(2):58–65, 2003.

[14] J. Magee, N. Pryce, D. Giannakopoulou, and J. Kramer. Graphical Animation of Behaviour Models. In *Proc. of 22nd International Conf. on Software Engineering*, pages 499–508, Limerick, Ireland, 2000. ACM Press.

[15] R. Milner, M. Tofte, R. Harper, and D. MacQueen. *The Definition of Standard ML.* MIT Press, 1997.

[16] C. Potts. From "Requirements Engineering" to "Design for Usefulness". In *Proc. of 4th International Symposium on Requirements Engineering*, page 189. IEEE CS, 1999.

[17] S. Robertson and J. Robertson. *Mastering the Requirements Process.* Addison Wesley, 1999.

[18] C. Salinesi and C. Rolland. Fitting Business Models to System Functionality Exploring the Fitness Relationship. In *Proc. of 15th International Conf. on Advanced Information Systems Engineering (CAiSE)*, volume 2681 of *LNCS*, pages 647–664, Klagenfurt, Austria, 2003. Springer.

[19] A. Sutcliffe. *User-Centred Requirements Engineering: Theory and Practice.* Springer, 2002.

[20] A. Sutcliffe. Scenario-based Requirements Engineering. In Wieringa [24], pages 320–329.

[21] W. van der Aalst and K. van Hee. *Workflow Management: Models, Methods, and Systems.* MIT Press, 2002.

[22] M. Weiser. The Computer for the 21st Century. *Scientific American*, 265(3):94–104, 1991.

[23] R. Wieringa. *Design Methods for Reactive Systems: Yourdon, Statemate, and the UML.* Morgan Kaufmann, 2003.

[24] R. Wieringa, editor. *Proc. of 11th International Requirements Engineering Conf.*, Monterey Bay, California, 2003. IEEE CS.

[25] CPN Tools. www.daimi.au.dk/CPNTools.

Correlating Architecture Maturity and Enterprise Systems Usage Maturity to Improve Business/IT Alignment

Claudia Steghuis, Maya Daneva, Pascal van Eck
University of Twente, Centre for Telematics and Information Technology
P.O. Box 217, 7500 AE Enschede, The Netherlands
Email c.steghuis@ewi.utwente.nl, m.daneva@utwente.nl, p.vaneck@utwente.nl

Abstract

This paper compares concepts of maturity models in the areas of Enterprise Architecture and Enterprise Systems Usage. We investigate whether these concepts correlate, overlap and explain each other. The two maturity models are applied in a case study. We conclude that although it is possible to fully relate constructs from both kinds of models, having a mature architecture function in a company does not imply a high Enterprise Systems Usage maturity.

1. Introduction

Current markets are highly competitive, making it very important to rapidly respond to changing business circumstances [14]. By optimizing business processes, the efficiency and effectiveness of a company can be increased. In today's companies, transactions have to be made in real-time, while communicating with customers and suppliers. To do this, the information systems in any company should have the latest data available, and therefore should also be integrated with each other.

Different approaches to integrating information systems and/or business processes have emerged, like data warehouses, applications of Enterprise Application Integration technology, and information systems supporting the entirety of business processes in a company. The latter are called Enterprise Resource Planning (ERP) Systems and were first specialized for manufacturing companies. Nowadays, these systems can support businesses in almost all sectors and are often referred to as Enterprise Systems (ES) [6, 19].

Improving enterprise integration with these solutions is a difficult task as it brings along many changes in a company at both organizational and IT infrastructure levels. Many ES implementations are not finished in time and within budget and, often, the desired business benefits are not realized [16]. Therefore, implementation of ES is an important field of study in which a lot of methods are now being proposed to improve current practice. In this paper, we contribute to this field by investigating the relationship between maturity models for ES usage and enterprise architecture. We have chosen to look at ES usage in relation with enterprise architecture because experiences from more and more companies indicate that an ES perspective alone is not enough [12]. When all major companies in a market adopt ESs, or event the same vendor's ES, the competitive gain resulting from using an enterprise system for one company is low [6]. Moreover, some business processes are just not suitable to fit in with an ES. Consequently, companies need to integrate, but also need to differentiate in their systems. Apart from an ES, companies also have legacy systems that add up to a complex ES implementation.

We use the term 'enterprise architecture' to refer to the constituents of an enterprise at both the social level (roles, organizational units, processes, etc.) as well as the technical level (information technology and related technology), and the synergetic relations between these constituents. Thus, enterprise architecture explains how the constituents of an enterprise are related and how these relations jointly create added value.

Capability maturity models (CMMs) provide a method to gain control over IT processes and improve them. The benefits of these models lie in the systematic use of practices to identify weaknesses, strengths, and improvement activities in IT-processes [13]. The models also assist in managing improvements by providing assessment standards that help express the maturity of the organisation in a scale of five maturity levels [15]. In the architecture field, different maturity models have been developed, called Architecture Capability Maturity Models (ACMMs) [7, 10, 23]. These models specify key components of productive enterprise architecture processes and pay very little attention to the possible integration solutions.

Furthermore, in the field of ES implementation, empirical research efforts by Holland and Light [12] as well as Markus et al. [16] have yielded staged maturity models for ERP systems use. Their value is in providing roadmaps for understanding the evolution of ERP systems in adopting organizations. Such a model is divided in stages of maturity in the use of an ES and illustrates the different challenges organizations cope with while and after implementing an ES. These models do not focus on IT processes as a whole, but on the ES implementation and use only.

When combining ERP usage maturity models with the ACMMs, a framework can be created to assess how well business processes and systems are aligned in ERP adopting organizations. Such a framework can also serve as a vehicle to integrate formal business case analysis into the process of engineering the requirements for ESs as it would help organizations focus on the business value they expect to achieve from the ESs and associated business changes [6]. Business cases specify real-life problems that ERP-adopters confront and the types of process, competitive, or financial capabilities they will have when implementation is over. Carrying out a business case analysis is recognized as a vital prerequisite for a successful ERP RE process [4] and is a common practice in organizations that were successful in aligning their ESs to business strategy [1]. Specifically, our efforts in combining ACMMs and ES usage maturity models are aimed at answering the following research question: In what way is architecture maturity linked to ES usage maturity? To uncover the interplay between these two classes of maturity models, we first compare their assessment dimensions and then contrast them by using real-life experiences.

The remainder of this paper is organized as follows: Section 2 places the topic of architecture and ES usage maturity in the broader context of enterprise integration (EI) and discusses the role of ES in it. Section 3 describes our research approach. Section 4 and Section 5 provide background information on the concepts of architecture maturity and ES usage maturity, respectively. Specific instances of each of the two model classes are discussed as well. In Section 6, relations between these two classes of models are identified. Then, an ACMM and an ES Usage Model are applied in a case study in Section 7. Section 8 outlines our conclusions and future research plans.

2. Background

The literature of today [14, 17, 19] reports on three approaches for systems integration: data warehousing, ES, and Enterprise Application Integration middleware. The data warehousing approach implies that the data of all systems in an organization are integrated in one 'warehouse' that makes it possible for systems to share data and do a variety of data analyses. The systems themselves are not supposed to be changed [17]. In contrast, ERP systems were the first, in which business functions were integrated to streamline data flows across business functions such as logistics, accounting, and human resources [14]. In the second half of the 90s these systems were extended with applications that supported business processes beyond the borders of one organisation. This was necessitated by the urge many companies had to optimize processes with suppliers and customers. Cross-organizational integration was further streamlined with the availability of the Internet. Also, today's cross-organizational ESs offer a combination of internal and external integration capabilities and make it possible for ERP adopters to seamlessly participate in virtual networks in which Enterprise Application Integration is used to let the ESs of different companies communicate with each other and with other systems [19].

Enterprise Application Integration is a business computing term for the plans, methods, and tools aimed at modernizing, consolidating, and coordinating the computer applications in an enterprise [24]. It is not a piece of software which is installed to work directly out of the box, but it is a useful method for planning how to integrate systems. It aims at bringing together business processes, applications, data, and platforms [9] in order to produce a flexible and agile information architecture, which permits rapid responses to new business opportunities [14]. Integration is achieved by using middleware and by applying different topologies. Typical business benefits of Enterprise Application Integration are cycle time and cost reductions as well as cost containment.

Intra- and inter-organizational integration through ESs is a very complex task, and case studies suggest that approximately 90% of the businesses did not completely succeed in this [2]. Common failure patterns that ES adopters indicate are (i) failure to meet project goals within specified time and budget and (ii) misalignments between organization's processes and data flows and the ones embedded in the ES. Therefore, for many companies it is also important to have a technology, for example Enterprise Application Integration middleware, which makes it possible to integrate both ES and legacy applications with each other. The ultimate objective of such integration is to ensure a relationship between business and IT decision making processes so than IT and business functions adapt their strategies together [15]. For companies to be able to assess where they are in business/IT alignment and what they can do to improve it, comprehensive vehicles in the form of maturity models should be available for architects to use. In the scope of our research, we cover two classes of models that are good candidates to serve as such vehicles, namely Architecture Maturity Models and ES Usage Maturity Models. Our choice of these models is dictated by our research context, namely the use of ES as enabler for intra- and inter-organizational integration.

3. Research Method

The goal of our study is to collect information that would help us assess the interplay of architecture maturity and ERP usage maturity in an ERP adopting organization. Since research studies in architecture maturity and studies in ERP usage maturity have been done in isolation from each other and research has been focused either on organization-specific architecture or ERP aspects, there is a distinct challenge to develop a research model that adopts the most appropriate constructs from prior research and integrate them with constructs that are most suitable for our context. Given the lack of research on the phenomenon we are interested in and the fact that the boundaries between phenomenon and context are not clearly evident, it seems appropriate to apply a qualitative approach to our research question. Specifically, we chose to use an approach based on the positivist case study research method [8, 26]. We have chosen this method for several reasons: (i) it was found particularly well-suited to IS research situations in which an in-depth investigation is needed, but in which the phenomenon in question can not be studied outside the context in which it occurs, (ii) it offers a great deal of flexibility in terms of research perspectives to be adopted and qualitative data collection methods, and (iii) case studies open up opportunities to get the subtle data we need to increase our understanding of complex IS phenomena such as ERP adoption and architecture.

Our analytical approach had three main objectives, namely: (i) to identify how existing architecture frameworks and ES usage models stand to each other, (ii) to assess the possible mappings among their assessment criteria, and (iii) to examine if the mappings between architecture maturity assessment criteria and the ERP usage maturity criteria can be used to judge the ERP usage maturity in an ERP adopting organization, provided architecture maturity of this organization is known.

The research approach involved five stages:

1. Literature survey and mapping assessment criteria of existing architecture maturity models.
2. Literature survey of existing ERP usage maturity models.
3. Identification of assessment criteria for architecture and ERP usage maturity that seem (i) to overlap, (ii) to correlate, and (iii) to explain each other.
4. Selection and application of two specific maturity models to real-life organizational settings.
5. Post-application analysis to understand the relationships between the two maturity models.

We discuss each of these stages in more detail in the sections that follow.

For the purpose of our research, the unit of analysis [26] is the ERP-adopting organization. We investigate two aspect of the ERP adopter: (i) the maturity of their architecture function and (ii) the maturity of the ERP usage. Our approach involves the use of qualitative architecture assessments and ERP usage assessments, architecture deliverables, ERP requirements documents, and project team members' observation data, to explore, understand, and explain the relationship between maturity in architecture and maturity in ERP usage.

4. Architecture Maturity

The notion of maturity was first introduced by IBM and, in early 90s, was extended and elaborated in terms of capability maturity models (CMMs) that are formal ways to gain control over and improve IT-related processes as well as to assess organization's development competence [20]. Today's ACMMs follow in structure and logics the original CMM. One of the first ACMMs is the IT ACMM of the Department of Commerce (DoC) of the USA. The goal of this model is to optimize architecture-related processes by identifying weak areas and providing an improvement path [7]. Furthermore, there are models linked to the Balanced Score Card concept [10] and models for extended-enterprise-architects [23]. All these models have five or six levels of maturity that vary from initial to optimized or measured. The extent to which these models pay attention to business issues varies widely. When we compared the different ACMMs to each other (see Figure 1), the Information Technology Balanced Score Card (IT BSC) maturity model was chosen as our point of reference. We chose it because this model rests on four viewpoints that make it possible to jointly consider both business and IT issues in organizations. As the two main reasons for failures in ES-implementations are organizational resistance to change and lack of top management commitment [27], we felt that it was important to use as a reference point a model giving enough attention to business issues. The four viewpoints in the IT BSC model are defined as follows: 'Customer Orientation' is about how the IT should appear to the internal questions, 'Corporate Contribution' is the contribution that IT can have to company's success, 'Operational Excellence' tells which services and processes should be supported by IT, and 'Future Orientation' focuses on the ability to change and improve the IT to better add up to the company's success [10]. The IT BSC maturity model includes five stages, in which these four areas should be better managed and optimized. We compared the assessment criteria of the IT BSC model to the different architecture criteria as defined in the Department of Commerce IT Architecture Capability Maturity Model (DoC ACMM) [7] and the Extended Enterprise Architecture Maturity Model (E2AMM) [23]. We arrived at the mappings in Figure 1.

IT BSC MM	DoC ACMM	E2AMM
Customer Orientation		Extended Enterprise Involvement
	Operating Unit Participation	Business units involvement
		Enterprise Program Management
Corporate Contribution	Business Linkage	Business & Technology Strategy Alignment
	Senior Management Involvement	Executive Management Involvement
	Governance	Strategic Governance
	IT investment & Acquisition Strategy	Enterprise budget & Procurement strategy
		Holistic Extended Enterprise Architecture
Operational Excellence	Architecture Process	Extended Enterprise Architecture Programme Office
	Architecture Development	Extended Enterprise Architecture Development
	Operating Unit Participation	
	Architecture Communication	Enterprise Program Management
	IT security	
	IT investment & Acquisition Strategy	Enterprise budget & Procurement strategy
		Extended Enterprise Architecture Results
Future Orientation	Architecture Development	Extended Enterprise Architecture Development

Figure 1 ACMMs compared and contrasted

The DoC ACMM is developed to make judgements of IT processes to evaluate the current organization and what the future should bring [7]. The E2AMM 'provides a path for enterprise architecture and procedural improvements within an organization' [23, p1]. There exist also other architecture maturity models, for example the IS/ICT Management Capability Maturity Framework [21]. These models work with assessment constructs which are very similar to the ones from the DoC ACMM and therefore we do not discuss them here.

5. ES Usage Maturity

Our review of the ERP literature points out that ES Usage maturity models are meant as theoretical frameworks for analysing, both retrospectively and prospectively, the business value of ES. As system evolution adds the concept of time to these frameworks, they tend to structure 'ES experiences' in terms of stages, starting conditions, goals, plans and quality of execution [16]. For example, the model by Markus et al [16] allocates elements of ES success to three different points in time during an organization's experience: (i) the 'project phase' in which the system is configured and rolled out, (ii) the 'shakedown phase' in which the organization goes live and integrates the system in their daily routine, and (iii) the 'onward and upward phase', in which the organization gets used to the system and is going to implement additions. Success in the shakedown phase and in the onward and upward phase is influenced by ES usage maturity. For example, observations like (i) a high level of success in improvements in business results, (ii) employees' willingness to work with the system, and (iii) adopting new releases, are directly related to a high level of ES usage maturity. Next, the staged maturity model by Holland and Light [12] suggests three stages and is based on five theoretical constructs as shown in the Figure 2. The model does not yet pay enough attention to certain determinants of the ES architecture, namely, cost, entropy, complexity, flexibility, and competitiveness. However, because these do not affect the way we approach our research question, we would not discuss them as part of this paper.

6. Relations between architecture maturity and ES Usage maturity

Our hypothesis is that the constructs in the AMM and the ES UMM differ, correlate but do not explain one another. That there is a relationship between architecture maturity and ES usage also becomes evident from the fact that the two types of models use the same factors to assess either maturity or alignment, for example, factors like governance, processes, communication, vision and executive sponsorship. These correlating factors are discussed in the sections that follow. We start with the constructs of the ES Usage Maturity Model (ES UMM) and we link them to the constructs of the IT BSC and DoC AMM. For clarity, the acronyms of the names of these models are given in brackets appended to the name of each construct.

Constructs	Stage 1	Stage 2	Stage 3
Strategic Use of IT	- Retention of responsible people - no CIO (anymore) - IS does not support strategic decision-making	- ES is on a low level used for strategic decision-making - IT strategy is regularly reviewed - ES Importance is high	- Strong vision - IT strategy through whole organization - CIO in the senior management team
Organizational Sophistication	- no process orientation - very little thought about information flows - no culture change	- significant organizational change - improved transactional efficiency	- process oriented organization - top level support and strong understanding of ERP-implications
Penetration of the ERP System	- the system is used by less than 50% of the organization - cost-based issues prohibit the number of users - few formalized training - staff retention	- most business groups / departments are supported - high usage by employees	- truly integrated organization - users find the system easy to use
Drivers & Lessons	Key drivers: - priority with management information - costs Lessons: - mistakes are hard to correct - high learning curve	Key drivers: - reduction in costs - replacement of legacy systems - integrating all business processes - improved access of management information	Key drivers: - single supply chain - replacement of legacy systems
Vision	- no clear vision - simple transaction processing	- performance oriented culture - internal and external benchmarking	- higher level uses are identified - other IT systems can be connected

Figure 2 ES Usage Maturity Model (based on [12])

6.1. Strategic use of IT

The first construct of the ES UMM is called 'The strategic use of IT' and deals with the importance of the IT function within a business [12]. This construct corresponds to the constructs 'Corporate contribution' (IT BSC MM) and 'Operational excellence' (IT BSC MM). Figure 3 shows the characteristics from the other ACMMs that are related to these two areas. 'Business linkage' (DoC ACMM) and 'Business/technology strategy alignment' (E2AMM) are important factors in this construct as these determine how the strategic goals of business and IT are related.

'Architecture process and development' (DoC ACMM) are the ones from 'Operational excellence' (IT BSC MM) that are related to this construct. These say how the architecture process is organized and what kind of developments is expected. 'IT investment and acquisition strategy' (DoC ACMM) is also a characteristic that falls within this construct.

6.2. Organizational Sophistication

This construct describes how the structure of the organization has changed after the ES implementation. Change is unavoidable due to the fact that an ES imposes its embedded processes and data management procedures to the ES-adopter. Also, either the business processes in the organization have to be adapted to the embedded processes in the system (the so called 'best practices'), or the ES has to be customized to the diverse processes of the company [12]. This is consistent with the strategy of the organization and with the 'Organizational sophistication' construct. Often, it is less expensive to change the business process to fit the system than the other way around. Customizing the ES can bring along problems with future versions of the software but sometimes an organization decides to change the software because their process is so specific or because of strategic advantages: when every organization uses the same ES, it is hard to compete [6]. The 'Organizational sophistication' construct has no specific equal within the ACMMs, but it can be mapped onto what is meant in the constructs of 'Corporate contribution' (IT BSC MM) and 'Architecture communication' (DoC ACMM) as all of these reflect strategic decisions being made.

6.3. Penetration of ERP

The penetration of the system in the organization can be measured by three indicators: (i) the number of employees who use routinely the system as part of their daily duties, (ii) the number of functions that are covered, and (iii) the retention of legacy systems [12]. This construct can be partially mapped onto the constructs of ‘Customer orientation’ (IT BSC MM) and ‘Operational excellence’ (IT BSC MM). The factors of ‘Participation of the Employees’ (DoC ACMM) and the ‘Involvement of the senior management’ (DoC ACMM) are important for the use of the system as experiences indicate that many ES-implementations fail due to a lack of senior management involvement [22]. ‘Architecture communication’ (DoC ACMM) also is important for the employees to understand why to use the system. This concept discusses the level of penetration of the architecture documents.

6.4. Vision

The vision defines the strategic potential for the ES and what the use of the system is [12]. This is about the strategy of the organization. In this construct, the factors of ‘Business linkage’ (DoC ACMM) and ‘Business/technology strategy alignment’ (E2AMM) are also important because these describe the relationship between the construct ‘Vision’ and the construct ‘Strategic use of IT’, both of the ES UMM. Based on the above consideration, we can conclude that these two constructs are interrelated. The ‘Vision’ also impacts on the type and the number of standards and rules used within the IT. ‘Governance’ (DoC ACMM) is the characteristic that deals with these standards and rules.

6.5. Drivers & Lessons

This construct deals with the business drivers in the implementation and the lessons learned afterwards [12]. It follows the implementation process and can therefore be compared to the architecture process. There is no dimension in the ACMMs that is exactly the same, but the concept of ‘Architecture process’ (DoC ACMM) may well include analysis of business drivers and use of lessons learnt.

6.6. Evaluation

The ES UMM constructs are in essence all related to the architecture maturity constructs. With exception of ‘Penetration of the ERP’, the ES UMM constructs refer to the strategy of the organization as the ACMM do. Therefore our logical conclusion is that to achieve ES usage maturity, the same constructs can be used as to achieve architecture maturity.

ES UMM construct	**Related ACMM constructs**
Strategic Use of IT	**IT BSC MM:** • Corporate Contribution • Operational Excellence
	DoC ACMM • Business Linkage • Architecture Process • Architecture Development • IT investment & Acquisition Strategy
	E2AMM • Business & Technology Strategy Alignment • Holistic Extended Enterprise Architecture • Extended Enterprise Architecture Programme Office • Extended Enterprise Architecture Development • Enterprise budget & Procurement strategy
Organizational Sophistication	**IT BSC MM:** • Corporate Contribution
	DoC ACMM: • Architecture Communication
	E2AMM: ***Not covered***
Penetration of the ERP	**IT BSC MM:** • Customer Orientation
	DoC ACMM: • Operating Unit Participation • Senior Management Involvement • Architecture Communication
	E2AMM: • Business units involvement • Executive Management Involvement • Extended Enterprise Involvement
Vision	**IT BSC MM:** • Future Orientation
	DoC ACMM: • Business Linkage • Governance
	E2AMM: • Business & Technology Strategy Alignment • Strategic Governance
Drivers & Lessons	**IT BSC MM:** ***Not covered***
	DoC ACMM: • Architecture Process
	E2AMM: Extended Enterprise Architecture Results
Not Covered	**DoC ACMM:** • IT Security
	E2AMM: • Enterprise Program Management

Figure 3 Comparing constructs of the ES UMM with the ACMMs

7. The case study

In this section, the ES UMM and the DOC ACMM are applied to a case study of a company implementing an ES. For this purpose, we use the ERP experiences at Telus Mobility, a Canadian communications company [4, 5]. This company completed 13 ERP projects within five years.

7.1. Architecture maturity

In 2000, after a series of corporate mergers, the company initiated a strategic planning exercise as part of a major business processes and systems alignment program. A key component of the strategic planning effort was the assessment of architecture maturity and the capability of the organization's architecture process. The DoC ACMM was used among other standards as a foundation and an assessment process was devised based on a series of reviews of (i) the architecture deliverables created for small, mid-sized and large projects, (ii) architecture usage scenarios, (iii) architecture roles, (iv) architecture standards, and (v) architecture process documentation. The nine maturity assessment aspects of the DoC ACMM (see the second column in Figure 2) were mapped into the types of architecture deliverables produced and used at the company. The highlights of the assessment are listed below:

- **Operating unit participation:** Since 1996, a business process analyst and a data analyst have been involved in a consistent way in any business (re)-engineering initiative. Process and data modeling were established as functions, they were visible for the business, the business knew about the value the architecture services provided and sought architecture support for their projects. Each core process and each data subject area had a process owner and a data owner. Their sign-off was important for the process of maintaining the repositories of process and data models current.
- **Business linkage:** The architecture deliverables have been completed on behalf of the business, but it was the business who took ownership over these deliverables. The architecture team was the custodian of the resulting architecture deliverables, however, these were maintained and changed based on requests by the business.
- **Senior management involvement / Governance:** All midsized and large projects were strategically important, as the telecommunication industry implies a constant change and a dynamic business environment. The projects were seen as business initiatives rather than IT projects and has strong commitment from top management.
- **IT investment and acquisition strategy:** IT was critical to the company's success and market share. Investments in applications were done as a result of a strategic planning process.
- **Architecture process:** The architecture process was institutionalized as a part of the corporate Project Office. It was documented in terms of key activities and key deliverables. It was supported by means of standards and tools.
- **Architecture development:** All major areas of business, e.g. all core business processes, major portion of the support processes, and all data subject areas were architected according to Martin's methodology [18]. The architecture team has a quite good understanding of which architecture elements were rigid and which were flexible.
- **Architecture communication:** Architecture was communicated by the Project Office Department and by the process owners. The IT team has not been consistently successful in marketing the architecture services. There were ups and downs as poor stakeholder involvement impacted the effectiveness of the architecture team's interventions.
- **IT security:** IT Security was considered as one of the highest corporate priorities. The manager of this function was part of the business, and not of the IT function. He reported directly to Vice-President Business Development.

7.2. ES usage maturity

To assess the ES usage maturity in this case, the ES UMM (Figure 3) is used. Throughout the first three projects, the organization was in the beginning of stage 1 of this model. Before the implementation was executed, little thought was given to how the organization should handle these projects in the long-term. During the first few projects, it became clear to the project implementation team that there was a lot of learning on the job, and this was used to reflect on success and failure experiences [5] and get more insights into the intricacies of the ES implementation. At the time of writing, Telus is extending its SAP portfolio and is currently in stage 2 of the ES UMM. Details on the qualitative assessments of the ES usage maturity with respect to the five constructs are discussed as follows:

- **Strategic use of IT:** The company started with a strong IT vision, the senior managers were highly committed to the projects. The CFO was responsible for the choice for an enterprise system, and therefore, moving to a new ERP platform was a business decision. The company also had their CIO on board. The SAP package was not implemented in all areas because this could have reduced their competitive advantage. As quality of service provisioning and client intimacy were the key priorities for the company, they decided to combine the SAP applications with a business-specific package (namely AMDOCS) for their competitively important domain of wireless service delivery (including client activations, client care, and rate plan management). This made the choice for SAP a well-considered one. The management team now decided to implement three additional SAP modules and, thus, more and more business processes are covered in the ES.

- **Organizational Sophistication:** Business users wanted to keep processes diverse, however the system pushed them towards process standardization and this led to cultural conflicts. Another challenge was the reluctance to change the organization. Users felt overwhelmed with the new ways of working and, for a while, have kept using both the old applications and the newly installed solution.
- **Penetration of the ERP system:** The level of process owners' involvement in the ES implementation was proportional to the quality level of results. The process owners were committed to reuse their old processes, which led to significant customization efforts. The penetration of the ERP was assessed according to two indicators: the number of people who used it and the number of processes covered. The latter gives a clearer picture of the use, than the first because many employees can be in functions in which they have nothing to do with the ES itself, for example, call centre representatives or field technicians in cell site building. Within the company, 30-40% of the business processes are covered with SAP and they are still extending.
- **Vision:** The organization wanted in a longer-term to achieve a competitive advantage by implementing the SAP solution. ERP was a pricy endeavor; once it was brought in, the users got to live with it. Therefore the focus is now on maximizing the value of ERP and extend it to other non-core activities and back office.
- **Drivers & Lessons:** The company's drivers were: (i) integration of sites and locations, (ii) reducing transaction costs, and (iii) replacement of legacy applications. There was a steep learning curve through the process. Some requirements engineering activities, like requirements prioritization and negotiation went wrong in the first place, but solutions were found later in the RE process. More about the lessons learned in the process can be found in [4].

7.3. Evaluation of the results

This section discusses the links between the two models as observed in our case study: we first start with 'Strategic use of IT' (ES UMM) and 'Vision' (ES UMM). The 'Business linkage' (DoC ACMM) in the architecture process was high: the business was responsible for the architecture deliverables as well as for the choice of the ES. In addition, the choice for SAP was an architecturally-sound and well-thought-out decision. This indicated a high level on both AMM and ES UMM levels.

Second, the 'Organizational sophistication' (ES UMM) was rated low which was due to insufficient stakeholders participation. This was also a weak point in the architecture process and reflected in a low level of 'Architecture communication' (DoC ACMM).

Third, the organization had process and data owners who were involved in both the architecture process and the ES implementation process. The organization was mature in terms of 'Operating Units Participation' and 'Business Linkages'. However, when assessing 'Penetration of the ERP' (ES UMM), it was found that the level of involvement of these process owners varied widely: some of them who were committed to the architecture process were not enough committed to the ES implementation process. These process owners did effectively negotiate their business requirements and signed-off them without suggestions for improvements at the end of the ERP RE process, but they did not return to the later implementation stages after the initial spirit has worn off. They did not show any enthusiasm for repeating the RE process in future projects and suggested other business representatives take over the remaining project stages [5]. This led us to the conclusion (i) that many factors – beyond maturity of the enterprise architecture in a company, can affect the level of ERP penetration in an organization, and (ii) a mature architecture team alone is not enough to positively impact business users' participation and involvement in implementing an ES.

Fourth, although business drivers were defined for each project, the organization found that some of them were in conflict; indeed, conflicting business drivers led to unnecessary complex SAP customization and needless installation of multiple system versions [4,5]. In the early projects, the organization failed to see the ERP initiative as a learning process as well.

To sum up, high architecture maturity does not necessarily imply coordination in determining ERP priorities and drivers; neither, it can turn an ERP initiative into a systematic learning process.

While the architecture maturity in the beginning of the project was very high, the organization could not set up a smooth implementation process for the first six ERP projects. So, at the start, the ES usage maturity was low (stage 1) although the company was clear on the strategic use of IT and treated the ES implementation projects as business initiatives and not IT projects.

8. Conclusions

In this paper, we examined the linkages between the assessment constructs of two types of maturity models, namely ACMM and ES UMM. We used one company's experiences in ERP implementations as a case study to get a deeper understanding of how these constructs refer to each other. We found that all ACMM and ES UMM constructs are interrelated. The ES UMM constructs are about the strategy and vision of the company, the penetration and use of the ES. However, although most of the ES UMM constructs correlate to the architecture model's constructs, the interpretation of them in both maturity models can be different. Furthermore, we found that a well-established architecture function in a company would not directly imply that there is support for an ES-implementation. This leads to the conclusion that a high architecture maturity will not automatically lead to a high ES usage maturity.

In our case study, we do not give exact measurements of the models. We used qualitative assessments because measurements are often not as precise as is thought [3]. In complex cases like ES implementation, indeed using one only model for assessment is not enough; information from more sources should be collected.

Finally, architecture maturity is a term used in many models, often related to business/IT alignment [15]. These models are much more elaborated than the ES UMM. Therefore, more research has to be done in the area of ES usage maturity to bring the ES UMM to the level of sophistication that other models offer.

Our future research towards refining ES UMM concepts involves case studies at companies' sites in which we plan to analyze how enterprise architecture is used in managing strategic change [25]. We also plan to investigate how calibration, capability assessments, and maturity advancement [11] are used to achieve business/IT alignment.

9. References

[1] Amram M., N. Kulatilaka, Real Options: Managing Strategic Investment in an Uncertain World, Harvard Business School Press, Boston, 2000.

[2] Chang, S. ERP Life Cycle Implementation, Management and Support: Implications for Practice and Research, Proc. of the 37th Hawaii Int'l Conf. on System Sciences, (CD ROM), 2004, pp. 80227c.

[3] O'Connell, E., Saiedian, H. Can You Trust Software Capability Evaluations? IEEE Computer, Feb 2000, pp. 28-35.

[4] Daneva, M. ERP Requirements Engineering Practice: Lessons Learned. IEEE Software, March/April 2004.

[5] Daneva, M. (2004) Patterns of Success and Failure in ERP Requirements Engineering: an Empirical Study, Proc. of Int'l Workshop on Software Measurement, Berlin, Shaker Verlag, pp. 527-546.

[6] Davenport, T. Putting the enterprise into the enterprise system. Harvard Business Review, 76(4), 1998, pp. 121-131.

[7] Department of Commerce, USA Government Introduction – IT Architecture Capability Maturity Model, 2003, retrieved 04-11-2004 from https://secure.cio.noaa.gov/hpcc/docita/files/acmm_rev1_1_05202003.pdf

[8] Dube, L, G. Pare, Rigor in Information Systems Positivist Research: Current Practices, Trends, and Recommendations, MIS Quarterly, 27(4), 2003, pp. 597-635.

[9] Gormly, L. EAI Overview. IT Toolbox EAI. retrieved 06-Sep-2004 from http://eai.ittoolbox.com/browse.asp?c=EAIPeerPublishing&r=%2Fpub%2Feai%5Foverview%2Ehtm

[10] Van Grembergen, W., Saull, R. Aligning Business and Information Technology through the Balanced Scorecard at a Major Canadian Financial Group: its Status Measured with an IT BSC Maturity Model. Proc.of the 34th Hawaii Int'l Conf. on System Sciences, 2001.

[11] Harigopal, U., Satyadas, A., Cognizant Enterprise Maturity Model (CEMM). IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews, 31(4), Nov. 2001.

[12] Holland, C., Light, B. A Stage Maturity Model for Enterprise Resource Planning Systems Use. The DATABASE for Advances in Information Systems, 32 (2), 2001, pp. 34–45.

[13] Hyde, K. D. Wilson, Intangible Benefits of AMM-based software Process improvements, Software Process Improvements and Practice, 9, 2004, pp. 217-228.

[14] Lee, J., Siau, K., Hong, S., Enterprise Integration with ERP and EAI. Communications of the ACM, 46(2), 2002.

[15] Luftman, J.N., Competing in the Information Age: Align in the Sand, 2nd Ed., Oxford University Press, 2003.

[16] Markus, M., Axline, S., Petrie, D., Tanis, C. Learning from Adopters' Experiences with ERP: Problems Encountered and Success Achieved. Journal of Information Technology, 15, 2000), pp. 245-265.

[17] Markus, M. Paradigm Shifts – E-Business and Business / Systems Integration, Communications of the AIS, 4(10) (2000).

[18] Martin, J., Strategic Data-planning Methodologies. Prentice Hall, 1982.

[19] Muntslag, D.R., Limburg, D.O., Katsma, C. The Art of Implementation. Springer-Verlag, Berlin, 2004.

[20] Paul, M. C., Weber, B., Chrissis, M.B., Curtis, W., The Capability Maturity Model: Guidelines for Improving the Software Process. Addison-Wesley Professional, 1995.

[21] Renken, J. (2004). Developing an IS/ICT Management Capability Maturity Framework, Proc. of SAICSIT 2004, pp. 53 – 62.

[22] Sarker, S., Lee, A., Using a Case Study to Test the Role of Three Key Social Enablers in ERP Implementation. Proceedings of the 21st Int'l Conf.

on Information Systems, 2000.

[23] Schekkerman, J. Enterprise Architecture Score Card, Institute for Enterprise Architecture Developments, Amersfoort, The Netherlands, 2004.

[24] SearchWebServices definitions, a TechTarget site for Web Services professionals. Retrieved 26-01-2005 from http://searchwebservices.techtarget.com/sDefinition/0,,sid26_gci213523,00.html

[25] Veasey, P. Use of enterprise architectures in managing strategic change. Business Process Management Journal, 7(5), 2001, pp. 420-436.

[26] Yin, R. K. Case Study Research, Design and Methods, 3rd ed. Newbury Park, Sage Publications, 2002.

[27] Zhang, L., Lee, M., Zhang, Z., Banerjee, P. Critical Success Factors of Enterprise Resource Planning Systems Implementation Success in China. Proc. of the 36th Hawaii Int'l Conf. on System Sciences. 2003.

Business and IT Alignment with SEAM

Alain Wegmann, Gil Regev
Ecole Polytechnique Fédérale de Lausanne (EPFL),
School of Computer and Communication Sciences
CH-1015 Lausanne, Switzerland
{alain.wegmann, gil.regev}@epfl.ch

Bertrand Loison
Office Fédéral de la Statistique
CH-2010 Neuchâtel, Switzerland
bertrand.loison@bfs.admin.ch

Abstract

Aligning business with IT requires understanding goals, strategies and needs. To be able to express them, an enterprise model can be developed. We present some of the traditional techniques used for the development of an enterprise model (value system, BPMN, UML) and compare them with a systemic method (SEAM). This comparison is done by presenting a real project done at the Swiss Federal Statistical Office. We also show that the concepts of goals, strategies and needs correspond to interpretations of the stakeholders of the enterprise model.

1 Introduction

Business / IT alignment is important for enterprises. It is believed that if this alignment can be maintained over time, it will contribute to the long term success of the enterprise.

Alignment or fit can be seen as the correspondence between a set of components [11]. This set of components can be defined in multiple ways. For example, Luftman & McLean define business-IT alignment as the correspondence between the strategies, goals, and needs of the business and the requirements of the IT system [7].

Knoll and Jarvenpaa [6] identify multiple dimensions of alignment, one of them being "external vs. internal" [11]. The strategies, goals and needs of the enterprise are most often related to external alignment. They seek to align the enterprise with its environment. Internal alignment addresses the way the enterprise implements its goals and strategies.

Enterprises maintain their alignment (external and internal) with respect to the constraints imposed on them by the environment and constraints, they impose on the environment. These constraints are often contradictory to one another, which forces enterprises to seek compromises between them [3]. This is the essence of strategic management [8].

Methods for business – IT alignment frequently analyze the alignment in terms of relations between a system, typically the IT system, and its immediate environment (e.g. group of users). This is especially true for the requirement engineering methods based on goals and scenarios [18].

However, considering the immediate environment of the envisioned IT system is not enough. The IT system and its users have themselves an environment (e.g. the rest of the enterprise). The enterprise has also an environment (e.g. the market in which it exists). For a complete alignment, all these environments must be considered.

Traditionally, each level (e.g. market, enterprise, IT system) is analyzed with its specific method. So, reasoning about alignment requires using different methods. In this paper we present the use of SEAM ("Systemic Enterprise Architecture Method"). SEAM is designed to reason in a systematic and systemic manner about all these levels [20]. The goal is to be able to design SEAMless alignment between these levels.

This paper is based on an example taken from a concrete project of the Swiss Federal Statistical Office (OFS[1]). The OFS is a governmental organization providing statistics about Switzerland. The OFS collects data from multiple sources such as individuals, states and enterprises, computes statistics and publishes its findings to the public at large. OFS publishes data and statistics on a large range of subjects. They are valuable instruments in government decisions and many governmental and non governmental organizations rely on them for policy making. The project we describe was triggered by the efforts to optimize the use of the OFS IT resources. In this paper, we describe the SEAM enterprise model used by the OFS CIO in his decision process.

In Section 2, we define the key concepts of SEAM and, in particular, the concept of alignment. In Section 3, we compare SEAM to traditional modeling techniques in the context of the OFS project; we conclude the section with a discussion on how a SEAM enterprise model supports reasoning about business goals, needs and strategies. In Section 4, we present some related work. In Section 5, we conclude with a discussion of the impact of using SEAM and an outlook on future possible research.

2 Alignment and the SEAM Paradigm

SEAM defines a systemic (or holistic) paradigm for analyzing enterprises and their IT systems. It defines a method, modeling principles, and theories useful to model and reason about enterprises, their IT systems and the changes they go through [20]. In this Section, we define the key concepts of SEAM. We then define what we mean by alignment.

Enterprise model: In SEAM, the perceived enterprise reality is represented in a hierarchical enterprise model that typically describes the markets of an enterprise, the enterprise itself and its IT systems.

As-is and to-be: An enterprise model represents two situations: the "as-is" and the "to-be". These two situations are useful to describe a project. The "as-is" is the situation at the beginning of the project. The "to-be" is the situation at the end of the project. Moving from a situation as-is to a situation to-be in which the business - IT alignment has been analyzed, designed and verified contributes to increasing the business-IT alignment of the enterprise.

Organizational level: Each organizational level represents a partial enterprise reality. Each organizational level contains systems. A SEAM enterprise model typically has three or more organizational levels. In the OFS example, we have three levels: business organizational level representing the OFS and its partners (i.e. data providers, customers etc); the operation org level representing some of the OFS organizational units (e.g. sections and divisions); the IT organizational level representing the OFS employee and the IT systems. Additional levels could be added to describe either the market or the IT architecture.

System: Systems are defined as sets of collaborating entities. A system can be an IT system, a department, an enterprise, a network of enterprises, or even a market. Systems can be modeled as wholes (useful to represent roles of systems) or as composite (useful to represent the system's components and their collaborations). In our example, we consider the OFS as a whole (to analyze/design its roles relative to its partners) and as a composite (to analyze/design the collaborations between the OFS organizational units – such as sections, divisions).

Role: Systems represented as wholes have roles[2]. A role is defined as a behavior that changes the properties of the system fulfilling the role and of its environment. The changes are described in terms of pre and post-conditions. In our example, the OFS (as a whole) has the role "product generation" and the role's post-condition is the set of new products generated by the role.

Collaboration: Collaborations are defined in terms of simultaneous changes of the participants to the collaboration. Collaborations can also be understood as the "joint-roles" of the participants to the collaboration. Collaborations, as roles, are behaviors

[1] In this paper we designate the office with the French acronym OFS, for "Office Fédéral de la Statistique" (http://www.bfs.admin.ch/bfs/)

[2] The term role can be considered as a synonym for "service". SEAM can be used to analyze and design services provided by systems.

that change the properties of the systems that participate to the collaboration. The difference is that, in a role, only one system changes. In the collaboration, all participating systems do change. Collaborations are useful to describe the results of an action without detailing who does what and how things are done. In the OFS example, the OFS (as a composite made of sections and divisions) has the collaboration "product generation" that express the fact that all the participants need to achieve, together, a product generation. This collaboration is then mapped in the role of each participant to the collaboration. For example, the sections need to collect data.

Functional level: Both the collaborations and the roles can be represented at different levels of details. We call these levels "functional levels". In our OFS example, the interaction between the OFS sections and the divisions of interest will be analyzed at two functional levels. The first functional level describes the collaboration "product generation". The second functional level refines this collaboration into the specific roles of the participants that are necessary to create the product (e.g. "data collection", "transformation", etc…).

In SEAM, we define the alignment as:

System alignment between organizational levels: Two representations of a system in two (adjacent) organizational levels are aligned if it is possible to identify the behavior (i.e. role) described in the higher organizational level in the behavior (i.e. collaboration) described in the lower organizational level.

System alignment between functional levels (in the same organizational level)*:* Two representations of a system at two functional levels are aligned when it is possible to identify the behavior (i.e. role or collaboration) described in the higher functional levels in the behavior (i.e. role or collaboration) described in the lower functional level.

Business and IT alignment: To have a business - IT alignment requires having system alignment between organizational levels (from business down to IT) and system alignment between functions levels (within the same organizational levels). Section 3 illustrates this concretely. A more detailed discussion on the techniques for comparing behaviors (collaborations and roles) is available in [21].

3 Enterprise Models and Business / IT Alignment

In this Section, we first present the business and IT needs of the OFS (Section 3.1).

Next, we compare how an OFS enterprise model can be constructed using traditional modeling techniques and using SEAM. We present the relevant diagrams that represent the business (Section 3.2), the operation (Section 3.3) and the IT (Section 3.4) of the OFS. These three levels are traditionally analyzed in enterprise architecture methods. For each level, we present an "as-is" and a "to-be" situation. For each one (business as-is/to-be, operation as-is/to-be, and IT as-is/to-be), we present two modeling notations: a "traditional" one (that changes from level to level) and SEAM (which is the same from level to level). In SEAM, the differences between the levels lie in the heuristics used to reason about the content of the diagrams and not in the notation.

We conclude (Section 3.5) by a discussion on how an enterprise model developed with SEAM can be used to reason about business / IT alignment as defined by Luftman and McLean [7].

3.1 The Needs of the OFS

The OFS is part of the Federal Department of Home Affairs. The OFS issues statistics in different domains (e.g. agriculture, industry, education, etc). It manages more than 125 statistical products that are available in multiple forms (paper, online, off-line). The OFS is composed of seven divisions totaling more than thirty sections. Approx. 25 of them are responsible of producing statistics. Each of these sections is responsible for a domain of expertise, such as agriculture, education, etc. In this paper, we analyze, in a generic manner, the role of these sections. We ignore the role of the divisions at the exception of one of them: the division "infrastructure". This division has initially two roles. Firstly, it manages the data registries (e.g. list of all commercial enterprises and of all people in Switzerland). Secondly, it operates a data warehouse that holds the statistical data ready for publishing. The section "publishing" use this warehouse to deliver the statistics to the OFS customers. In this example, we will illustrate how a third role is identified for the division "infrastructure": the management of the geographical meta-data (e.g. definition of cities and states boundaries).

The partners of the OFS are the Data Providers and an IT Service Provider (another office of the Federal Department of Home Affairs).

To make its products, the OFS uses both commercially available statistical tools and proprietary tools developed within the OFS. It so happens that for historical reasons the different sections use different tools. The latest trend for commercial statistical tool makers is to provide suites. They develop a price scheme that encourages customers to purchase full suites (very expensive single modules, advantageous price for overall suite). As a consequence, the OFS is forced to purchase complete suites multiple times, which is not a financially acceptable solution. The custom OFS tools are also expensive, as they require maintenance which has to be done by each section.

To control these costs, the OFS has launched a major project called the "90 degree rotation" project. It is a major undertaking as it involves the whole OFS organization (several hundred people). One of the goals is to standardize the commercial tools: i.e. to reduce the number of commercial tools used within the OFS. Another goal is to standardize the custom tools: i.e. to maximize the reuse of the custom tools between sections. An extra benefit expected is the simplification of the data exchanges between sections.

In parallel, the OFS products and services need to evolve. We can illustrate this with two examples. First of all, customers require that more and more statistical data be represented on maps (e.g. statistical map with number of students per city). The OFS needs to improve the integration between geographically referenced data and regular statistical data. This requires a close partnership with the Swiss Federal Office of Topography (SwissTopo) [15] which defines the geographical meta-data for the Swiss government. In addition, (and last for this article), the OFS customers expect to get their data as OLAP cubes. An OLAP (On-Line Analytical Processing) cube is a form of data structure that enables interactive multi-dimensional analysis. This new need is the consequence of the new capabilities provided by the commercial statistical suites used by both the OFS and its customers. This illustrates that a change in IT capabilities can drive customer needs. It represents an additional challenge for the OFS.

In summary, it appears that the strategy of the IT tool vendors and the business strategy of the OFS influence each other. It also appears that, even if the standardization of the statistical tools is the largest project, this project is an opportunity for multiple smaller projects to be launched. This justifies the overall effort of explicitly analyzing and designing the business – IT alignment. The SEAM diagrams in Section 3.2 to 3.4 represent the result of this effort. When reading the paper, it appears as if the project follows a top-down approach. In practice these diagrams were developed through multiple iterations. In some cases, the business requirement was identified first and the goal was to implement this requirement. In other cases, the implementation was identified first and the goal was to understand the business requirements. As our goal in this paper is only to illustrate how a SEAM enterprise model can be used to support reasoning about business and IT alignment, we present the final OFS model and we do not present how it was developed. The benefits of using an approach such as SEAM are discussed in the conclusion.

3.2 Business: Modeling Business Relations

Modeling the environment of an enterprise requires the modeling of the enterprise's relations with other enterprises and individuals. Aspects such as relationships with customers, suppliers, regulators etc. are modeled and analyzed. We therefore present the way the OFS business relations would be modeled with a traditional technique, i.e. Porter diagrams, followed by the same relations modeled with SEAM.

3.2.1 Traditional Business Relation Modeling

Probably the most popular business modeling tools for understanding the situation of an organization in its environment is the value system [10]. We can use this tool to represent the OFS and its current environment (as-is), and the desired OFS in its desired environment (to-be).

Figure 1 represents the OFS value system, as-is. Each "arrowed rectangle" (shape defined by Porter in [10]) represents an enterprise, e.g. the OFS, the OFS customer etc. The "product" flow goes from left to right. The diagram hints that the OFS aggregates and analyzes data coming from its data providers and delivers it to its customers.

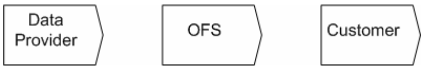


Figure 1: Porter's Value System as-is of the OFS

Figure 2 represents the OFS value system, to-be.

In Figure 2, SwissTopo, provider of standardized geographic meta-data, is added.

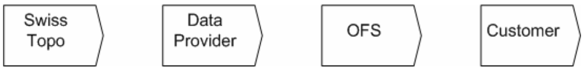


Figure 2: Porter's Value System to-be of the OFS

The advantage of the Porter notation is its simplicity. However, this simplicity creates some challenges. First, the sequence of the enterprises is not always obvious (e.g. unclear whether the meta-data provider needs to appear before or after the data provider). This is a consequence of the linear nature of the diagram. Second, the value system diagram doesn't convey why the cooperation with the partner enterprises is necessary (e.g. why are the meta-data necessary). Third, the diagram does not show the other needs of the enterprises, in particular, the needs not directly related to the structure of the value system (e.g. what is exchanged between companies or the need to develop new products).

3.2.2 The SEAM Business Organizational Level

The SEAM Business Organizational Level is a richer representation of the Porter's Value System.

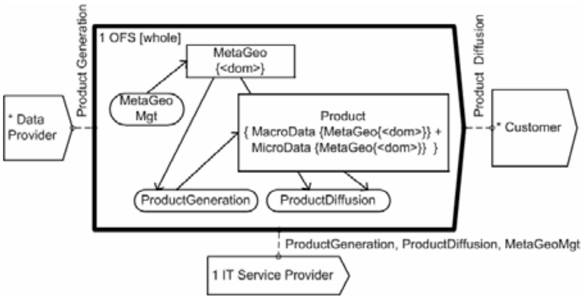


Figure 3: SEAM Business Org Level, as-is

Figure 3 represents the business org level as-is. Figure 3 is the SEAM equivalent of Figure 1: the value system as-is.

Figure 3 represents the OFS as the central system and its partners are around it. On the associations between the partners and the OFS, it is possible to see in which role the partners participate. For example, Data Provider participates to ProductGeneration, Customer to ProductDiffusion and IT Service Provider to all.

Within the OFS, we represent the main roles: ProductGeneration which creates the Product and ProductDiffusion which distributes the Product to the Customer. Each role is described in terms of the system properties involved in the role. For example, Product Generation creates Product and uses MetaGeo data. ProductDiffusion uses Product and MetaGeo.

In SEAM it is possible to describe the Product characteristics. For example, the Product contains MacroData (technical term for the statistics) and MicroData (processed raw data). Both depend of the MetaGeo (geographical meta-data). These meta-data vary within the OFS as indicated by the parameter <dom>. <dom> represents a domain of statistics. This reflects an internal OFS issue that will be discussed in Section 3.3. This variation of meta-data is actually a business issue that has to be addressed by the OFS project.

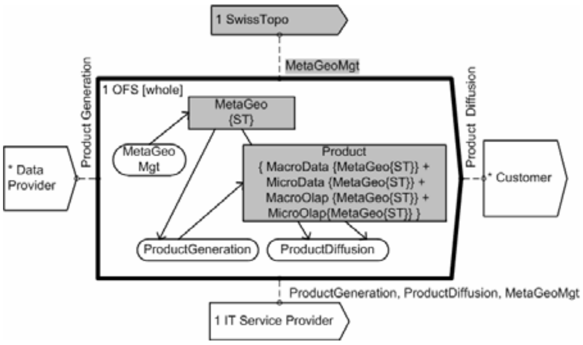


Figure 4: SEAM Business Org Level, to-be

Figure 4 represents the business org level to-be. Figure 4 is the SEAM equivalent of Figure 2: the value system to-be. Within the SEAM enterprise model, it is related by an as-is / to-be relationship to Figure 3.

Figure 4 shows the OFS goals at the business level. The graphical elements in gray put an emphasis on what is important. We can see a new partner, SwissTopo (ST). It is involved in the management of the geographical meta-data. Thanks to this partner, the geographical meta-data can be standardized. This is illustrated by the change of state of MetaGeo from {<dom>} in Figure 3 to {ST} in Figure 4. Finally, two new products have also appeared (MicroOLAP and MacroOLAP).

The SEAM diagrams provide more information than the Value System diagrams. In particular, they make explicit the role of the enterprise and when are its partners involved. The drawback of the SEAM notation is its relative complexity compared to the Porter's notation (Figure 1 and 2).

3.3 Operation: Modeling Business Processes

In this Section we show an operational model of the OFS. It describes the OFS business processes.

3.3.1 Traditional Operations Modeling

We analyze the OFS product generation business process: i.e. the activities needed to develop a new statistical product. The notation is the Business Process Modeling Notation (BPMN) [2]. Note that other notations (such as UML [17], IDEF [5], UEML [16], etc) could be used to represent the business process.

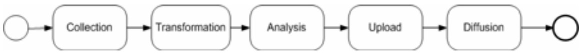


Figure 5: BPMN Business Process of OFS (as-is)

Figure 5 represents the operations as-is of the OFS. The diagram is implicitly aligned to the as-is value system shown in Figure 1. The alignment can be guessed as Collection (Figure 5) is performed because the OFS has DataProvider as a predecessor in the value system (Figure 1).

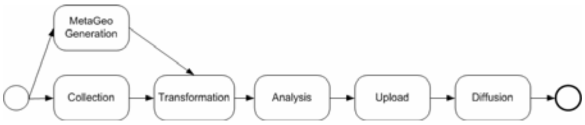


Figure 6: BPMN Business Process of OFS (to-be)

Figure 6 shows the operations to-be of the OFS. The diagram is implicitly aligned to the to-be value system shown in Figure 2. In the new business process, the management of the geographical meta-data is made explicit (although it is not visible that the generation of the meta-data is done asynchronously to the generation of the statistics).

3.3.2 The SEAM Operation Organizational Level

The SEAM operation level also describes the OFS business processes. We represent two functional levels. The first functional level is useful to make explicit the alignment between the business org level (Section 3.2.2) and the operation org level (current section). The second functional level is useful to make explicit the alignment between the operation org level (current section) and the IT org level (Section 3.4.2). In both cases, an as-is and a to-be are developed.

All diagrams in this Section represent the OFS system as a composite. The OFS sections and the OFS division infrastructure are visible together with their roles and the collaborations between them.

First functional level:

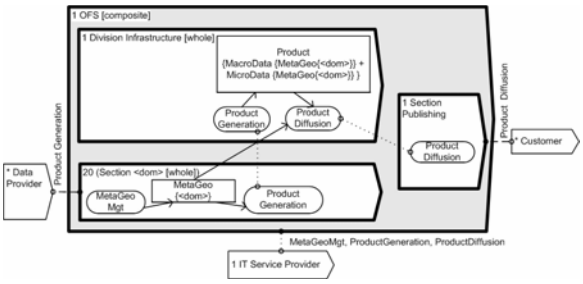


Figure 7: SEAM Operation Org Level; first functional level, as-is

Figure 7 shows the as-is of the first functional level of the operation org level. It is not equivalent with Figure 5 as the process is not shown at the same level of details. Within the SEAM enterprise model, it is *organizationally* aligned with Figure 3 which shows the responsibilities of the OFS.

This diagram makes explicit which OFS organizational units fulfill the OFS responsibilities. For example, the role ProductGeneration of the OFS in Figure 3 corresponds to the collaboration ProductGeneration happening between Section <dom> and Division Infrastructure in Figure 7. We also make explicit who is in charge of storing information.

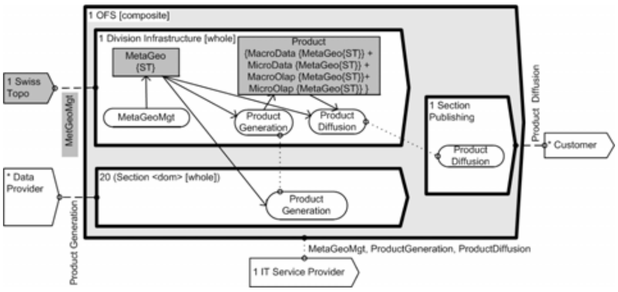


Figure 8: SEAM Operation Org Level, first functional level; to-be

Figure 8 shows the to-be of the first functional level of the operation org level. It is not equivalent to Figure 6 (not the same level of details). Within the SEAM, enterprise model, it is *organizationally* aligned with Figure 4. In addition, it is related by an as-is / to-be relationship to Figure 7.

The comparison between Figure 7 and Figure 8 highlights the impact of the described project. We can see in the as-is diagram that the geographical meta-data is managed by each of the domain-related sections. As the goal of the OFS is to get a better standardization of these geographical meta-data, the OFS needs to transfer the responsibility to manage these meta-data from each section to one entity that will manage it centrally, in collaboration with SwissTopo. This is visible in Figure 8: the geographical meta-data are managed by the Division Infrastructure. Figure 8 also shows the appearance of the “OLAP” products at the operation level (as it did appear in the business org level to-be).

Second functional level:

In the second functional level, the specific sub-roles that need to be executed by the sections and by the Division Infrastructure are identified. This more detailed description of the business process is useful to establish the alignment between operation and IT. As more details are required to describe the situation, we focus on the “ProductGeneration” to keep the diagrams simples.

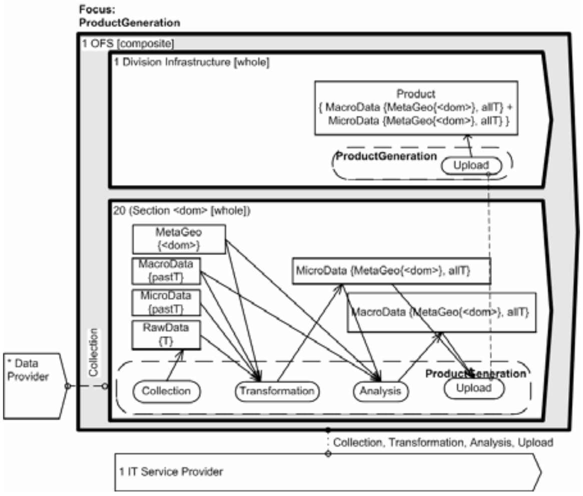

Focus:
ProductGeneration
1 OFS [composite]
1 Division Infrastructure [whole]
Product
{ MacroData {MetaGeo{<dom>}, allT} +
MicroData {MetaGeo{<dom>}, allT} }
ProductGeneration
Upload
20 (Section <dom> [whole])
MetaGeo
{<dom>}
MacroData
{pastT}
MicroData
{pastT}
RawData
{T}
MicroData {MetaGeo{<dom>}, allT}
MacroData {MetaGeo{<dom>}, allT}
* Data
Provider
Collection
Collection
Transformation
Analysis
ProductGeneration
Upload
Collection, Transformation, Analysis, Upload
1 IT Service Provider

Figure 9: Operation Org Level; second functional level; as-is

Figure 9 shows the as-is of the second functional level of the operation org level. It is equivalent to Figure 5. Within the SEAM enterprise model, it is *functionally* aligned with Figure 7.

The diagram in Figure 9 makes the current product generation explicit. The Section <dom> collects the Raw Data at a given time. These Raw Data are then process in Transform (i.e. made anonymous, verified, merged with the MicroData of the previous time periods). The result is a set MicroData for all time periods. The Section <dom> then Analyze these MicroData to produce the MacroData (which are the actual statistics). Both MicroData and MacroData are exported to the Division Infrastructure that stores them till they are used by the Section Publishing upon requests from the Customers.

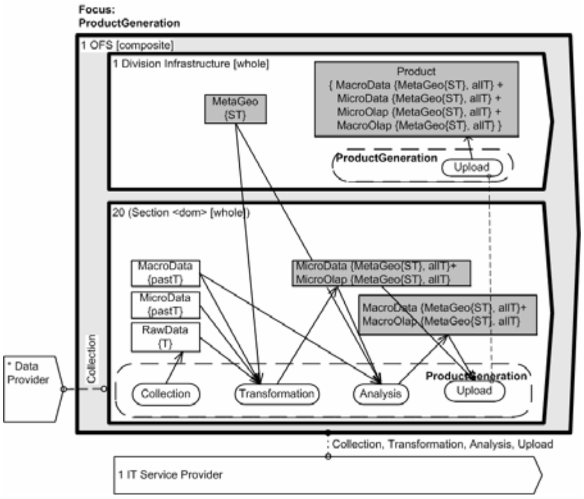

Focus:
ProductGeneration
1 OFS [composite]
1 Division Infrastructure [whole]
MetaGeo
{ST}
Product
{ MacroData {MetaGeo{ST}, allT} +
MicroData {MetaGeo{ST}, allT} +
MicroOlap {MetaGeo{ST}, allT} +
MacroOlap {MetaGeo{ST}, allT} }
ProductGeneration
Upload
20 (Section <dom> [whole])
MacroData
{pastT}
MicroData
{pastT}
RawData
{T}
MicroData {MetaGeo{ST}, allT}+
MicroOlap {MetaGeo{ST}, allT}
MacroData {MetaGeo{ST}, allT}+
MacroOlap {MetaGeo{ST}, allT}
* Data
Provider
Collection
Collection
Transformation
Analysis
ProductGeneration
Upload
Collection, Transformation, Analysis, Upload
1 IT Service Provider

Figure 10: SEAM Operation Org Level; second functional level; to-be

Figure 10 shows the to-be of the second functional level of the operation org level. It is equivalent to Figure 6. Within the SEAM enterprise model, it is *functionally* aligned with Figure 8. It is related by an as-is / to-be relationship to Figure 9.

By comparing this diagram with Figure 9, it is possible to see the new products generated and the change of responsibilities relative to the geographic meta-data.

3.4 IT: Modeling IT Systems’ Roles

In this Section we briefly describe how the IT system can be modeled. A more detailed example on how an IT infrastructure can be modeled with SEAM can be found in [20].

3.4.1 Traditional IT Functional Modeling

UML is the industry-wide standard for modeling IT systems. UML can be used to represent software systems in their environment as well as the implementation of these systems. At the level of description relevant for the OFS problem, we would represent the IT system with use case diagrams.

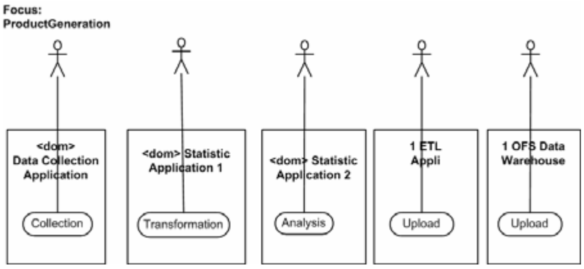


Figure 11: UML use case diagram (as-is)

Figure 11 represents the as-is situation. It is aligned with Figure 5.

Each section uses a specific application, potentially different for each step in the business process. This means that the number of IT applications is at least equal to the number of "domain" multiplied by the number of steps (approx. 75 = 25 "domain" * 3 steps).

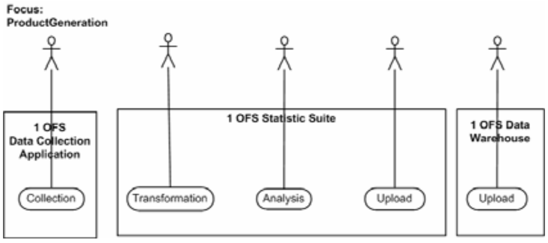


Figure 12: UML use case diagram (to-be)

Figure 12 represents the to-be situation. It is aligned with Figure 6.

It is possible to see that one statistic suite exists for all OFS (which means all sections use the same application as opposed to one per section) and that multiple steps in the statistical analysis are made within the same tool (part of the suites that the statistical tool vendors provide). So the number of applications is drastically reduced.

3.4.2 The SEAM IT Organizational Level

The SEAM IT organizational level describes the roles of the IT systems as well as in which organization the IT systems are managed. This makes explicit the outsourcing strategy of the OFS.

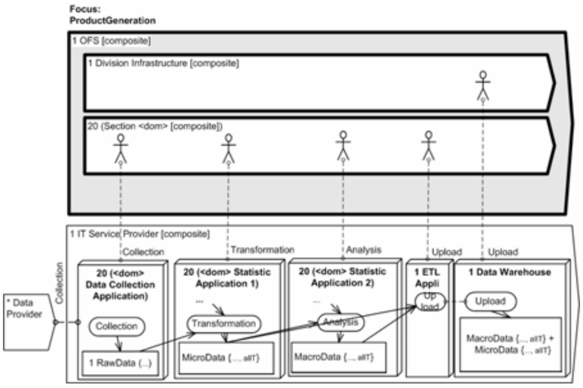


Figure 13: SEAM IT Org Level; as-is

Figure 13 represents the IT org level as-is. It is the SEAM equivalent of Figure 11. Within the SEAM enterprise model, it is *organizationally* aligned with 9. Note that the IT systems are outsourced to the IT Service Provider.

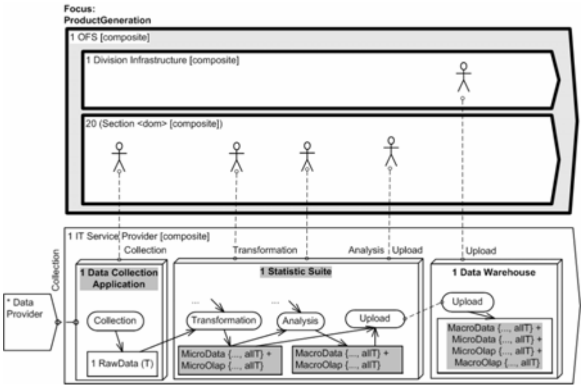


Figure 14: SEAM IT Org Level; to-be

Figure 14 represents the IT org level to-be. It is the SEAM equivalent of Figure 12. Within the SEAM enterprise model, it is *organizationally* aligned with 10. It is related by an as-is / to-be relationship to Figure 13. As for Figure 12, it is possible to see that the number of IT applications is reduced when moving from the as-is to the to-be. The diagram has also the additional benefit to highlight the need to analyze the responsibilities of the employee of the division infrastructure and the ones of the section.

In Summary, in sections 3.2.2, 3.3.2 and 3.4.2 we have illustrated how an enterprise model can be systematically developed. As discussed in the next Section, this model can be used to formalize the goals, strategies and needs of the enterprise.

3.5 Identifying Needs, Goals and Strategies

Luftman and McLean [7] define business/IT alignment as "applying IT in an appropriate and timely way, in harmony with business strategies, goals, and needs." Even if what appear in the SEAM diagrams do not refer explicitly to the terms "goals", "needs" and "strategies" proposed by Luftman and Mclean, SEAM is closely related to these terms. In the following paragraphs we make this relationship explicit.

First, let's analyze the concept of goals. SEAM presents a hierarchical model that describes business, operations and IT. This set of organizational levels constitutes the enterprise model. This enterprise model is used by different specialists to reason about the project. Each specialist will see a different part of the SEAM enterprise model as their goal. For example, Luftman and McLean refer to business goals. Typically, in the OFS, we could consider that Figure 4 (business to-be) represents the business goal of the project as probably defined by the OFS CEO. Figure 8 (Operation, 1st functional level, to-be) represents the goals for the managers of the OFS sections (while being the means for reaching the goals of the CEO). Figure 10 (Operation, 2nd functional level, to-be), can be considered as the means to achieve the goal defined in Figure 8. Figure 10 can itself be considered as the business goal for the IT managers. Hence, the concept of goal is useful to describe what is expected to happen. The goals are contextual and differ for each specialist. In SEAM, the construction of the "to-be" diagrams defines the goals of the project. Each specialist can recognize herself in the SEAM to-be diagrams.

Second, we analyze the concept of strategies. Luftman and McLean do not formally define what a strategy is. In [8], Mintzberg et al define five kinds of strategies: strategy as a plan of actions, strategy as a pattern of realized actions, strategy as position, strategy as perspective, and strategy as a ploy. In SEAM, strategies, just like goals, are not explicitly visible. However, they are captured in the decisions made when a model element as whole is refined as an element as a composite. For example, when the OFS decides to work with SwissTopo to generate geographical maps with statistical data, this is a partnership strategy. Another example is when the OFS as an enterprise is organized into sections and divisions with specific responsibilities; this is an organizational strategy. So, with a SEAM enterprise model it is possible to describe multiple strategies (business, operation, IT) existing in a project.

Last, we need to analyze the needs. The needs are actually not represented in the SEAM diagrams but can be described by the difference between the as-is and to-be diagrams.

In summary, in SEAM the alignment between business and IT corresponds to the traceability between the business org level, the operation org level and the IT org level (done though the two kinds of alignments defined in Section 2). Making the SEAM enterprise model does capture the needs of the enterprise (the difference between the as-is and the to-be), the goals (to evolve toward the to-be) and the strategies (the structure of what is represented). Luftman and McLean speak more in project terms (goals to reach, needs that drive the project, strategies that constrain the solution). SEAM focuses more on describing the enterprise as it is and as it should be.

4 Related Work

As we have stated in the introduction, all RE methods fundamentally seek to align the properties of an envisioned system with the properties of its environment. In the case of IT systems this environment is the enterprise and the enterprise's environment. Most RE methods propose to align the IT system with its immediate environment, i.e. the enterprise. RE methods also lack the integration with strategic management and marketing language and methods complicating the alignment with business goals, strategies and needs.

Goal-Oriented RE (GORE) methods [19], [13], for example, use goals and scenarios to perform this alignment from strategic business objectives to detailed IT requirements [18, 19]. However, most GORE methods consider goals to be self contained within the enterprise. They do not provide sufficient tools for linking these goals with the enterprise's environment. The diagrams and terms used in these methods (goal reduction, and/or diagrams etc.) do not match strategic management and marketing concepts.

SEAM is one of a number of RE methods that take business issues into consideration in order to improve the alignment of business and IT systems. In the following, we briefly describe some of them.

The e^3-value method [4] consists in modeling a set of interrelated enterprises as a network of value exchanging actors. Value flows can be quantified in order to determine whether actors are profitable or not. IT system high-level requirements are defined based on this need for actor profitability and value exchange.

Osterwalder and Pigneur [9] propose an ontology for e-business models in which IT system high-level requirements are explored in terms of the support they can provide to an enterprise's e-business strategy.

Robertson and Robertson [14] propose to use contextual diagrams in order to understand the role of a software based system within an environment constituted by a network of actors.

Alexander [1] explores the requirements for a system by modeling its environment in several layers referred to as the "onion model" Each layer contains a model of the system's stakeholders. Each stakeholder is represented as a whole with their corresponding roles.

The i* method [22] proposes a modeling technique where a network of enterprises are modeled using a strategic relationship diagram. This kind of diagram shows how these enterprises are dependent on each other in the achievement of their goals. Goals can be either (hard) goals for which there are agreed upon criteria for their achievement and soft goals for which these criteria are not well defined. These goals can be refined (maintaining the alignment of lower level goals with higher level goals) until they can be assigned to individual agents, human, machines, IT systems.

The main difference between SEAM and these methods lie in the way SEAM models behavior systematically across organizational levels. The above techniques could be considered as adding additional information to the SEAM models. The SEAM model can be considered as a complementary model that defines the "business-specific terminology" used in the models developed with the above techniques.

A lot of work exists on enterprise modeling based on activity diagram [2], [16], and [17]. SEAM relies also on a kind of activity diagrams. Quite often the SEAM diagrams can be related to regular BPMN or UML diagrams (e.g. activity diagrams). The difference is that, in SEAM, more contextual information is made explicit. This is why they are better suited for multi-disciplinary teams.

5 Conclusions and Future Work

Luftman and McLean claim that business and IT alignment requires taking into consideration needs, goals and strategies. Our goal with this paper was to show that working on such issues can be done when making an enterprise model that represent how business, operation and IT have to evolve. Once such a model is made, each specialist can recognize her needs, goals and strategies in this model. So, developing an enterprise model such as what we illustrate with SEAM can be useful to reason about business and IT alignment.

SEAM is illustrated in this paper on a typical enterprise architecture project. Such project is a large undertaking that includes multiple sub-projects. SEAM has been used successfully on other, smaller, industrial projects (e.g. equipment of a new building, introduction of an MRP system in a manufacturing environment). The observed benefits of making a SEAM enterprise model are:

- Development of a shared understanding (and a glossary) within the project team.
- Better planning of the evolution of the enterprise. In particular: identification of the "unexpected" projects necessary to support the evolution; sizing of the projects; understanding the organizational impacts of the projects.
- Development of better business case to justify the project funding. The SEAM model allows understanding precisely the business impacts of the projects.

The SEAM diagrams are good tools to reason and to support the decision process within the project teams. However, they are in general simplified when used to communicate with people outside of the project.

To be truly practical, SEAM needs to have tool support. A prototype tool does exist. We are currently finalizing the formalization of the notation. This will allow us to provide a tool support for projects such as the one described in this paper.

6 References

[1] Alexander, I.F., A Better Fit – Characterising the Stakeholders, *Proceedings of the 5th BPMDS Workshop*, Riga, Latvia, June 2004.

[2] http://www.bpmn.org

[3] Checkland, P. and Scholes, J., *Soft System Methodology in action*, Wiley, Chichester UK, 1990.

[4] Gordijn, J., Kartseva, V., Schildwacht, J., Wieringa, R. and Akkermans, H., "Developing a Domain-Specific Cross-Organizational RE Method" in *Proceedings 12th IEEE International Requirements Engineering Conference*, Sept. 6-10, Kyoto, Japan, 2004.

[5] http://www.idef.com

[6] Knoll, K. and Jarvenpaa, S., L., Information technology alignment or "fit" in highly turbulent environments: the concept of flexibility, *Proceedings of the 1994 computer personnel research conference on Reinventing IS*, Alexandria, Virginia, United States: 1 – 14, 1994.

[7] Luftman, J. and McLean, E.R., "Key Issues for IT Executives" *MIS Quarterly Executive*, Vol. 3 No. 2, June 2004.

[8] Mintzberg, H., Ahlstrand, B. and Lampel J. *Strategy Safary – The complete guide through the wilds of strategic management*, London: Prentice Hall, 1998.

[9] Osterwalder, A. and Pigneur, Y., "An ontology for e-business models", chapter in Wendy Currie (ed) *Value Creation from E-Business Models*, Butterworth-Heinemann, 2004

[10] Porter, M. E., *Competitive Advantage*: Free Press, 1985.

[11] Regev, G. and Wegmann, A., Remaining Fit: On the Creation and Maintenance of Fit, *Proceedings of the 5th BPMDS Workshop*, Riga, Latvia, June 2004.

[12] Regev, G. and Wegmann, A., "Defining early IT system requirements with regulation principles: the lightswitch approach" in *Proceedings 12th IEEE International Requirements Engineering Conference*, Kyoto, Japan, Sept. 6-10, 2004.

[13] Regev, G. and Wegmann, A., "Where do Goals Come from: the Underlying Principles of Goal-Oriented Requirements Engineering" Accepted *13th IEEE International Requirements Engineering Conference*, Paris, France, August, 2005.

[14] Robertson, S. and Robertson, J., *Mastering the Requirements Process*, Addison-Wesley, Reading MA, 1999.

[15] Swiss confederation, swisstopo, http://www.swisstopo.ch/fr/

[16] http://www.ueml.org

[17] http://www.uml.org

[18] van Lamsweerde, A. "Goal-Oriented Requirements Engineering: A Guided Tour," in *Proceedings RE'01, 5th IEEE International Symposium on Requirements Engineering*, Toronto, August 2001.

[19] van Lamsweerde, A. "Goal-Oriented Requirements Engineering: A Rroundtrip from Research to Practice," in *Proceedings 12th IEEE International Requirements Engineering Conference*, Sept. 6-10, 2004.

[20] Wegmann, A., "On the Systemic Enterprise Architecture Methodology (SEAM)," in *Proceedings of International Conference on Enterprise Information Systems 2003* (ICEIS 2003), Angers, France.

[21] Wegmann, A., Balabko, P., Lê, L.S., Regev, G. and Rychkova, I., "A Method and Tool for Business-IT Alignment in Enterprise Architecture", Proceedings of CAiSE'05 Forum, Porto, Portugal, June 2005.

[22] Yu, E.S.K. "Towards modelling and reasoning support for early-phase requirements engineering," in *Proceedings of the Third IEEE International Symposium on Requirements Engineering*, Annapolis, Maryland, January 1997

Examining Stakeholder Requirements for Software Quality

Maria Haigh
University of Wisconsin,
Milwaukee, WI
mhaigh@uwm.edu

June Verner
National ICT Australia Ltd
Sydney, Australia
june.verner@nicta.com.au

Abstract

'High quality' might seem an obvious requirement for any piece of software, but do the different stakeholder groups involved in its production and use conceptualize this requirement in the same way? Many existing models refine the broad concept of quality into a number of well-defined and measurable attributes related to the software product itself and the development process which produced it. Until now, however, little attempt has been made to empirically examine the requirements for software quality held by different groups involved in the development process. We conducted a survey of more than 300 students and alumni of one of the leading Executive MBA programs in the United States, asking them to rate the importance of each of 13 widely-cited attributes related to software quality. The results showed business role-related differences in some specific areas and agreement in many others. We also consider the implications of these results and their relevance to software requirements analysis.

Keywords: Software quality metrics, perceptions, priorities, software stakeholders, business need, requirements.

1. Introduction

In 1964, U. S. Supreme Court Justice Potter Stewart was faced with the need to define obscenity. Abandoning any attempt to define specific acts, depictions or measurable characteristics he instead noted that "I shall not today attempt further to define the kind of material I understand to be embraced… [b]ut I know it when I see it." This statement would accurately capture the attitude of many people towards software quality. We all think we know what it means, but most people have difficulties in defining it. As a result we can no more be sure that two different groups would view a piece of software as high quality than we could be sure that the citizens of San Francisco and Salt Lake City would uphold the same standards of obscenity. Both are in the eye of the beholder.

To overcome this problem, many models of software quality have been proposed, each of which has tried to separate the broad concept of quality into a number of well-defined and measurable attributes related to the software product, its fidelity to requirements, and the development process which produced it. The best of this research, seeking empirical confirmation, has tied observed attributes to project outcomes [14].

Any software project includes several different sets of "stakeholders," including users and developers, and managers and non-managers. In this research, we conceive of these stakeholder responsibilities as being business roles adopted by particular individuals with respect to specific pieces of software. Someone with the stakeholder role of manager of development for one software project might be a user of another piece of software and a developer of a third. We see attitudes to software quality among these different groups as indicative of their perceptions of the business needs the software will be required to satisfy. In this sense, software quality requirements may be thought of as a specialized subset of business requirements, or at least as desired characteristics that will allow the software to satisfy those requirements.

Our research asks whether these different stakeholder groups value the same attributes when defining their requirements for software quality. By asking a variety of software stakeholders to evaluate the importance of different commonly used attributes of high quality software we aim to determine their implicit personal definitions of software quality. This allows us to explore the relationship between business roles and software requirements. If profound differences are found between holders of different stakeholder roles, this signals a need to take steps to bridge this cultural gulf between participants. Alignment of software quality conceptions between holders of these different business roles will allow organizations to devote resources to agreed upon high-priority attributes with an expectation that all stakeholders groups will value the results.

2. Background

Requirements for software quality can be defined from many points of view, depending on the role the person plays with the software and on the type of system being developed [1], [3], [6], [10]. Existing research shows that we have to view software quality requirements not as an absolute measure, but in terms of trade-offs [7]. The implications for requirements analysis and perceptions of business need are obvious. If quality is refined to a set of effective and comprehensible metrics, then the required and desired levels of each attribute can be specified during the requirements specification phase of any project [5], [9]. Because recent models indicate correlations (both negative and positive) between desirable attributes (such as maintainability and efficiency), devoting resources to maximizing inappropriate attributes might actually damage the effectiveness of the software produced [8]. Quality therefore can be viewed as a set of unavoidable trade-offs, existing beyond the familiar tensions between time, cost, and quality.

A better understanding of software quality requirements for different stakeholder groups will lead to better communication between the parties involved with the system. To understand business need, managers and developers should understand what aspects of software quality are important to them, and to users, so that they can ensure that developers of the system implement the features with the highest priority.

3. Method

We conducted an online survey of 315 software stakeholders. The survey made available using a web interface connected to a database. The URL was distributed via email to the Executive MBA students and alumni at one of the most highly ranked business schools in the United States. Distribution of the survey to this sample facilitated reaching a homogeneous group of people with the same education, yet representing managers, users, and technical personnel from all sectors of the U.S. economy.

Respondents used a wide variety of different software packages. We therefore asked each respondent to select the piece of software most important to them in carrying out their work responsibilities and answer questions with respect to this piece of software. This gives more meaningful results than simply asking the respondent about his or her attitudes to software in general.

Stakeholder role was defined with respect to the specific piece of software chosen for evaluation. We used two axes on which to divide our respondents into four distinct software stakeholder roles. There is an axis of users versus developers: stakeholders who are involved in managing or performing the software development process and those who are not directly involved in these tasks. There is also an axis of managerial versus non-managerial responsibilities with regard to the software.

We are interested in finding out whether members of the four different stakeholder groups largely agree on the priorities assigned to different software quality attributes or whether widespread and systematic divergences exist in the priorities assigned to different software quality attributes by members of the different stakeholder groups. Thus, the null hypothesis of the study can be expressed as follows:

H_0: There is no significant difference in software quality priorities between different software stakeholder groups.

The corresponding alternative hypothesis is thus:

H_1: There is a significant difference in software quality priorities between different software stakeholder groups.

The survey included questions covering stakeholder's job function, their relationship to software product most important for their job function, and a set of questions asking the respondent to rate the importance of each of 13 software quality attributes. Each attribute was rated independently on a scale of 1-7, where 7 meant very important and 1 meant not important.

The software quality attributes and accompanying definitions provided to the survey respondents were as follows.

- ACCURACY: The degree to which the software outputs are sufficiently precise to satisfy their intended use
- TESTABILITY: The effort required to test the software to ensure that it performs its intended functions
- USABILITY: The effort required to learn and operate this software
- SECURITY: The extent to which access to this software by unauthorized persons can be controlled
- EFFICIENCY: The amount of computing resources required by this software to perform its function

- CORRECTNESS: The extent to which this software satisfies its specifications and fulfills your mission objectives
- PORTABILITY: The effort required to transfer this software from one hardware configuration or software system environment to another
- AUGMENTABILITY (SCALABILITY): The extent to which this software can take advantage of additional resources to deal efficiently when increased demands are placed on it
- INTEROPERABILITY: The effort required to couple this software with another
- ROBUSTNESS: The degree to which this software continues to function in the presence if invalid inputs or stressful environmental conditions
- FLEXIBILITY: The effort required to modify this software for uses or environments other than those for which it was specifically designed
- MAINTAINABILITY: The effort required to locate and fix an error in this software, or to change or add capabilities
- REUSABILITY: The extent to which components or modules of this software can be used for other purposes

These attributes were selected from the review of existing literature [8]. The list attributes used is neither complete with respect to every attribute proposed in the literature, nor entirely orthogonal. Some of the attributes overlap in their meaning. Many of the attributes came from one of the most heavily cited software quality models - the Boehm et al. software quality model [2]. Boehm's model implies relationships between software quality attributes: the model is not a list of independent qualities, but an interconnected hierarchy of attributes. Some attributes from more recent quality models were incorporated, and many of the descriptions were updated or simplified to make them more relevant to non-specialists and to reflect technological changes.

4. Results

We present our results in the following order: a summary of the background of the respondents by industry sector, stakeholder, and application area of the software they evaluated. Our review of the results continues with a discussion of the data analysis.

4.1 Demographic and Related Data

The main purpose of the study is to explore the software quality priorities held by different software stakeholder groups. Each respondent identified him- or herself as either a user or developer of the software concerned, and as either a manager (managing its users or developers) or non-manager (personally using or developing the software concerned). Combining these two variables thus divided respondents into four groups, which we refer to here as stakeholder roles: User, Manager of Users, Developer, and Manager of Development. Table 1 shows the distribution of respondents by their stakeholder roles.

Table 1. Respondent distribution by stakeholder role

Stakeholder Group	Frequency	Percent
Developer	46	14.6
Manager of Development	52	16.2
User	155	49.2
Manager of Users	59	18.7
Missing Data	3	0.9
Total	315	100

Thirty one percent of the respondents were responsible for development of the software concerned: 16.2% were managing its development, while a further 14.6% were personally performing development tasks. The remaining 69% of the respondents were not associated with the development of the software evaluated, and are therefore treated here as users. Fifty percent personally used the software they evaluated and 18.7% identified themselves as managers of the users of the software they evaluated. (35% of the respondents fell into one or other of the management roles).

The respondents came from a variety of industries as shown in Table 2.

Table 2. Respondent distribution by industry sector

Industry Sector	Frequency	Percent
IT and Telecomm	92	29.2
Government	16	5.1
Healthcare	32	10.1
Manufacturing	55	17.5
Military	5	1.6
Academic and Research	15	4.8
Service-Non-Computer	100	31.7
Total	315	100.0

Most of the respondents (60%) came from two sectors: (1) IT and Telecommunications, and (2) non-IT services. Overall, however, seven major industry categories were represented.

Table 3 shows the distribution of stakeholder roles by industry. Responses associated with developers and developer managers mainly came from IT and Telecommunication industries: 43% and 44% respectively. The service-non-computer industry was the most represented for respondents not associated with software development: 39% of software users and 32% of user managers were from this industry. While each stakeholder role was found across the full range of industries, there is clearly some covariance between industry and role – some of which may reflect the nature of each industry and some of which may be due to random variation in the sample.

Table 3. Stakeholder roles by industry

Industry (column %)	Dvlp. n=46	Mgr. Dvlp. n=52	User n=155	Mgr. User n=59
IT and Telecomm . n=92	43.4	44.2	21.3	25.4
Govt. n=16	10.9	1.9	3.4	6.8
Healthcare n=32	6.5	7.7	12.3	10.2
Manufact. n=55	13.1	13.5	18.7	22
Military n=5	2.2	3.9	0.7	1.7
Academic and Research n=15	6.5	11.5	3.2	1.7
Service-Non-Computer n=100	17.4	17.3	40	32.2

Respondents evaluated a variety of software packages. These packages were categorized across two axes:

- **software application area**: business administration, manufacturing or production, scientific/research activities, creativity-related software (e.g., games, art/graphics, music, etc.), and other;

- **software type:** off-the-shelf-software; off-the-shelf-software customized for respondent's company use, in-house developed software for sale, in-house developed software for the use within respondent's organization, and “other”, software did not fit into any of the previous categories.

Table 4. Application areas of the evaluated software.

Application Area	Frequency	Percent
Business Administration	147	46.7
Creativity	4	1.3
Manufacturing	28	8.9
Other	100	31.7
Scientific	30	9.5
Missing values	6	1.9

Forty seven percent of the respondents evaluated business administration software, making this by far the most represented category of software in the survey. Thirty two percent of the software evaluated was categorized as "other" – meaning that the respondent did not believe it to fit into any of the pre-defined application area types. Scientific and manufacturing software were other two most popular application areas (9.5% and 8.9% respectively). (Table 4).

Table 5. Software application area chosen for evaluation by stakeholder role

Appl. Area (Column %)	Dvlp. n=46	Mgr. Dvlp. n=52	User n=155	Mgr. User n=59
Business Admin. n=147	37.8	30.6	59.7	37.9
Creativity n=4	0.0	0.0	2.0	1.7
Manufact. n=28	8.9	24.5	2.0	15.5
Other n=100	44.4	24.5	28.6	37.9
Scientific n=30	8.9	20.4	7.8	6.9

Table 5 shows the software application areas evaluated by respondents in different stakeholder groups. Data in this table reflects missing data and rounding errors.

Table 6. Software type chosen for evaluation by stakeholder role

Software Type (Column %)	Dvlp. n=46	Mgr. Dvlp. n=52	User n=155	Mgr. User n=59
Off-the-shelf-software	15.2	5.8	62.6	20.3
Off-the-Shelf-Customized	17.4	25.0	19.4	45.8
In-house developed to sell	39.1	32.7	7.1	8.5
In-house developed for the use within own organization	23.9	28.9	9.0	20.3
Other	4.4	7.7	1.9	5.1
Total	100	100	100	100.0

Table 6 shows the development sources of the software being evaluated by members of each stakeholder group. (Respondents were asked to evaluate the piece of software most important to them in carrying out their primary job functions). This shows that 62% of users primarily used off-the-shelf software for their business responsibilities. Developers and developer managers were involved with in-house software developed for sale, off-the-shelf customized software, and in-house developed software for internal use only. Business stakeholders along the managerial axis commonly used off-the-shelf customized software and in-house software developed for the use within their own organization.

Table 7. Average satisfaction with evaluated software by stakeholder groups

Stakeholdr Role	Satisfaction Avg
Dvlp.	3.78
Mgr. Dvlp.	3.88
User	3.95
Mgr. User	3.91

Respondents were reasonably happy with the software under consideration: 78.2% measured their satisfaction with the software as '4' on a 7-point scale.

The differences in software satisfaction between the stakeholder groups were not statistically significant. It is interesting to notice that both developer groups were less satisfied with software than either of the user groups. Developers and managers of development were thus more critical towards software than other stakeholders: they value software quality more and have higher expectations for the software products than respondents who are not involved with software development process.

In the next section we present the results of our analysis of the stakeholders' quality priorities regarding software used for their jobs.

4.2 Data Analysis Results

The aim of this research is to discover if there are systematic differences in software quality requirements priorities between respondents with different stakeholder roles associated with software. Individuals and, more importantly, stakeholder groups, showed substantial variance in the mean scores they assigned to attribute importance. This made the raw data less useful for evaluating systematic divergences in priorities. Our interest here is in software attribute priorities, which we operationalized as the importance assigned to an attribute by a given respondent relative to the average importance assigned by the same respondent to all attributes. These priority scores are obtained by applying simple linear scaling to the results of each respondent. Trochim [15] suggests this type of scaling: dividing the score assigned to an attribute by the sum of scores assigned to all attributes by the same respondent and then multiplying by the number of attributes (13 in our case). The formula for score scaling is as follows:

$$Adjusted\_Attibute\_Priority_{ij} = Raw\_Score_{ij}*N / \sum(Raw\_Score_i)$$

Where i is the record number (one record for each respondent); j is the column number (one column per each quality attribute); Raw_score is the rating entered by a respondent; N is the number of attributes, 13 in our case. Comparison of the importance of the software quality attributes mean frequency distribution analysis and ANOVA analysis were applied to examine collected data.

Differences in software quality attribute priorities between stakeholder groups revealed the following:

- Users ranked accuracy, correctness, integrity, interoperability, robustness, and usability higher than any other group.
- Developers ranked maintainability and testability higher than other groups.
- User managers ranked augmentability, efficiency, and flexibility higher than other stakeholders.
- For development managers reusability was more important than for other groups.
- Developers and development managers appear to be in general agreement. User managers seem to be closer in their software quality priorities to development managers (and to developers) than they are to users.
- Maintainability was significantly more important for managers and developers than for the user group.
- Testability was more important to the development managers and developers than the other stakeholder roles.

Table 8 shows rankings of all quality attributes within the different stakeholder groups. Software quality attributes in Table 8 are ordered by ranking for all respondents.

Table 8. Software quality attributes ranking by stakeholder role

Stakeholder Role	Dvl pr	Mgr. Dev	Usr	Mgr. User	All
**Correc.	1	1	1	2	1
Accuracy	2	2	2	1	2
**Usabil.	5	6	3	4	3
Robust.	3	4	4	3	4
Interop.	7	7	5	6	5
Integrity	8	8	6	7	6
**Maint.	4	3	8	5	7
Augment.	9	9	7	8	8
Effic.	10	10	9	9	9
**Testab.	6	5	11	10	10
Portabil.	13	13	10	12	11
**Flexib.	11	12	12	11	12
**Reusab.	12	11	13	13	13

The differences for testability and maintainability are not surprising: developers and development managers care more about these attributes because they are directly related to their responsibilities toward the software. Perceptions toward these attributes reflect their perceptions toward business need. These groups mainly dealt with in-house software developed for sale, off the shelf customized software, and in-house software developed for internal use only. They are the people responsible for developing or customizing the business software. Therefore, their perception of business need is to cut costs by developing software with the highest levels of maintainability and testability. They are concerned not just about the cost of developing the software but also for the long term cost of the software over its entire life. The results for other attributes raise questions of applicability to respondents' real experiences with software packages today. We can speculate on the inherent appeal of terms: "correctness", "accuracy", "integrity", "robustness" and their linguistic association with word "quality". Other terms such as maintainability, testability and reusability are less likely to be naturally associated with quality for those respondents who are without significant exposure to the specialized terminology of software development. This may explain why these attributes were the most important for the majority of respondents, and were ranked particularly highly by users – who as a group had little or no involvement with the software development process - certainly likely to be less than the other respondent groups.

Given the apparent agreement between users and developers on the general importance of attributes like "correctness" (very high for both groups) and "reusability" (low for both groups) we must, however, suggest that further research is needed to discover exactly why respondents ranked these attributes as they did. Such research should also investigate the results of modifying the supplied definitions, or using different but synonymous term (such as "Fidelity to Specification" rather than "Correctness").

Six software quality attributes showed statistically significant differences for the different stakeholder groups. The strongest results, and those that held up best under multivariate regression analysis, concerned three attributes: usability, testability and maintainability. While usability was ranked as one of the most important six attributes by members of all groups, users ranked it more highly than did the members of any other stakeholder roles. Importance of usability to users reflects their perception of business need. Users' business need consists of learning and using software, therefore, by definition, usability becomes very important. They are probably not interested in the software other than that it is easy to use and provides appropriate functionality.

5. Conclusions

This work explores the differences in software quality perceptions between different business software stakeholders. Three hundred and fifteen respondents ranked each of thirteen generally accepted attributes of software quality on a scale of one to seven according to their perceived importance for the piece of software most vital to that individual's work. We have identified that stakeholders required different types of software for their jobs and that majority of stakeholders in the non-development group are more satisfied with the software they are using.

The main conclusion of this study is somewhat surprising and positive in terms of its real-world implications: the null hypothesis has been largely upheld. Within this survey population few significant and systematic divergences were observed in the conceptions of software quality held between developers and users, and between managers and non-managers. Given widespread perceptions of fundamental cultural clashes between these groups, and equally widespread concern over the ability of software systems as implemented to satisfy real business needs, this is surely a reassuring finding.

Of course, the survey was administered to a group of respondents enrolled in or graduated from a leading executive MBA business school program. While the respondents filled a variety of stakeholder groups, they might reasonably be supposed to have been admitted into the program according to their managerial potential and to have been exposed to a demanding core curriculum and a strong shared culture during their studies. In this they are unlikely to be entirely representative of the broader population of users, managers and developers. Achieving such agreement in most organizations might require significant investments and the development of a strong cross-functional culture.

Within these constraints, our research suggests that a piece of software might plausibly satisfy the quality requirements of users, managers, and developers. One implication of this finding is that tactics such as formally specifying the required levels of each attribute early in the development process might win agreement across roles [5]. In particular, developers and developer managers were in agreement on software attribute priorities.

The survey did reveal significant differences between the priorities assigned to a number of attributes by holders of different roles according to their perceptions of business need: usability (favored by users) and testability and maintainability (favored by development staff). This suggests that attempts to educate users and developers about each others' priorities should be focused on these three attributes. For example, users might lack an appreciation of the relationship between testability and the other attributes with which they are more directly concerned. Fortunately, the attributes are not among those widely seen as hard to achieve in combination and so it may be possible to satisfy all groups (in contrast with the negative relationships sometimes identified between attributes such as flexibility and efficiency) [12], [14]. Armed with the knowledge of these systematic differences in perceptions, project managers may also be better able to deal with and balance the necessary tradeoffs.

6. References

[1] Arthur, l. J. Measuring Programmer Quality, John Wiley and Sons New York 1985

[2] Boehm, B.W., Brown, J.R., and Lipow, M. (Eds.), “Quantitative Evaluation of Software Quality”, Proceedings of the Second International Conference of Software Engineering, (1976), pp. 592-605.

[3] Boehm, B.W., In, H. (1996), "Identifying Quality-Requirement Conflicts", IEEE Software, vol.13, no.2, pp.25-35

[4] Cerpa N. and Verner J.M., “Practitioners’ Views of Software Quality”, The International Journal of Computing and Engineering Management,, Vol. 4, No. 3, September-December 1996, pp. 1-15.

[5] Dekkers N., "Maximising Customer Satisfaction.", Proceedings of the 12th European Software Control and Metrics Conference in London Eds K. Maxwell, S Oligny, R Kusters and E. van Veenendaal, April 2-4th , 2001

[6] Deutsch, M. S and Willis, R. R. "Software Quality Engineering" Prentice-Hall Englewood Cliffs NJ (1988).

[7] Gentleman, W. M. “Software Quality World-wide: What Are the Practices in a Changing Environment”, Proceeding of the Sixth International Conference on Software Quality (6ICSQ), Ottawa, Canada, 28-30 October, 1996, pp.335-345.

[8] Haigh, M “Software Quality Revisited: Diverging Priorities Between Stakeholder Groups?”, Ph.D. Dissertation, Drexel University, 2002

[9] Jacobs, S. “Introducing Measurable Quality Requirements: A Case Study”, IEEE International Symposium on Requirements Engineering June 07 - 11, 1999 Limerick, Ireland.

[10] Kusters, R.J.; van Solingen, R.; Trienekens, J.J.M., "User-perceptions of embedded software quality", Eighth IEEE International Workshop on Software Technology and Engineering Practice incorporating Computer Aided Software Engineering, p.184-97, 1997.

[11] McConnell, Steve. Code Complete: A Practical Handbook of Software Construction. Redmond, WA: Microsoft Press, 1993

[12] Perry W.E. “Quality Assurance for Information Systems: Methods, Tools and Techniques”, QED Technical Publishing Group, 1991.

[13] Sheehan, G. M. (1955). An Application to Payroll. In R. N. Anthony (Ed.), Automatic Data Processing Conference (pp. 145-157): Graduate School of Business Administration, Harvard University

[14] Shumskas, A. F. “Software Risk Mitigation” in Schulmeyer, G. Gordon and James I. McManus, ed.. Total Quality Management for Software. 1992: NY, Van Nostrand Reinhold. pp. 190-220.

[15] Trochim, W. (2000). “The Research Methods Knowledge Base”, 2nd Edition. Atomic Dog Publishing, Cincinnati, OH.

[16] Verner, J.M., Moores, T.T. and Barrett, R., "Software Quality: Perceptions and Practices in Hong Kong", in Achieving Quality in Software (eds) S. Bologna and G. Bucci Chapman and Hall 1996, pp. 77-88.

[17] Wallmuller, E., Software Quality Assurance: A Practical Approach, Prentice-Hall Englewood Cliffs NJ 1994

[18] Wilson, D. N. and Hall, Tracy, "Perceptions of Software Quality: A Pilot Study". Software Quality Journal, 7 pp. 67-75, 1998.

The Role of Stakeholder Understanding in Aligning IT with Business Objectives

Stasys Lukaitis and Jacob Cybulski

School of Information Systems
Deakin University
slu@deakin.edu.au, jacob.cybulski@deakin.edu.au

ABSTRACT

This article reports a study of senior management experience and their opinions on the issues of effective stakeholder communication and the evolving understanding between business and IT. In particular, we explore the impact of modern business context and practices, the issues of trust, nomenclature and the main barriers to the mutual stakeholder understanding. We find that a lack of communication and a lack of understanding between stakeholders impacts negatively on good alignment as manifested by scope creep, the desire to outsource and a lack of trust.

> *"In order to be able to ask [a question], one must want to know, which involves knowing that one does not know." [1]*

I. Introduction

In March 1991, the Software Engineering Institute (SEI) hosted the Requirements Engineering and Analysis Workshop in Pittsburgh, Pennsylvania [2]. The workshop's main objective was to explore and discuss issues concerning effective development of requirements for mission-critical systems. At the time, workshop participants were not surprised to find stakeholder communication to be a major problem in requirements engineering and in particular requirements elicitation - as stated quite unequivocally in the workshop report, *"communication is a major source of difficulty because elicitation is primarily a process of communication by its nature"* [2, p 2]. What was surprising to many, however, was the extent of communication problems leading to impaired understanding between project stakeholders and the degree of difficulty in removing the barriers to more effective communication practices. It was noted that unless properly dealt with communication deficiencies could result in a serious loss of software product quality right at the very beginning of its development cycle due to requirements omission, misinterpretation, over-specification or under-specification. Inadequate communication was also claimed to further propagate system flaws during the subsequent maintenance and the associated requirements evolution. In fact, a year later SEI researchers, Christel and Kang [3], reported some frightening statistics on the system error rates, reaching 56% and using up to 82% of the available staff time, due to poor communication and a considerable divide in understanding between users and requirements analysts. While recognising the seriousness of this situation, the organisers of the Requirements Engineering and Analysis Workshop issued a number of recommendations for improving the communication processes in requirements engineering [2, p 3 and 35-36], i.e.

> *"Improve communication by fostering contact between all stakeholders and removing management constraints. This can be achieved by educating managers and removing contractual, legal, and financial barriers between communicating groups, including modifications to the acquisition process."*

Fifteen years later, we can witness the ever-present awareness of communication issues in requirements elicitation. This awareness is clearly visible in organisational readiness to adopt stakeholder-oriented and participative system development methods, such as socio-technical design methods [4] and user-centred development [5]. This awareness is quite transparent in developing quality standards, such as CMM, which recognise the importance of effective requirements elicitation in software projects and thus strive to improving approaches to stakeholder communication and collaboration with a view to create organisation's shared vision and promoting team's integrative behaviour [6, p 65]. This awareness should also positively impact management exploits in better aligning IT solutions with stakeholder and business objectives - the new and enlarged scope of requirements engineering effort [7, 8]. It should, but has it?

In fact, this very last point created unease in our initially informal discussions with some of our senior management colleagues, who struggle daily in their attempts to align the goals of their IT departments with the core of their business, to align IT infrastructures with business processes, and to align information system requirements with business needs. The obvious discrepancy between our intuition, as based on the promise of participative information systems development and improved stakeholder communication, with the hard facts of the currently adopted IT and business practice motivated our industry-wide inquiry into the impact of real gaps in IT and business stakeholders' communication and their mutual understanding.

In our pursuits of insights on the impact of stakeholder communication on alignment [9], we have taken a commonly accepted view of alignment as related to the business scope, being a collection of key business descriptors [10, p 143-151], i.e.

- Vision and its guiding theme;
- Mission or a high-level business objective;
- Values;
- Customer / markets;
- Products / services;
- Geography and the business location;
- Strategic intent as given by the long-term objectives;
- Driving force being the primary business determinant; and,
- Sustainable strategic advantage.

In this context, *alignment* can be viewed as the process of ensuring that business is in the state of strategic fit, i.e. all business functions operate in harmony with each other to support business scope via effective :-

- Coordination;
- Perseverance; and,
- Significant concentration of effort towards business objectives.

In terms of business / IT relationship, Ward and Peppard [11, p 45] offer a demand / supply model of alignment (see Figure 1), which emphasises strategic and functional fit of business and IT domains within a single organisation. In this model, the pursuit of successful alignment of IT with the business, relies on coordinated effort in gathering requirements to establish both business demand and the technological supply, and on the ability of all parties involved to effectively communicate the business mission and objectives, organisational values and culture, information about customers and products, the primary business circumstances and the driving forces to accomplish organisational strategic advantage.

Nevertheless, as noted by Dale [12], requirements definition processes are not straightforward and are often clouded by tensions between business stakeholders and the IT group. These tensions commonly create an "emotive complexity" making it difficult to manage stakeholder expectations, and thus colouring and politicising requirements determination process, and turning stakeholder communication into impassioned negotiations and consensus making [13].

This article therefore undertakes an in-depth exploration of executives' experience and their opinions on the issues of effective stakeholder communication and the evolving understanding between business and IT and how that impacts on alignment.

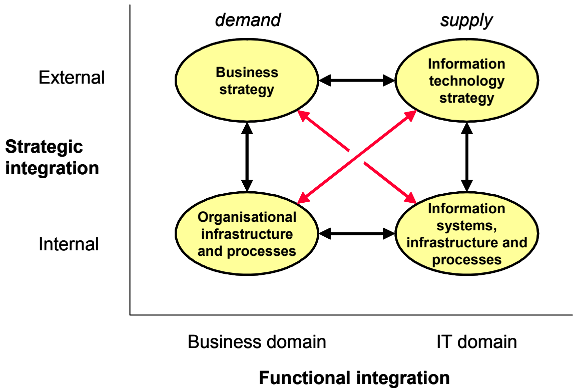

demand
supply
External
Business strategy
Information technology strategy
Strategic integration
Internal
Organisational infrastructure and processes
Information systems, infrastructure and processes
Business domain
IT domain
Functional integration

Figure 1: Business / IT Alignment Model

II. RESEARCH METHOD

The researchers conducted two focus groups [14] of senior business executives to talk about issues surrounding the alignment of business and Information Systems. The two focus groups involved a total of 16 participants.

Given the nature of the issues under discussion, the participants played quite distinct roles in their organisations, e.g. those of Chief Executive Officers (CEOs), Chief Information Officers (CIOs) and Chief Financial Officers (CFOs), project managers, senior managers and senior consultants.

The mix of organisational positions, responsibilities, tasks and views benefited the group dynamics and stimulated discussions. The focus group members represented a variety of substantial and long-standing companies in Australia, of which activities were ranging from software development and management consulting, through health care, banking and finance, to logistics and business intelligence.

The dynamics between different industry groups and the IT and non-IT executives was exceptional which is reflected in the richness of the collected data.

The initial questions that were put to both groups were about the alignment between business (problem area) and IT (solution area). The participants were asked to consider a number of propositions (such as the impact of alignment on project success) and to discuss these and to add their own experiences and knowledge (such as the impact of alignment on requirements quality) into what factors influenced this alignment. The follow up interviews, of about 90 minutes each, were then conducted with the focus groups participants to further elaborate their views and opinions.

The researchers videotaped the focus group sessions and audio-taped the interviews, which resulted in hours of video and audio streams that were later transcribed and analysed. As both focus group discussions evolved into heated debate, the videotapes captured some invaluable details of participants' interactions that is missing from the respective paper transcripts. Not only were the body language, repartee and "robust" arguments in clear evidence, but the actual way that the group dynamics drove the discussions also emerged. From the viewpoint of critical hermeneutics, the socio-political nature of the responses was quite pronounced, perhaps stimulated by the group dynamics.

It should be noted that in interpretive studies, such as hermeneutics, interviewed participants are treated on equal footing with the investigators and considered co-researchers.

The resulting transcripts of the focus groups and the interviews were the data from which the analyses were done. Given that the data is in an unstructured textual format, it was felt that a hermeneutic analysis was the most appropriate method.

All transcripts were analysed using the Ricoeur's principles of critical hermeneutics [15] to drill down through the data creating derivative documents.

Harvey and Myers [16, p20] quote Paul Ricoeur:

> *"In critical hermeneutics the interpreter constructs the context as another form of text, which can then, of itself, be critically analysed so that the meaning construction can be understood as an interpretive act. In this way, the hermeneutic interpreter is simply creating another text on a text, and this recursive creation is potentially infinite. Every meaning is constructed, even through the very constructive act of seeking to deconstruct, and the process whereby that textual interpretation occurs must be self critically reflected upon." [15]*

The very act of creating this derivative document forces the researcher to engage with the data, sorting and categorizing it artificially [1], engaging with all the components of the knowledge fragments and building them into new understanding. Critical hermeneutics, as previously adapted by Lukaitis and Cybulski to analyse some well-known case studies [17], can be shown to be of great value to identify clear cut categories and topics, and the resulting derivative documents subsequently allow quick ranking of the factors impacting some of the issues under consideration.

The adopted method [17] relies on the set of iterations - also known as hermeneutic cycles or circles - to gather small pieces of knowledge, often out of context, and reconcile these smaller pieces with the gathering horizon of understanding of the whole phenomenon. As each small piece (a morsel of knowledge) is reconciled with the whole (an understanding of a domain), the whole then becomes the horizon that contains all the knowledge. This gathering understanding of the domain under investigation then causes the existing smaller individual parts to be re-evaluated and possibly their new meanings re-integrated again into the new understanding [18, 19].

Through the hermeneutic cycle, researchers can commonly observe an oscillation between individual fragments of knowledge and the understanding of the whole of a domain. One can tell when understanding has been reached because all the data and observed phenomena are consistent, no longer appear strange and simply make sense [20]. It is often described as data saturation, when any new data neither adds to, nor detracts from the understanding developed.

That hermeneutics can be an asset in an interpretive research, such as this study of contradictory and seemingly irreconcilable views of domain practitioners, is especially evident when dialectics [21, p1197] is deployed to thoroughly investigate the "truth" or otherwise of our growing understandings of a domain under investigation. Dialectics can be understood as the search for knowledge and understanding without applying judgmental attitudes. In other words, we seek all the arguments and issues involved, irrespective of whether they are for or against the proposition under investigation. And if we find too many arguments in favour of a given position, then under the rules of dialectic, we are obliged to seek out as many arguments against the proposition.

Hermeneutics further acknowledges that the distance between the investigator and the subject can be great. Kidder states *"... what is clear and obvious to one in reading a text is likely to be a function of one's own cultural orientation and one's own prejudices rather than the function of some given accessibility of the text"* [21, p1194]. This "distance" then, can be equally ascribed to that existing between the business executive and the requirements engineer during the elicitation process, or even after requirements documents have been transcribed and are under investigation or reconciliation.

III. DISCUSSION

If one assumes that the overarching goal of requirements engineering (RE) is the ultimate delivery of information systems that are aligned with an organisation's business, then every link in the RE process is critical to this successful delivery. As succinctly summarised by Bleistein and colleagues [22, p14]:

> *"For the requirements engineer, this means that the tools and techniques must integrate means of capturing systems requirements such that they are in alignment with the highest-level of business objectives in order to ensure success".*

Bleistein et al. went on to further elaborate their SOARE approach to strategy-oriented alignment, which could potentially resolve some of the most intricate alignment problems by enlisting patterns of domain best business practice [22, p20] :-

> *"... understanding of the business model can mean knowing a large number of system requirements in advance of stakeholder interviews while also having confidence in the quality and appropriateness of those requirements thanks to cumulative industry experience".*

Such patterns therefore represent shared and reusable domain "experience" [23, 24], which could effectively be deployed to close many types of commonly encountered business / IT alignment gaps.

The main areas of such gaps strongly emerged from our first focus group, which identified eleven principle issues that bore on the successful alignment of IT with the business. These issues included management inability to estimate projects and return on investment, problems with acceptance testing, project and risk management, trust, scope creep, resistance and change management, aspects of project and product ownership, vendors and business integration, and finally, the issue which was discussed most vigorously - the effectiveness of stakeholder communication and mutual understanding.

Not surprising, stakeholder communication and understanding by Executives bodies to be the Achilles heel of the requirements engineering process and as such the main thorn in the business / IT alignment - this observation closely paralleled the findings by Reich and Benbasat [25]. Even with some of the benefits of the SOARE framework and its methods, well before business / IT alignment could be forged, before the patterns of best practice could be incorporated as part of the organisation's strategy, and before shared requirements could be reused, it is the stakeholder communication that negatively influences the effectiveness of requirements interviews, negotiations and meetings, and which defines the quality of interaction between the project initiator, management, requirements engineer and the end users.

The stakeholder problems are further confounded, as Gadamer [1, p387] resolutely states, not only by the communication media, such as language, but also - and more importantly - by the communication subject matter and its understanding. Recent studies [13] suggest that understanding gaps between requirements engineers and business can be quite pronounced, and the resulting tensions between the stakeholder communities could in fact lead to organisational or inter-organisational conflict [23].

As was repeated in both focus groups and overwhelmingly reiterated in our interviews, the primary issue mitigating against good alignment was indeed "understanding", stemming from poor stakeholder communications. Interestingly, the recurring theme of this lack of understanding was being attributed as the fault of both the business executives and also the IT group. We will illustrate these issues with some of the collected data.

In the hermeneutic-dialectic tradition [20] we will make our co-researchers' participation in the dialectics clearly visible, and thus we will let them speak for us in the following sections.

It seems that, in general as clearly felt by some of our participants, IT people feel a frustration that the business people appear not to have a sufficiently detailed grasp of their requirements (note that the initials in brackets indicate the participant's code).

[BS] *That is the senior managers don't understand their business processes down to a level of granularity and detail that they need to, to make wise decisions about which part of this process can be changed this way and that way with the technologies. That's my view. And the ownership and responsibility moved out of the technology camp into the business camp.*

[BS] *Of actually having a, what we called systems analysis and design – those disciplines being learnt by the business folk and going through the process mapping. And, the business folk don't understand the detail we need it necessarily. Particularly at the senior management level who are trying to make a strategic decision.*

This frustration seems to get quite heated. What becomes evident is that the IT side of the understanding chasm suspects that there is some detail, some deeper understanding of the business that they are unaware of, yet need to know to enable a system to operate correctly.

[WD] *But when it comes down to the alignment to the business there's two parties. There is IT and there is the business. And I think both are at fault at this. But it's totally different trying to expect that the business sponsors that we deal with are going to have an adequate understanding of IT. So if those business leaders don't understand that one concept, that it is their business, they will not survive two hours in the marketplace without that system running. I think that is the biggest initiative we can push across them.*

[WD] *And I think that probably we are forced, have to go back to business to push back and say "if you don't understand it, you'll have to understand it, otherwise it will fail".*

The IT participants alluded to their belief that business executives needed to better understand the technology and how it can be better used. But it is not all about just a simple appreciation of how technology plays a part in a successful business, there is also the understanding of the business itself.

During the first focus group the dynamics between the business participants and the IT participants was quite interesting when one IT executive suggested that both sides of the understanding equation were at fault.

[BS] *You need to understand what you are trying to achieve in the business model and business model changes. What does that mean to my processes and how can I get a grip on them? That debate is not uniformly high level I have to say on both the technical side and on the business management side [smiling broadly].*

The response from the banker appeared to recognise the need for a better understanding between the different parties, even acknowledging that different parts of businesses are also quite unique…

[PC] *Is that businesses are all different and bits of businesses are different. This is basically interpersonal stuff [interjections of agreement from CF], it's about relationship building and about being able to understand who it is you are trying deal with and how you need to operate in respect to that particular piece of culture that you are operating with. Which touches on what Bob [point towards BJ] talked about earlier on. And the other thing, my third and final one just carries;*

... your point forward a little bit further is that there really needs to be a level of understanding and consideration for the position of the other person in the process. And what do I know about what I am talking about. And I'm not the expert, I need your help. That's why I am seeking to engage with you in this process to get to the end. And as a broken down old salesman, the concept of mutual gain has to permeate right through the whole process. There's got to be mutual gain [mumbles of agreement all round].

And the sharing of knowledge now needed between business and IT because of increased complexity…

[BS] I mean the point I was getting to in a lot of this, is I see the responsibility of understanding of information flows and modelling information flows in an organisation which is sort of what we're all about, and making it concrete in technology. Realising it in technology. The understanding of that has moved from the purely IT end of the spectrum and has now been picked up the systems and process understanding is becoming required on the business side, for businesses to actually understand their own business models, their own information flows. Because we have much more complicated businesses, interactions.

Doing business in China, marketing into Europe and North America is not something that is done by a couple of people with a couple of good ideas There's all of that happening, but you've got the information flows [which] are now global. And tracking the economics and logistics and all the rest of it is reasonably demanding. It's a much more complex problem. What I'm getting at is we're only part way through the process and business people are picking up on that [interrupt CF "Totally agree"].

Nevertheless, senior executives from business appear to be quite concerned that IT seems to be unable to understand what is needed unless it is spelt out in some considerable detail. This theme where the business appears to be almost "putting up with" IT's inability to understand the detail of the business requirements keeps emerging throughout these encounters. This seems at odds with the claims of the IT people that business "doesn't understand enough of IT to be able to help".

It would seem that "understanding" simply does not exist between the two camps.

[MD] What we, what we find I guess is that whenever we request anything we actually have to go into a lot of detail to actually tell them exactly what we want it to do, and you know what options we want; what parameters it needs to be based on; what the desired outcome is. Otherwise, they'll go away and come up with this is what the software can do and just say that's it – take it or leave it. So you have to go into a lot of detail to actually explain to them exactly what the need is; why it's required; what the software, what we'd like the software to do and what the outcome is, that it's needed

This seems to be confirmed from the IT camp by a throw-away remark made during a follow-up interview…

[PR] ...and maybe really our problem is in requirements. Well their problem probably is in requirements and that's where most people have their most largest [expletive deleted]-ups.

Once the data from the follow-up interviews and the second focus group are woven into the hermeneutic cycles, the key findings begin to emerge.

It is useful to remember that because of the nature of this qualitative research the amount of data coming in to the analysis is considerable. There are an enormous number of issues emerging. It is quite beyond the scope of this paper to go into any degree of detail about the "richness" of the collected data.

Interestingly, all of the problems with stakeholder communication were vigorously debated in 1980s and 1990s [26], and the communication break-downs were noted on the level of analyst / user interaction. However, now these issues re-emerge with even stronger emphasis and even wider-ranging impact on the level of executive communication.

IV. Key Findings

When hermeneutically dissecting the issues surrounding the impact of "understanding" on the overall alignment problem, a number of interesting findings emerge (See figure 2).

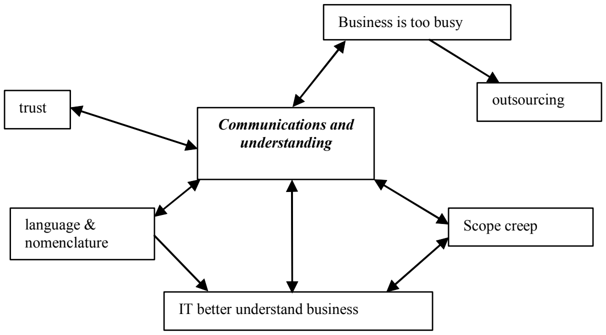


Figure 2: Emerging Issues Impacting on Communications and Understanding, and Consequently Alignment

Looking at good communications and understanding as being the overall goal (Figure 2), the departures from the ideal appear to be either from a simple lack on interest by the business – "Business is too busy", through to IT not having a sufficient grasp of what their businesses are about.

Thus where the business people show a lack of interest in IT, there appears to be a relationship with their desire to outsource some or all of the IT function. Similarly, where IT shows a lack of understanding and communicative ability, then scope creep emerges as well as a lack of shared language.

Trust seems to either act as a lubricant for communications and understanding between the business and IT, or as a resistor or abrasive between the two.

A. Business is too busy

Throughout the discussion so far, it has been repeatedly raised that the responsibility for ensuring that communications has occurred effectively rests with IT, not business. Business is too busy to learn enough about IT to be able to talk with IT people on IT matters.

[CF] *I think the first level is that there is just generally conceded by business people that are non-technologists that it's a level of technical understanding that they can't have and don't want to have.*

One CIO remarked that business is now engaging at such a complex level that there is great difficulty just understanding the processes that go on, and in engaging the right people at the right time.

[BS] *That's where we got to on that project I described as business led with a [expletive deleted] you just have to do this and this and so here's a prototype. Yeah that's ok but you just need this bit and you know it looks pretty good and then we involve more people from the business and they said oh [expletive deleted] no you've got to do all this other stuff. Then we got through that then somebody else came in from the business and said no! Over here we've got 19 different services that we offer and they are all tracked with different rates – and it just explodes. That was really badly done. That's an example of not involving knowledgeable people across the businesses at the right stages and finding out as you went. And that prototype builds took over a year while we were battling synchronising databases, foreign databases and those sorts of things.*

And in some cases the business went one of two ways. Either they started to disengage with IT and simply said "this is what we want just go and do it", or they wanted to get dangerously involved.

[CP] *... some of the people in the business side they sort of say, I don't care how you do I just want you to do this, you go away and you work it out cause that's why I'm paying you lots of money or whatever.*

It's one of those things, is it really the IT's responsibility to understand it or it is, are we going to be asked in the business people to become IT literate, literate to a point where they're coming up with a solution for you?

The problem with that is when they do do that is because they don't a lot of times understand the IT side of things, they are creating the Ben Hur's of the world.

B. Outsourcing

The outsourcing issue emerged quite strongly as a response to the "I don't care how it is done, so long as it is done and done cheaply" attitude. It seems that some businesses have become so disenchanted with their own IT people and the difficulties associated with them that they become disenfranchised.

In extreme cases, some companies determined that IT was not their core business and opted for outsourcing as a way of divesting expensive energy away from the business to an outside body. They did not want to know about IT, they did not care about IT, all they wanted was for it to be done.

[CP] *...you get it from a different perspective when they have outsourced, because when they outsource, that's why they outsource in the first place - a lot of the companies is because they just don't want to know [about their IT]. They don't really care, they just want it done. IT is seen as one of the most expensive things out there that is costing, that the company is wasting their money on. IT is very expensive in comparison to the rest of the organisation out there.*

[A-IH] *As long as it works I don't care.*

[A-IH] *It just doesn't matter?*

[A-IH] *It doesn't matter. It doesn't matter where it comes from.*

In the repartee that surrounded the focus groups and the subsequent follow-up interviews, an interesting contradiction appeared. On the one hand we have some pretty large (say) finance/banking organisations happily outsourcing extremely large components of their core IT business to external providers, and on the other hand, we find a company in the same industry space stating what looks like the opposite. They are saying that IT is their core business.

[CF] *They've, that has been an ongoing... and that's one of the things that sort of fires me up and engages me is that in financial services particularly, it seems particularly that the product is the system – the system is the product. You know there's a piece of plastic at the end but the product and the way it's run, charged, fees, all that kind of stuff sits in the system. And for a long time it was considered throw it over the wall – it's an IT problem.*

The outsourcers, on the other hand, often take in some of the IT people directly from that business and use them and maybe their infrastructure as part of the outsourcing arrangements. That way, the existing business knowledge (i.e. understanding) or intellectual capital is not entirely lost.

[CP] *...the organisation has agreed with that because a lot of organisations actually say we will outsource but only if you employ 80% of our staff or 30% or whatever it may be.*

The outsourcers then found that after numerous acquisitions of IT staff from companies who elected to outsource that they were slowly acquiring individuals with expert domain knowledge in various industry groups.

C. Scope creep

Scope creep can be attributed to being a symptom of poor communication and understanding. However, in the discussions with our co-researchers scope creep has been found to be perceived in two ways. Either in a pejorative sense where additional functionality is being added to a project potentially jeopardizing its success, or as a way of both parties (IT and business) better understanding each other's needs and capacities.

It is curious that throughout the investigation that it was not possible to find agreement about this issue. On one hand we had the example of an IT consultant being quite intolerant of scope creep…

[WD] *I think scope creep is initially an IT stuff up. I'm working on the basis that people, IT people, have done what their doing before, so the scope is the first part of the project and you need to identify what it is from there.*

Then once the pejorative sense of the term was discarded two quite distinct understandings of scope creep began to emerge. The first came exclusively from the business end of the group.

They acknowledged that the world is a changing place and the flexibility had to be considered because of changing circumstances. The best argument offered was about a long-term project that was well underway when the Australian Government announced the creation of a Goods and Services Tax (GST). That particular project had an instant scope creep – the addition of an allowance for the GST. It was simply not negotiable.

[A - IH] *The world's ever changing so if you think you've got an agreed scope on day one, depending on how long the project is, by day ninety the world may well have changed and that also will, well could be scope creep. It could be got to do something different, good flexibility. It could just mean you've got to be flexible.*

Because of the cognitive and experiential distance between the business and IT it often took some time for understanding to flow freely between the two. Scope creep was thus seen as a resolution of understanding rather than an extension of functionality.

[A - IH] *I'd call it clarification if it was there in the first place.*

[Q - IH] *They've misunderstood?*

[A - IH] *Misunderstood, yeah.*

It was interesting to observe that these comments were more often than not made by the business based individuals rather than the IT people in the group of participants. The IT people were "less forgiving" about scope creep.

[PR] *This is really nobody's fault in some ways. I mean it is of course somebody's fault, but this can happen and the fact is that this means you do have scope creep. I mean what has happened is we had an imperfect understanding.*

Traditionally, scope creep is managed as part of the overall project management charter (whichever one you follow). It is treated as an aberration and as a threat to the overall health of a project. One individual described it succinctly…

[AP] *That's why I define scope in these terms. You manage scope creep by ensuring that any changes in any of those parameters including the dollars spent are treated as a scope change and goes to steering committee for resolution where it gets [expletive deleted]. Scope creep occurs because of uncertainty, because at the start you don't have a detailed analysis of all the business areas. As you go into that detailed analysis of course people will come with thoughts and say we meant to do this or we didn't understand that it didn't include this or why don't we do that. There is a lot of that sort of discussion before you finalise your requirements.*

And again we notice the familiar term of "understanding" creeping into the discussions. This lack of understanding having a rippling effect right down through the course of the project.

D. Trust

Trust suffers as a consequence of reduced communication and understanding. It was raised as an issue in that business did not trust IT for a variety of reasons. Among the issues preventing this trust was IT's inability to correctly estimate its figures and timelines.

[PR] *When you have a total discrepancy between an ability to forecast what costs are going to be for these things and what they are not going to be, then you can't get any kind of business alignment. Because business doesn't trust IT. IT's numbers are wrong and IT's numbers are continuously and perennially wrong. And so therefore even very good projects, very good projects can be canned because their initial forecasts are wrong.*

Sometimes IT have a habit of purposefully inflating their estimates of costs and that might impact the degree of trust that business has in them. However, one of the CFO participants felt this was not specifically an IT trick and that most budget submissions had a degree of "fat" in them.

[IH] *I mean you always get the people who over-estimate the costs of things and they do it a couple of times and then you automatically compensate for it. You know if they say well this is going to cost a hundred grand, you'd know that whenever they say a hundred grand it really means fifty because they've got a buffer up their sleeve.*

[Q - IH]: *So this is just something you expect?*

[A - IH]: *Yeah. And they're no different to anyone else. Everyone would put in a budget higher than they need to make sure they can deliver.*

Emotion plays a part in trust as well. The business has an need that is often coloured with an emotional response and it is IT's responsibility to turn that around using a suitable methodology. Achieving this has shown to be extremely beneficial in engendering trust between business and IT.

[CF] *And we've also, we've found the most use of building trust is where people come with an emotional response and you're able to turn it around using a methodology.*

And my favourite is this failure modes effects analysis where people come and say I'm scared about; I'm nervous about.

And the best way to build trust at that point is to say I want you to articulate that to me and I want to put it into this process so we can work out why you're afraid, and again it's leading people to this level of simplicity.

Another unfortunate effect of the loss of trust is that the IT group can lose their independence and self determination.

[IH]: *I think there's a lot more scope to do things if there is trust. I think you very rapidly lose control if there's no trust. You typically get told specifically what to do and expect it do exactly that and nothing else if there's no trust.*

E. Language and nomenclature

In an effort to improve the chances of better communications occurring between business and IT, one organisation renamed the traditional IT roles into titles that reflected better the individuals' relationship with the business units. Names such as "architects" were used in preference to business analysts or systems analysts.

[CP] *We have that a lot with, I've seen it a lot with the architectural space as well because they may have not been called architects, they may have been called business analysts or project managers in their own business but really that's what they were doing. They were creating requirements documents. They may not call it a requirements document but that's what they were doing. They were identifying what was the business need and putting together some form of proposal, solution, this is my options paper or whatever you want to call it. It is difficult. What happens though is that sometimes having them being moved into different parts of the organisation helps.*

In some cases, these roles were carried out by non-IT trained people because of their expertise in the business. This was the case in recent core banking application's project.

[BJ] *So we had so that all the departments, there were about eight departments – loans, credit control, finance, the whole lot, that all had to put their expert on the team, and we did that. But what we found, and the whole idea of having these departments involved for twelve to eighteen months was that they had the expertise in the areas.*

So that when we had builds or upgrades they could do it.

F. Better IT understanding of the business

Several of the participating businesses actually placed their IT staff into the target business units for several months so that they could learn about the business. The experience of working with the business gave the IT people insight into the local issues.

[CP] *What happens is, it's really being able to put in those people in place that are able to see the business side of things and also able to have IT knowledge.*

That goes back to employing the right people I guess at times and also being able to put in, those people have to have the two areas of knowledge to be able to, that's why when you really see in the insource environment that the IT department is really successful is when they have their IT people have a really good understanding of the business.

If I was to use some examples of companies I've worked for where they have had their own IT department, it has really been around the fact that a lot of their IT people and we have actually done that in some companies which is where you sort of say ok you're an IT person go and spend 3 months working with the business to understand what it is that the business really wants done and how do they really want to do it.

One company with a very low IT staff turnover noted that their IT staff were already distributed throughout the business and were very well versed in the needs and operations of the business [BS].

[BS] *It's a worry (talking about churn rate of IT staff), I mean we had 2 celebrations last month. One for a developer who has been with the company 35 years and one who has been with the company 20 years. Late last year we had one for somebody who has been 25 years. It's interesting, it's been an interesting journey but I deliberately go looking for people who, we have a number of them who are coming up to their 10th anniversary of senior IT developers who I hired 10 years ago looking for people who wanted to be around for 10 years. They were at that stage in their life and career who want stability, opportunity for growth.*

Once projects were underway, experts from the business units are brought into the project team to make it happen. All participants bemoaned the difficulties associated with getting the best people out of the business units into the project teams. One found that placing the business experts onto the IT Project payroll helped the affected business unit.

V. Conclusions

We have found that what has meant to have been a fairly straightforward stage in the requirement's engineering process for over twenty years, requirements elicitation is still fraught with difficulty and traps.

Understanding seems to be still the principal issue at stake here with continued uncertainty about stakeholders' ability to "be on the same page".

Understanding can be enhanced by ensuring that enough of the right business people are actively involved on the same level as the IT group in projects. It can also be helped by embedding IT people into the actual business units themselves, just so that they can get a better appreciation of the needs of that particular business unit.

Trust is intrinsically related to understanding and when one is high, then the other appears to follow.

If the business is sufficiently disenfranchised from their IT group there is a chance that the business might start seeing IT as not part of their core business and seek outsourcing as a way of cost containment and allowing them to focus on what they think is their core business. Business will often use terms such as "being too busy" or they "just want the job done". But this seems to happen only when the internal IT group are unable to deliver the IT that the business needs.

Scope creep has always been a problem that highlights a lack of understanding. This research has help focus on that issue by suggesting that there are several types of scope creep, ranging from the traditional additional functionality through to the clarification of understanding that we have found.

Surprisingly, business did not find scope creep to be the thorn that IT has perceived it.

Strict adherence to titles and roles has been blurred so that both domain experts and IT experts are all sharing roles and sharing the same table in an effort to enhance that alignment between business and IT.

The alignment between business and IT, nowadays considered in the scope of requirements engineering activities, was seen as occurring in small layers, similar to agile development.

[CP] *Just when I was saying we were aligned in little layers I suppose where I am talking about this team of people, this is purely from my central point of view.*

And the alignment was something that had to be maintained, nurtured. It is seen as happening at multiple levels in a project, involving varying numbers of people, and importantly, over a period of time.

[CP] *Some of the issues are that one group of people go away, they talk, they understand by then a year's gone past and a whole group of new people have come in and the trust isn't there, the ownership isn't there and the relationships aren't there.*

The understanding is not there.

Alignment is being seen as a dynamic state that is dependent on time, the relationships that exist between people, the success of communications and understanding, and the success of the business.

As observed by Luftman [27], more research, and in particular empirical study, should be devoted to the issues of strategic alignment of business and IT:

> *"While alignment is discussed extensively from a theoretical standpoint in the literature, there is scant empirical evidence regarding the appropriate route to take in aligning business and IT strategies."*

VI. Future Research

Because of the nature of qualitative research, more questions are posed than are answered. While we have identified some of the factors that impact on alignment, we have not tried to explain these behaviours. This is best left to a separate critical hermeneutic investigation using Habermas' [28, p173] theory of communicative action to explain these behaviours.

Several important issues appear to surface which could do with further investigation…

- In the communications between business and IT, what is the impact of IT practitioner experience on the effectiveness of these communications? Many companies often send in junior people to start the investigations and requirements gathering. Does this have a negative impact?
- Where a company elects to outsource their IT requirements, what is the impact of the loss of IT intellectual capital from that organisation?

VII. References

[1] H.-G. Gadamer, *Truth and Method*. London: Sheed and Ward, 1975.

[2] Cmu/Sei, "Requirement Engineering and Analysis," Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, Technical Report CMU/SEI-91-TR-30, 1991.

[3] M. G. Christel and K. C. Kang, "Issues in Requirements Elicitation," Software Engineering Institute, Carnegie Mellon University CMU/SEI-92-TR-12, 1992.

[4] E. Mumford, *Effective Systems Design and Requirements Analysis - the ETHICS Approach*. Basingstoke, UK: Macmillan Press, 1995.

[5] D. D. McCracken and R. J. Wolfe, *User-Centered Website Development: A Human-Computer Interaction Approach*. Upper Saddle River, NJ: Pearson-Prentice Hall, 2004.

[6] Cmu/Sei, "Capability Maturity Model® Integration (CMMISM), Version 1.1," CMU Software Engineering Institute CMMI-SE/SW/IPPD/SS, V1.1, 2002.

[7] B. Boehm and L. G. Huang, "Value-Based Software Engineering: A Case Study," in *IEEE Computer*, vol. 36, 2003, pp. 33-41.

[8] A. M. Davis, "The Art of Requirements Triage," in *IEEE Computer*, vol. 36, 2003, pp. 42-49.

[9] J. C. Henderson and N. Venkatraman, "Strategic alignment: leveraging information technology for transforming organisations," *IBM Systems Journal*, vol. 38, pp. 472-484, 1999.

[10] B. H. Boar, *The Art of Strategic Planning for Information Technology*. New York: Wiley, 2001.

[11] J. Ward and J. Peppard, *Strategic Planning for Information Systems*, Third ed. Chichester, England: Wiley, 2002.

[12] M. Dale, "Defining User Requirements for a Large Corporate Data Warehouse: An Experiential Case Study," in *9th Australian Workshop on Requirements Engineering*. University of South Australia, 2004, pp. 5.1-5.11.

[13] J. Price and J. L. Cybulski, "Influence of Stakeholder Communication on Concensus Making in Requirements Negotiation," in *9th Australian Workshop on Requirements Engineering*. University of South Australia, 2004, pp. 6.1-6.13.

[14] D. W. Stewart and P. N. Shamdasani, *Focus Groups Theory and Practice*, vol. 20. Newbury Park California: Sage Publications, 1991.

[15] P. Ricoeur, *The Conflict of Interpretations: Essays in Hermeneutics*. Evanston: Northwestern University Press, 1974.

[16] L. J. Harvey and M. D. Myers, "Scholarship and Practice: The Contribution of Ethnographic Research Methods to Bridging the Gap.," *Information Technology & People*, vol. 8, pp. 13-27, 1995.

[17] S. Lukaitis and J. Cybulski, "A Hermeneutic Analysis of the Denver International Airport Baggage Handling System," presented at Information Systems Foundations: Constructing and Criticising, Australian National University, 2004.

[18] W. Dilthey, "The Rise of Hermeneutics," in *The Hermeneutic Tradition: From Ast to Ricoeur*, G. L. Ormiston and A. D. Schrift, Eds. New York, NY: State University of New York Press, 1990.

[19] F. D. E. Schleiermacher, "The Hermeneutics: Outline of the 1819 Lecture," in *The Hermeneutic Tradition: From Ast to Ricoeur*, G. L. Ormiston and A. D. Schrift, Eds. New York, NY: State University of New York Press, 1819.

[20] M. D. Myers, "Dialectical Hermeneutics: a Theoretical Framework for the Implementation of Information Systems," *Information Systems Journal*, vol. 5, pp. 51-70, 1994.

[21] P. Kidder, "The Hermeneutic and Dialectic of Community in Development," *International Journal of*

Social Economics, vol. 24, pp. 1191-1202, 1997.

[22] S. J. Bleistein, A. Aurum, K. Cox, and P. K. Ray, "Linking Requirements Goal Modelling Techniques to Strategic e-Business Patterns and Best Practice," presented at 8th Australian Workshop on Requirements Engineering, University of Technology, Sydney, Australia, 2003.

[23] P. K. Sarkar and J. L. Cybulski, "Analysis of Stakeholder Concerns With a View to Avoid Organisational Conflict in B2B Systems," presented at The 15th International Conference in Electronic Commerce, Bled, Slovenia, 2002.

[24] P. K. Sarkar and J. L. Cybulski, "Process of Requirements Evolution in Web-Based Information Systems," presented at Twelve International Conference on Information Systems Development ISD'03, Melbourne, Australia, 2003.

[25] B. H. Reich and I. Benbasat, "Factors that Influence the Social Dimension of Alignment Between Business and Information Technology Objectives," *MIS Quarterly*, vol. 24, pp. 81-113, 2000.

[26] C. Urquhart, "Exploring Analyst-Client Communication: Using Grounded Theory Techniques to Investigate Interaction in Informal Requirements Gathering," in *Information Systems and Qualitative Research*, A. S. Lee, J. Liebenau, and J. I. DeGross, Eds. London: Chapman and Hall, 1997, pp. 149-181.

[27] J. Luftman, "Assessing business-IT alignment maturity," *Communications of AIS*, vol. 4, 2000.

[28] D. Macey, *The Penguin Dictionary of Critical Theory*. London: Penguin Books, 2000.

Predicting good requirements? A Pilot Study

June M. Verner
Empirical Software Engineering
National ICT Australia Ltd.
Sydney, Australia.
june.verner@nicta.com.au

Karl A. Cox
Empirical Software Engineering
National ICT Australia Ltd.
Sydney, Australia.
karl.cox@nicta.com.au

Steven J. Bleistein
CSE,
University of NSW &
National ICT Australia Ltd.
Sydney, Australia.
steven.bleistein@nicta.com.au

Paul L. Bannerman
Empirical Software Engineering
National ICT Australia Ltd.
Sydney, Australia.
paul.bannerman@nicta.com.au

Abstract

We surveyed software practitioners from business organizations in the U.S. regarding software development practices used during recent projects. We used chi square and correlational analyses to investigate the relationships between project practices and good requirements. We report on five groups of questions broadly related to requirements: 1) the sponsor, 2) customer/users, 3) requirements issues, 4) the project manager and project management, and 5) the development process. We compare our results against the software engineering research literature. Using logistic regression, the best predictor of good requirements was 1) the project had a well-defined scope, with 2) customer/users had a high level of confidence in the development team, and 3) risks were controlled and managed by the project manager.

1. Introduction

Good requirements are significantly related to successful software project outcomes [45]. Requirements management is one of the first steps in the software development process, with implications that extend throughout the entire project [9]. Organizations that implement effective requirements engineering (RE) practices reap multiple benefits, with great rewards coming from the reduction of rework during later development stages and throughout maintenance [47]. Hull *et al.* [22], suggest that activities related to getting good requirements such as user involvement, management support, a clear statement of requirements, realistic expectations and ownership, account for over 46% of successful projects. Other evidence suggests that some of the most common and serious problems associated with developing software can be traced back to requirements management [27] with incomplete requirements, lack of user involvement, unrealistic expectations, lack of executive support, changing requirements and specifications accounting for 64% of project failures [22].

When requirements are poorly defined and RE processes are *ad hoc*, the end result is nearly always an unsatisfactory product or a cancelled project. A Standish Group Report revealed that three of the top ten reasons for challenged projects or outright project failure were lack of user involvement, unstable requirements and poor project management [40]. A survey of twelve UK companies found that requirements problems accounted for 48% of all software problems [21]. Another survey of 150 companies in the U.S. showed that the majority requirements modelling technique of choice was "none" [31].

RE can be simply described as identifying a customer's problem's context, and within that context, locating the customer's requirements and delivering a specification that meets customer needs. There are many requirements methodologies that purport to do this, for example, soft systems methodology [10], scenario analysis [8], and UML [6]. Sometimes they work, and sometimes they do not. The implication of such requirements methodologies (if we can label at least aspects of them as such) is that the application of 'x' method will produce the right requirements irrespective of the problem's characteristics. This is conventional wisdom and, unsurprisingly, the creators and vendors of requirements methodologies claim (with one exception [23]) that their approach is a hammer and all problems are nails.

Concern has been expressed about the lack of RE in industrial projects, with managers suggesting that there is a need for more practical RE research [48]. While there is plenty of prescriptive research proposing new and better ways to do RE, we believe that it is important to examine what methods are used in practice and which practices lead to good requirements. In a world of constrained resources it is essential that project managers understand which practices work and which do not. We need to be aware of what is really going on, to be able to position our research within an appropriate context [13]. In order for us to have a better idea of which practices work where, research on effective RE practices should be done within different industrial contexts [48]. Without this, we will forever practice our art in a context-free bubble.

Our paper is organized as follows: in Section 2, we describe our study and discuss some details of the questionnaire responses; in Section 3, we discuss the results of the questionnaire; and in Section 4, we present our results. Section 5 provides some conclusions and suggestions for further research.

2. Our Study

To document practitioners' views regarding software project success and failure, and the practices they consider important to successful projects, we conducted wide-ranging structured discussions with twenty-one senior software practitioners employed by a major U.S. financial services company. We later had similar discussions with another group of U.S. software practitioners working in a variety of companies. Based on these discussions we developed a questionnaire to investigate those software development practices that lead to successful project outcomes. We chose a survey because of its simplicity and our wish to find relationships amongst variables.

The original practitioner group responded to our questionnaire by answering it twice, once for a recent project they considered successful and once for a recent project they considered a failure. Our questionnaire was later distributed to the second group of practitioners. Our sample is not random but rather a convenience sample of practitioners known to us.

The questionnaire was organized into a number of sections covering the entire software development process. We asked respondents if they considered the project they referenced when answering the questionnaire, 1) to be a success and 2) if it had good requirements at some stage during the development process. We define good requirements as those that are complete and fully understood by the development team and the customers/users [34].

Only questions relating to the development of good requirements are considered in this paper. Sections of the questionnaire not considered here are discussed elsewhere e.g., [43, 44, 45].

Because most software engineering research has emphasized "technical matters above behavioural matters" [20] most of our questions focus on stakeholder behaviour during software project development. Moreover, there has been a general lack of quantitative survey-based research regarding early aspects of software development. In addition, in-house software development failure is unlikely to receive the same attention as third-party software development failures [44]. Therefore, we review in-house development and management practices with the intention of showing what practices are directly related to the development of good requirements.

We received completed questionnaires from 102 respondents, reporting on 123 distinct projects. A sample of 123 projects is a reasonable size for empirical software engineering research. As noted earlier, the majority of our respondents were developers involved with software for use within their own organizations (financial institutions, banks, pharmaceutical companies, insurance companies, etc.). The responses to the first set of 42 questionnaires described 42 separate projects, 21 regarded as successful and 21 unsuccessful. The second set of responses included descriptions of 81 unique projects reported from various companies in the northeastern U.S.

Sixty-two percent of projects were regarded as successful and 38% unsuccessful, 87% were development projects (55% successful), and 13% were large (in terms of effort) maintenance/enhancement projects (66% successful). The percentage of projects by number of full-time IT employees is 1-4 = 44%; 5-9 = 19%; 10-19 = 21%; 20-29 = 7%; 30-39 = 3%; 40-99 = 5%; and 100-180 = 1% (range 1-180, mean 13, median 6).

3. Results and Analysis

The developers we surveyed mainly develop in-house software for their organization's use. The organizations rely heavily on software for many of their business functions. While we would not assume that our results are typical of all organizations, we believe that they are reasonably typical of organizations that develop in-house software. Surveys are, of course, based on self-reported data that reflect what people say happened, not necessarily what they actually did or experienced. Because we surveyed software developers, our results are limited to their knowledge, attitudes, and beliefs regarding the projects and project managers (PM) with which they were involved. However, as the majority of projects are fairly small (63% employed fewer than 10 people and 84% fewer than 20), we believe that our respondents have a reasonable knowledge of most project events. However, the results may be biased by the preponderance of small projects in our survey sample.

The percentages of "yes" responses to the survey questions are shown in Table 1 (see footnote to Table 1 for more detail). Table 2 shows which variables have a significant association with good requirements as well as some associations between responses to selected questions. We performed chi square tests to determine the degree of association between variables, and correlation analyses to provide the direction of that association. In the rest of this paper, if a pair of variables is significantly associated (<0.05) and positively correlated, we refer to them as *significantly associated.* When it occurs, we mention negative association explicitly. If we refer to practitioners we are referring to those practitioners who participated in our discussions.

In Tables 1 and 2, our questions are classified as follows: "S" refers to questions that deal with the project sponsor/senior management, "C" refers to customers and users, "R" refers to questions directly related to specific requirements issues, "M" to questions related to the project manager and project management, and "P" to questions related to the development process.

3.1 Project Sponsor/Senior Management

A powerful political sponsor can assure that a project is adequately resourced, and that customers and users make sufficient time available for requirements gathering. Politically powerful sponsorship may ensure that other senior managers do not hinder the project, e.g., through reassignment of essential personnel (although, this may depend on factors such as the size of the organization and priority changes within it). A high level of sponsor participation can support realistic scheduling and resource planning by preventing unrealistic schedules, schedule changes or other undermining changes [28, 29]. Similarly, sponsor participation can help enhance control practices [28]. A committed sponsor is important to software project success because he or she impacts a project throughout its life-cycle [28, 35, 36]. Loss of sponsorship or failure to properly establish it can indicate that the project is in jeopardy [28, 34].

All "S" variables were significantly associated with good requirements. All associations were positive except for S4, which was negatively associated. The variables S1, *the project began with a committed sponsor*, S2, *sponsor commitment lasted through the project*, and S3, *the sponsor was involved in project decisions,* showed a high degree of multi co-linearity. Surprisingly, there was no correlation of S1, S2, or S3 with S4, *senior management negatively impacted the project.* This finding appears to contradict practitioners' initial comments that powerful sponsorship protects a project against interference from competing interests. We suspect this may be because of changed organizational priorities or because the project sponsors were not senior enough to protect the project from external interference.

Using logistic regression with the responses to "S" questions, the best predictor of good requirements was S2 (*sponsor commitment lasted right through the project*) which predicted 86% of projects with good requirements, 65% of projects without good requirements, and 76% correctly overall.

In summary: the relationships between all of the "C" variables and S2 are in agreement with research that stresses the importance of a committed sponsor whose commitment lasts throughout the project [39]. Having a committed sponsor who lasts the distance lends stability to requirements.

3.2 Customer/users

Good requirements are traditionally viewed as the outcome of a positive relationship between customers/user and the development team. This is especially important as customers and users often cannot easily articulate what they really need at the start of the process. Further, users are rarely experienced in requirements elicitation, particularly at the necessary level of detail [7]. Unrealistic customer and user expectations can arise because projects start with incomplete requirements [46]. Hence, an explicit user-inclusion strategy should be used for effective requirements gathering as user support and enlightened involvement are important for ownership [17, 28, 33, 38, 42].

Evidence shows that a high level of customer/user involvement throughout the project, from requirements elicitation to acceptance testing, is necessary for project success, and helps with "buy in" to the project [40]. Customer/user participation can reflect confidence in the development team, positive expectations, and the desire to contribute knowledge of the business needs. Of course, if there is a large number of customers/users, it is more difficult to ensure a feeling of involvement. Representative groups of customers/users must be carefully identified. User participation supports more realistic expectations, which reduces conflict [17]. An amicable relationship between customer/users and the team reduces distractions, resulting in a more efficient development effort leading to higher motivation among team members [28].

While user participation has far reaching implications for the development process, some research suggests that users are "rarely involved in product development" [42]. Furthermore, stakeholders often see requirements effort as a disruption to their work [37].

We found a high degree of multi co-linearity among C1, C2, C3, C4, C5, and C6, suggesting that (C1), *a high level of customer/user involvement,* may result in (C4), *commitment and involvement of other stakeholders,* (C2), *customers and users having a high level of confidence in the development team*, and (C3), *involved customers and users will then stay right through the project.* Our analysis also suggests that C1 is very important as *a high level of customer/user involvement* may lead to (C6), *customers/user will have realistic expectations,* and (C5), *they will make adequate time available for requirements gathering,* thus implying good requirements. The importance of user involvement in requirements gathering (C5) supports observations of both Clavadetscher [11] and Glass [18]. Contrary to what practitioners had suggested in our initial discussions, we did not find that large numbers of customers and users impacted the development of good requirements.

Using logistic regression with the responses to the "C" questions, the best predictor of good requirements was C4 (*customer/users had a high level of confidence in the development team*) which predicted 86% of projects with good requirements, 73% of projects without good requirements, and 77% correctly overall.

The relationship between customer/user involvement (C1) with level of confidence in the development team (C4) is interesting and leads us to ask about causal effects. Are customers/users involved because they are confident in the development team or do they become more confident in the development team because of their involvement? Certainly there is likely to be a reciprocal effect: a positive involvement experience is likely to reinforce confidence in developers and vice-versa. This establishes a collaborative environment, which leads to a win/win outcome [47]. Accordingly, development teams that do not present themselves well to users and manage customer/user expectations, may be sowing the seeds of failure.

In summary:

- We were surprised that large numbers of customers and users did not impact establishing good requirements. This may reflect the relatively small size of the projects in our sample. Further research will clarify the effects of large numbers of customers and users on the requirements elicitation process.
- We were also surprised that a high level of confidence in the development team was the best predictor of good requirements. The confidence that the customers and users have in the development team is not an area typically addressed in the RE literature.
- Our research supports customers/users making adequate time available for requirements gathering as an important requirements determinant. This is one of the most frequently identified factors for the development of good requirements.

3.3 Requirements Issues

Given that control over requirements is necessary to move from the lowest CMMI level, it was clear that many of the organizations in our sample are still at the lowest level [12]. The results in Tables 1 and 2 support the view that requirements continue to be a problem for software development [19, 30]. Our results agree with [31], whose respondents thought their companies did not do enough requirements engineering.

Definition of a requirements development process at the start of a project will normally include the use of a RE methodology [47]. We found that gathering requirements with a specific methodology (R3) was not significantly associated with good requirements. This may be because, in 79% of our projects, respondents did not know what requirements methodology was used. For the ones that did know, three projects used prototyping and six used JAD sessions with prototyping. For the remainder of projects, interviews and questionnaires were the main requirements gathering method. Eight of the nine projects using prototyping and/or JAD had good requirements. Eight projects used UML to document requirements; five had good requirements, though only four projects were successful. The use of UML as a requirements modelling notation has been criticized [23, 26]. Robertson and Robertson [37] suggest that "most UML models are not appropriate for requirements work, while they are good design models they are lousy requirements models". Our results lend support to the view that the value of UML for requirements still needs to be established. Using UML, however, was better than using no methodology at all.

It is essential to manage requirements throughout the development process [47]. We found that R1, *there was a central repository for requirements,* was significantly associated with good requirements, indicating that a central repository supports effective requirements management. The fact that only 59% of the projects used a central repository tends to support the view that there is significant room for improvement in requirements management.

Practitioners suggested that large projects, in terms of functionality, are less likely to be successful than smaller projects. R2, *project size impacted elicitation of requirements,* was significantly negatively associated with good requirements. This result agrees with [18], suggesting that project size hampers requirements gathering, and can lead to unclear, incomplete, and potentially unstable requirements.

Wiegers [47] addresses a number of good RE practices including the need for a well-defined project scope. R4, *the project had a well-defined scope,* and R5, *project scope increased during the project,* were both significantly associated with good requirements, R5 negatively. An increase in scope and creeping requirements pose major risks to software projects [24]. R5, *project scope increased during the project,* was significantly associated (0.000) with R2, *project size impacted elicitation of requirements.* The longer the project goes on, the more growth in scope developers are likely to experience [25].

Table 1: Percentage "Yes" Responses to Questions

ID	Question	With good requirements[1] % Yes	Without good requirements[2] % Yes	All projects[3] % Yes
S1	The project began with a committed sponsor	90	63	80
S2	Sponsor commitment lasted through the project	82	35	66
S3	Sponsor was involved in project decisions	77	40	64
S4	Senior management negatively impacted the project	19	38	26
C1	High level of customer/user involvement	73	43	62
C2	Other stakeholders were committed and involved	73	36	60
C3	Involved customers/users stayed right through project	82	57	73
C4	Customers/users had high level of confidence in development team	73	14	52
C5	Adequate time made available by customers/users for requirements gathering	80	43	68
C6	Customers/users had realistic expectations	63	17	46
C7	Customer/user's expectations managed throughout	82	40	65
C8	Problems caused by large numbers of customers/users	23	38	28
R1	There was a central repository for requirements	69	35	59
R2	Project size impacted elicitation of requirements	26	51	35
R3	Requirements gathered using specific methodology	56	35	49
R4	Project had a well-defined scope	84	33	67
R5	Project scope increased during the project	58	80	66
R6	Requirements were managed effectively	69	23	51
M1	Project manager given full authority to manage project	70	50	63
M2	PM was above average	68	23	54
M3	PM related well to staff	69	26	56
M4	PM had a clear vision of the project	83	49	72
M5	PM really understood the customers problem	75	46	65
M6	PM communicated well with staff	65	23	52
M7	PM was experienced in the application area	68	68	68
M8	Years of experience of the PM < 10	66	82	71
M9	Project manager's background (IT, Business, other)	47, 38, 15	65, 26, 9	53, 34, 13
P2	Development methodology appropriate for project	62	23	47
P3	PM able to choose the methodology	34	39	36
P4	Risks identified at the beginning of the project	80	33	62
P5	Risks incorporated into the project plan	66	32	53
P6	Risks controlled and managed by the PM	63	10	43
P7	Project had effective change control	73	30	60
P8	An approach to control quality used	76	37	59
P10	Other projects negatively impacted this project	25	57	38

If a PM has a sufficient vision of the project and begins with a well-defined scope then this is a first step in managing scope creep [47]. R4, the *project had a well-defined scope,* and R5, the *project scope increased during the project,* were both significantly associated with M4, the *PM had a clear vision of the project* (0.000, 0.004), R5 negatively. Good requirements management impacts the cost of developing software as it helps to alleviate costly rework [3, 4, 5]. Problems such as missing functionality are considerably more expensive to correct later in the development process [5, 27, 28]. In agreement with this research, R6, *requirements were managed effectively,* was significantly associated with good requirements. There is a high degree of multi co-linearity between most of the "R" variables. Analysis of our data suggests that a project with good requirements is a *project with a well-defined scope* (R4), *that did not increase during the project* (R5), has *a central repository*

[1] This column represents the percentage of "yes" answers to questions for projects that had good requirements

[2] This column represents the percentage of "yes" answers to questions for projects that did not have good requirements

[3] This column represents the percentage of "yes" answers to the questions for all projects.

for requirements (R1), and that *requirements were managed effectively* (R6).

Using logistic regression with the responses to the "R" questions, the best predictor of good requirements was R4 (*the project had a well-defined scope*) which predicted 85% of projects with good requirements, 77% of projects without good requirements, and 82% correctly overall.

Table 2 Correlations of Questions to Good Requirements and to Some Other Questions (123 cases)

ID	Question	Direction of relationship	Sig. relationship with good requirements	Sig. relationship with other questions
S1	The project began with a committed sponsor	+	0.001	S2, S3
S2	Sponsor commitment lasted through the project	+	0.000	S1, S3
S3	Sponsor was involved in project decisions	+	0.000	S1, S2
S4	Senior management negatively impacted the project	-	0.034	C4(-), M1(-), M2(-), P6(-)
C1	High level of customer/user involvement	+	0.001	S2, C2, C3, C4
C2	Other stakeholders were committed and involved	+	0.001	S2
C3	Involved customers/users stayed right through project	+	0.007	S2,C1, C4, C6
C4	Customers/users had high level of confidence in development team	+	0.000	S2,C1, C5, C6, C7
C5	Adequate time made available by customers/users for requirements gathering	+	0.000	S2,C1, C3, R4
C6	Customers/users had realistic expectations	+	0.000	S2,C1, C3
C7	Customer/user's expectations managed throughout	+	0.000	S2
C8	Problems caused by large numbers of customers/users		NS	
R1	There was a central repository for requirements	+	0.000	R3, R5(-)R6
R2	Project size impacted elicitation of requirements	-	0.006	S2(-),R4(-), R5
R3	Requirements gathered using specific methodology		NS	C5
R4	Project had a well-defined scope	+	0.000	R1, R5(-)
R5	Project scope increased during the project	-	0.015	R4(-)
R6	Requirements were managed effectively	+	0.000	R1, R4
M1	Project manager given full authority to manage project	+	0.043	S2, C1, C3, C6, P3
M2	PM was above average	+	0.000	M4, M5, M6, P2, P6
M3	PM related well to staff	+	0.000	M4, M6, P2, P6
M4	PM had a clear vision of the project	+	0.000	S2, C1, M1, M5, P2, P6
M5	PM really understood the customers problem	+	0.003	C1, C3, C6, C8(-) P2, M1
M6	PM communicated well with staff	+	0.000	C3, C5, R4, P8
M7	PM was experienced in the application area		NS	
M8	Years of experience of the PM < 10		NS	
M9	Project manager's background (IT, Business, other)		NS	
P2	Development methodology appropriate for project	+	0.000	R1, R4, R6, M2, P6
P3	PM able to choose the methodology		NS	
P4	Risks identified at the beginning of the project	+	0.000	P5, P6
P5	Risks incorporated into the project plan	+	0.000	
P6	Risks controlled and managed by the PM	+	0.000	
P7	Project had effective change control	+	0.000	C4, R1, R4, R6, M2, P2, P6,
P8	An approach to control quality used	+	0.001	C4, M2, R1, R4, R6, P2
P10	Other projects negatively impacted this project	-	0.001	C8, R1(-), R4(-), M2(-)

In summary:

- Our results reinforce research that identifies the importance of a well-defined scope, emphasizing that understanding the problem context and its boundaries is critical to good requirements.
- The importance of a central repository as an aid in the development of good requirements is frequently underestimated. It is surprising that fewer than 60% of projects used a repository, as it is a readily available and relatively inexpensive aid to requirements management.

3.4 Project manager and project management

Our practitioners described situations where they worked with project managers who were not given full authority to manage a project (M1). They reported that senior management constantly interfered with and second-guessed the PM. These situations led to lack of motivation and, in many cases, project failure. In agreement with our practitioners, the data shows a significant relationship between M1, *PM was given full authority to manage the project,* and good requirements. We were surprised to find that a good PM was just as likely to suffer from interference as a poor PM.

M2, *PM was above average*, is significantly associated with good requirements (even when their management of the project has suffered from interference). This result is not surprising since "poor management can increase software costs more rapidly than any other factor" [5].

In discussions, the practitioners suggested M3, *a PM who related well to staff*, was a key attribute of good project management. The results support this view as M3 was significantly associated with good requirements.

M4, *the PM had a clear vision of the project*, was associated with good requirements. Defining project vision is a good engineering practice [47]. A project that is without a clearly defined and well-communicated direction invites disaster [47]. Lack of a clear vision leads to poorly defined goals and specifications, poor requirements, insufficient time planning the project, lack of a project plan, and unrealistic deadlines and budgets [15]. This underscores the importance of understanding requirements beyond micro-level user needs [38]. M4 is significantly associated with M5, *the PM really understood the customer's problem*, (0.000), and both are significantly associated with good requirements. A clear vision is necessary for a PM to really understand the customer's problem.

Communication between the PM and the project team is also important. Project success is dependent on the quality and effectiveness of communication channels established within the development team [2]. M5, the *PM communicated well with staff*, was significantly associated with good requirements.

Common wisdom suggests that M7, *the PM is experienced in the application area*, will increase the chances of a project's success. However, our data did not support this. M7 was not significantly associated with either project success or good requirements.

Years of project management experience ranged from under 6 months to 22 years, with over 60% of PMs having more than three years experience, and 15% more than 10 years. Our practitioners suggested that an experienced PM is more likely to be associated with a successful project. The data did not support this as M8, *PM's years of experience,* was not significantly associated with either project success or good requirements.

Our practitioners also suggested that a PM with an IT background was more likely to be associated with a successful project. However, the results did not support this as M9, the PM's background, was not significantly associated with either project success or good requirements. PMs with business or other backgrounds were just as likely to be successful.

There was a high degree of multi co-linearity between most "M" variables. Analysis suggests that a project with a *PM who is given full authority to manage the project* (M1*), who is above average* (M2*), relates well to staff* (M3*), has a clear vision of the project (*M4), *really understands the customer's problems* (M5), and/or *communicates well with staff* (M6), is likely to have good requirements. These results show that, for PMs, vision, communication and relationships with team members are more important than any particular background, underscoring research that stresses the need for a PM to have good interpersonal skills [15, 16].

Using logistic regression with the responses to the "M" questions, the best predictor of good requirements was M1 (*PM was given full authority to manage the project)*, with M4 (*PM had a clear vision of the project*) and M6 (*the PM communicated well with staff*). This combination predicted 94% of projects with good requirements, 49% of projects without good requirements, and 82% correctly overall.

In summary:

- Because the project management literature generally assumes that a project manager has full authority to manage a project, we had not expected that M1, *the project manager was given full authority to manage the project*, would enter into the prediction equation for good requirements. In initial discussions, practitioners had suggested that the absence of this factor threatened project success. We were surprised that more than one third of projects were subjected to interference. Analysis suggests that, when interference occurs, it is mainly related to staffing issues, and adequate staffing is significantly associated with good requirements.

- The importance of M4, *PM had a clear vision of the project*, reinforces the importance of project scope, but includes an extra dimension; the importance of knowing expected business outcomes beyond just the project parameters.
- Effective communication is frequently suggested as a key to good requirements, and our analysis supports this.

3.5 Development Process

Good RE practices include processes such as selection of an appropriate lifecycle methodology, managing risks, specifying quality attributes, and change control processes [47].

Using a *methodology appropriate for the project* (P2) is significantly associated with good requirements. An appropriate methodology and a well-defined scope allow for well-defined deliverables.

While our practitioners suggested that P3, *the PM is able to choose the development methodology*, was important for a successful project outcome, our results did not support this. Some organizations forced PMs to use a specific life-cycle development methodology, irrespective of the problem. However, when PMs were given a choice, their projects were no more successful.

Change happens. Change is not a bad thing as it is virtually impossible to define all the requirements up front [47]; hence, managing requirements successfully includes effective change control. A change control process lets the project stakeholders make informed business decisions to provide the greatest customer and business value while controlling the project's lifecycle costs [47]. P7, *the project had effective change control,* was significantly associated with good requirements.

The quality of software project management is characterized by active risk management [18]. This observation is supported by the correlation between responses to questions P4, *risks were identified at the beginning of the project*, P5, *risks were incorporated into the project plan* and P6, *risks were controlled and managed by the PM*, and M2, *the PM was above average*. Even though risk management practices are significantly associated with good requirements [1], most developers and project managers perceive risk management activities as extra work and expense [19]. Glass suggests that risk management is the least practiced discipline within project management [20]. Our data clearly supports this view. Just identifying the risks without doing something about them is not enough. While 62% of projects had their risks identified, only 53% had the risks incorporated into the project plan. The number of projects that then had their risks controlled and managed by the PM dropped to 43%. Respondents indicated that 33% of projects had no risks, even though 62% of these projects did not have good requirements at any stage and 90% failed.

P10, *other projects impacted this project*, clearly a risk factor, was significantly negatively associated with good requirements and P6, *risks were controlled and managed by the PM.*

Using logistic regression with the responses to the "P" questions, the best predictor of good requirements was P6 (*risks were controlled and managed by the PM*, with P10 (*other projects negatively impacted this project*). This equation predicted 87% of projects with good requirements, 76% of projects without good requirements, and 82% correctly overall.

In summary: our results suggest that when risks are controlled and managed by the PM, we will get good requirements. However, there are risks outside the control of the PM such as other concurrent projects that can compete for scarce resources.

4. Discussion

We recognize some limitations of the study. The developers we surveyed mainly develop in-house software for their organization's use. As noted earlier, surveys are based on self-reported data which reflects people's perceptions, not what might have actually happened. Because we surveyed software developers our results are limited to their knowledge, attitudes, and beliefs regarding the projects and PMs with which they were involved. The dominance of small projects may have biased our results. However, the questions in our survey were based on discussions with practitioners who raised issues that they perceived as important to their day-to-day activities on real projects.

The best prediction equation for good requirements is C4 (*customer/users had a high level of confidence in the development team*), with R4 (*the project had a well-defined scope*), and P6 (*risks were controlled and managed by the PM*). This equation predicted 92% of projects with good requirements, 85% of projects without good requirements, and 88% of projects correctly overall.

Overall, R4, *the project had a well-defined scope*, is the most influential factor as alone it predicts 82% of projects with good requirements correctly. The addition of C4 *customer/users had a high level of confidence in the development team* raises prediction accuracy to 86%. A lack of risk management was the best predictor of projects without good requirements.

Our results support a hypothesis proposed by Davis and Zowghi [14] who suggest that good requirements practices are not sufficient for success. Our results show that practices beyond the scope of RE, such as the commitment of the sponsor throughout the project, the confidence of the customer/users in the development team, a skilled project manager and project processes that include risk management, not only lead to good requirements but ultimately to project success.

5. Conclusions and Further Research

To get good requirements we found that:

1) it is not having a sponsor, but having a sponsor whose commitment lasts throughout the project;
2) it is not the number of users involved that is important, but rather the size of the project in terms of functionality;
3) it is not the requirements methodology per se, but rather use of an appropriate software development methodology into which the requirements methodology fits;
4) it is not avoiding requirements creep, but rather having a well-defined scope when requirements creep;
5) it is not having a project manager with years' of experience, or a project manager experienced in the application area, but rather a project manager who manages requirements effectively;
6) it is not just the identification of project risks, but doing something about them, after they have been identified;
7) it is projects that have one, and only one, central repository for requirements.

Customer/user confidence in the development team, a well-defined scope and effective risk management are the best predictors of good requirements.

Table 1 shows that current practices are fair at best. Analysis of our survey suggests further research is required in order to investigate:

- The value of distinguishing more clearly between requirements scope versus project scope. Does a good definition of scope at the outset of a project enable project teams to better manage requirements that change or evolve over the course of a project?
- Customer involvement and customer confidence in the project team indicate better likelihood of success. How are these interrelated? Do customers become more involved because they are confident in the team, or are they confident because they are involved? What motivates customer involvement with the development team? What instills customers with confidence in the development team?
- How generalisable are the factors identified in this study? While we believe that the results of this initial study are significant on their own, we intend to compare against factors important for good requirements in other environments. This research serves as a starting point for motivating our continuing research into requirements practice in industry.

The major contribution of this study is to reinforce the importance of grounding RE research in practice.

References

[1] Addison T. and Vellabh S.(2002), "Controlling Software Project Risks- an Empirical Study of Methods Used by Experienced Project Managers" Proc. of SAICSIT, pp128-140 .

[2] Bandinetti S., Di Nitto E. and Fuggetta A., "Supporting Co- operation in the SPADE-1 Environment, IEEE TSE, Vol 22 No 12 pp 841-865.

[3] Boehm B. W. (1981), Software Engineering Economics, Prentice Hall, Englewood Cliffs, NJ.

[4] Boehm B. W. (1991), "Software Risk Management Principles and Practice", IEEE Software Vol. 8 No 1, pp32-41

[5] B. W. Boehm, V. R. Basili (2001), "Software Defect Reduction: Top 10 List", IEEE Computer, 34(1): 135-137.

[6] Booch, G., Rumbaugh, J., Jacobson (1999), The UML User Guide, Addison Wesley.

[7] Brooks F. P. Jr. (1975), The Mythical Man Month. Essays on Software Engineering, Addison Wesley, USA.

[8] Carroll, J. (ed.) (1995), Scenario-Based Design: Envisioning Work and Technology in System Development, Wiley Publications.

[9] Chatzoglou, P.D. (1997) "Factors affecting completion of the requirements capture stage of project with different characteristics", Information and Software Technology. 39(9): 627-640.

[10] Checkland, P. (1981), Systems Thinking, Systems Practice, Wiley Publications.

[11] Clavadetscher, C.(1998) "User Involvement: Key To Success", IEEE Software. Vol. 15, Issue 2 (Mar/Apr) p. 30, 32.

[12] CMMI http://www.sei.cmu.edu/cmmi/ accessed 5th July 2004

[13] Davis, A., and Hickey, A. (2002), "Requirements Researchers: Do We Practice What We Preach?", Requirements Engineering Journal, 7, pp107-111.

[14] Davis A and Zowghi D (2005) "Good requirements practices are neither necessary nor sufficient" Requirements Engineering Journal (to appear).

[15] DeMarco T. and T. Lister (2003), Waltzing With Bears, Dorset House Publishing New York NY.

[16] DeMarco T and Lister T., Peopleware: Productive Projects and Teams, (1987) Dorset House Publishing Co, NY.

[17] Gause D and Weinberg G, Exploring Requirements: Quality Before Design (1989), Dorset House, NY.

[18] Glass, R. L. (1998) "How Not To Prepare For A Consulting Assignment And Other Ugly Consultancy Truths", Communications of the ACM. Volume 41, Issue 12 (December) pp11-13.

[19] Glass R. L. (2001), "Frequently Forgotten Fundamental Facts about Software Engineering" IEEE Software, May/June, pp112, 110, 111.

[20] Glass R. L. (2002), "Project Retrospectives, and Why They Never Happen", IEEE Software, September/October, pp112, 111.

[21] Hall, T., Beecham, S., and Rainer, A. (2002), "Requirements problems in twelve companies: an empirical analysis" *IEE Proceedings Software*, 149 (5), pp153-160.

[22] Hull E., Jackson K., and Dick Jeremy (2002) "Requirements Engineering" Springer Verlag.

[23] Jackson, M. (2001), Problem Frames, Addison Wesley.

[24] Jones T.C. (1994), Assessment and Control of Software Risks, Prentice Hall, Englewood Cliffs NJ.

[25] Jones T.C. (1996)," Strategies for Managing Requirements Creep", IEEE Computer, Vol 26, No 6, pp. 92-94.

[26] Kovitz B., (1999), Practical Software Requirements, Manning

[27] Leffingwell, D. and D. Widrig (2000) Managing Software Requirements: A Unified Approach, Addison-Wesley.

[28] McConnell, S. (1996), Rapid Development, Redmond, Washington: Microsoft Press.

[29] McKeen, J.D. and T. Guimares (1997) Successful strategies for user participation in systems development, Journal of Management Information Systems. 14(2): 133-150.

[30] Moynihan T. (1997), "How experienced Project Managers Assess Risk", IEEE Software, Vol 14, (MAY/JUNE) pp35-41.

[31] Neill C. J. and Laplante P. A. (2003), Requirements Engineering: State of the Practice" IEEE Software, November/December, pp 40-45.

[32] Osmundson J. S., Michael J. B., Machniak, M. J., and Grossman M. A. (2003), "Quality management metrics for software development" accepted for Information and Management.

[33] Paulk, M., B. Curtis, M. Chrissis, C. Webster (1993) Capability Maturity Model for Software, Technical Report, CMU/SEI-93-TR-024, Software Engineering Institute, Carnegie Mellon.

[34] Pressman, R. (2000), Software Engineering: A Practitioners Approach, adapted by Darryl Ince, 5th edition, McGraw Hill, London.

[35] Procaccino, J.D. and Verner J. M., (2002) Software practitioner's perception of project success: a pilot study, International Journal of Computers, The Internet & Management. 10(1): 20-30.

[36] Procaccino, J.D., Verner, J. M., Overmyer and M. Darter (2002) Case study: factors for early prediction of software development success, Journal of Information & Software Technology. 44: 53-62.

[37] Robertson J. and Robertson S. (2000), "Requirements Management: A Cinderella Story", Requirements Engineering Journal, 5 (2), pp134-136.

[38] Roy V., and Aubert B. (2003), "A Dream Project Turns Nightmare: How Flawless Software Never Got Implemented", Annals of cases on Information technology Idea Group Publishing pp 98-111

[39] Schmidt, R., Lyytinen, K., Keil, M., and Cule, P., (2001) "Identifying Software Project Risks: An International Delphi Study", Journal of Management Information Systems, Spring, Vol 17, No 4, pp 5-36.

[40] Standish Group (1999), "Chaos: A Recipe for Success", Standish Group International.

[41] Sumner, M. (1999) Critical success factors in enterprise wide information management systems projects, SIGCPR Conference, New Orleans, LA. 297-303.

[42] Tackett, B. and van Doren B., (1999), "Process Control for Error-Free Software: A Software Success Story", IEEE Software 16(3): 24-29.

[43] Verner J. M. and Evanco W. M. (2003), "A Investigation into Software Process Knowledge" in Managing Software Engineering Knowledge, (eds) A. Aurum, R. Jeffery, C. Wohlin and M. Handzic, Springer-Verlag, pp29-47.

[44] Verner J. M. and Evanco W. M. (2005), "In–house Software Development: What Software Project Management Practices Lead to Success?" IEEE Software Jan/Feb vol. 22, issue 1, pp86-93.

[45] Verner J. M, Cox. K. Bleistein S and Cerpa N. (2004) "Requirements Engineering and Software Project Success: An Industrial Survey in Australia and the U.S. Proceedings of AWRE, Adelaide Australia, Dec 2004.

[46] Verner, J. M., Overmyer, S. P. and McCain, K.W. (1999) In the 25 years since the mythical man-month what have we learned about project management?, Information and Software Technology. 41(14): 1021-1026.

[47] Wiegers, K. (2003), Software Requirements, Second Edition Microsoft Press, Redmond WA.

[48] Weiringa, R. and Ebert, C. (2004), "RE'03: practical Requirements Engineering Solutions" IEEE Software March/April, pp. 16-18.

A risk based guide to business process design in inter-organizational business collaboration

Michael Schmitt, Bertrand Grégoire and Eric Dubois
Public Research Centre Henri Tudor

Abstract—**The same business model may be implemented using a variety of business process alternatives. This position paper investigates a risk-based approach to explain the choice of the business process for a given business model. Risk goals and risk mitigation instruments are discussed as a major factor in the alignment between business model and operational implementation. A requirements approach based on i* is proposed for capturing the needs and constraints governing this alignment.**

Index Terms— **business models, value web, risk goals, risk mitigation instruments, business process patterns**

I. INTRODUCTION

In its traditional view, requirements engineering (RE) offers a set of methods, techniques and tools for reasoning on the 'why' behind the introduction of an information (software-based) system within an organization. This 'why' is expressed in terms of the different expectations of different stakeholders, with respect to the future system. Requirements capture non-functional constraints (like, e.g., security, performance, usability, etc) regarding the implementation of the system and functional aspects, which guarantee that the system is aligned with the operational goals of the organization, i.e. its business process model. In short, RE defines the set of properties expected from the information system and thereby restricts the number of possible alternatives regarding its development and implementation.

The design of the business process model (BPM) itself is not an easy task since, given a particular situation, alternative BPM are conceivable. In this paper we propose to extend the use of RE techniques to the modelling and the understanding of the 'why' *behind the design of a BPM*. In particular we will show how the i* requirements framework, and goal-oriented techniques generally speaking, can help in reasoning on the business goals of an organization and on the different constraints relevant to the identified BPM.

More specifically, we will associate business goals with a high-level business model (BM), with a value exchange perspective and constraints, and with an analysis of risks associated with the different BPM alternatives.

In section II, we will better characterize the differences between a BM and a BPM. Section 0 will then identify the risks guiding design decisions during the transformation a BM into a BPM. Finally, before Conclusion, section IV will discuss the possibility of identifying business process patterns associated with risk mitigation instruments, without enforcing any representation of business processes, not to overload this article.

All along the paper the concepts will be illustrated through the handling of a B2B case study regarding distribution of electronic parts.

II. BUSINESS MODEL AND PROCESS MODEL

A. Business Model

Business models (BM) explain the nature of a business case, that is, they characterize who is responsible for which part of the value creation, who brings in which capabilities and resources and what he expects in return. They provide a high-level view on what will be offered to the customer, which business partners, resources, capabilities and activities the value creation will be based upon. At the core of a BM are the value exchanges between the business partners involved. Value exchanges are by definition reciprocal so that every partner brings in something into the common business activities as well as he benefits from his participation [1]. BMs help to reach a shared understanding among stakeholders about the core of the business and to align everybody's objectives towards the common business goals.

Fig. 1 depicts a simplified BM that shows only the value exchanges between the business partners. We refer to such a type of business model as a *value web* in order to draw the borderline between a company centric and comprehensive business model, as described by Osterwalder in [14] and a value exchange perspective on a network of business partners. It represents the roles and flows of value objects of a common supply chain transaction [2], where a buyer purchases electronic components from a seller, showing. The value proposition offered to the buyer comprises on top of the sales of the electronic components (parts) additional services such as transportation and stocking in the proximity of the buyer. This permits the buyer to quickly adapt to changing production needs and reduces the lead-time of the components. In order to be able to deliver within a short lead-

Michael Schmitt, Bertrand Grégoire and Eric Dubois from the Public Research Centre Henri Tudor (http://www.citi.tudor.lu), 29 Avenue John F. Kennedy, L-1855 Luxembourg. Phone: +352 42 59 911, mail: {michael.schmitt, bertrand.gregoire, eric.dubois} @tudor.lu.

This work has been supported by FNR Grant 01/01/07, as part of the *Efficient toolset* for the design and validation of e-business transactions (http://efficient.citi.tudor.lu)

time and to ensure a maximal flexibility, the seller contracts a third-party warehouse for the stocking services required. The buyer considers the warehouse as being part of the value chain provided by the seller; there is no contractual relationship between the warehouse and the buyer.

The entire communication and object flows that are necessary to perform the value chain activities (order handling, inventory management, goods delivery, financial settlement) do pass between buyer and seller and between the seller and the warehouse. No communication takes place between the buyer and the warehouse.

In our business case we assume that the seller and buyer plan a long-term relationship with regular delivery and replenishment cycles. Both parties fully trust each other. The profit margin associated with the sales activity does hardly leave a navigation range for risk mitigation. The long-term and regular nature of the business scenario allows both business parties to spread the operational risks to many value exchanges and hence to minimize their respective risks exposure.

Note that this BM depicts an ideal world in that it does not detail the way in which the two business partners coordinate their respective value creation activities; So for example it does not incorporate the fact that a business partner may not be able or not willing to fulfil his contractual obligations [13].

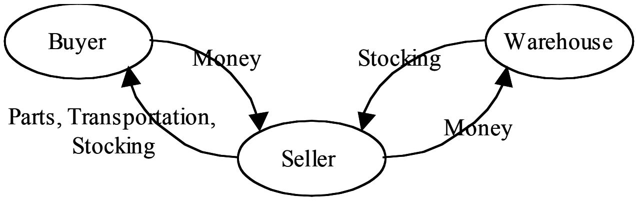


Fig. 1: simple distribution model of electronic parts

This ideal nature of the value web makes that there can be many transactions, or business processes that implement the same business model. A value web depicts the exchange of *value objects* (tangible or intangible) in such a way that each actor involved brings and receives objects of value and profits from his participation in the value web [1].

B. Business Process Model

As compared to the value web, the business *process* model (BPM), which implements the value web on an operational level, adds to the value web

- an order onto the exchanges of value objects
- additional supportive information flows that facilitate the coordination and communication the business partners, but that do not exchange objects of value. An example is the confirmation of the receipt of an object of the announcement that a value object will be sent in a specified time period.

For our sample distribution case, there are many BPM alternatives such as the following [4]:

- Vendor managed inventory process
- Consignment inventory management
- Seller contracted 3rd Party warehouse in a consignment scenario (see [4])
- Min-Max processes
- Kanban
- Breadman model

In the following paragraphs we discuss the factors that impact on the selection of a business process scenario for a given value web, and in the next section we will show how an appropriate process model is chosen.

C. From Business Model to Process Model

What makes a network of trading partners decide which process model suits best their business needs? Which ordering constraints hold for the value exchanges of a BM, and how can supportive information flows be added to the value exchanges in order to facilitate the coordination of value creation activities between the business partners?

The answer to these questions forms a decision support process that aims at creating a fit between the process model and the value web it implements.

There are two types of *ordering constraints* for value exchanges:

- Compulsory or hard constraints are such that the business model would not make sense without them or such that are imposed. Examples are flow constraints [5] where a value object that results as an output from one value transfer is an necessary input for another value transfer.
- Soft constraints are such that they are negotiable in accordance to the business partner's preferences and needs. Whether or not the delivery of electronic components follows or precedes its payment does not change the business case as such. However, both parties the seller and the buyer may have a clear preference as regards the sequence of value exchanges.

At the level of an individual value exchange, we argue that there are two main characteristics that determine the way the actual value transfers is organized:

- The subjective *perception of the risk* (probability, impact) associated with the value exchange. Both the buyer and the seller in our business case will assess whether or not they consider it risky to fulfil their part of the contractual obligations before receiving the other one's contribution, that is, to perform the payment before receiving the goods and vice versa. One parameter that impacts on the assessment of risk is the level of trust between the business partners. However, other characteristics may be taken into account such as:
  - o risk rating for the country where the buyer resides.
  - o probability and impact of currency fluctuation between the receipt of a purchase order and the payment in case that the payment currency differs from the home currency of the seller.
  - o difficulties related to the transport and customs declaration.
  - o need of the seller to pre-finance the goods production, and hence the capital commitment involved.
- The *navigation range* defined as the difference between the achievable profit level and the cost of the application of risk mitigation instruments to secure the exchange. That is to say

that the cost of risk response activities do not outweigh the achievable profit margin.

Some of the requirements and goals of the actors involved do apply to the characteristics of the value transfers; other goals can be derived from the business context and the objectives of the common value creation. We propose to draw a map of these goals using the classical requirements engineering techniques of goal engineering, as proposed by [15]. An application of such an approach to our sample business case is depicted in .

The buyer wants to satisfy his demand, which can be decomposed into its offerings such as 7x24 delivery and short lead times. The seller addresses the buyer's demand by offering him a particular value proposition that meets both his own requirements (such as being profitable for him) and the requirements associated with the buyer's demand. The seller himself wants to have as much control on the value chain as possible and wants to minimize the risk of non-payment by the buyer.

Besides, the fact that the buyer requires a 7x24 delivery capability and flexible lead-times requires the seller to purchase stocking services from a third party. Because of the long-term nature of the relationship with the buyer (with repetitive orders signalling a continuous demand), the Seller is willing to secure his supply, engaging in a trusted commercial partnership with that third party offering stocking services.

III. A RISK ANALYSIS FOR ECONOMIC VALUE TRANSFER

A. Typology of risks

When considering the risks associated with a value transfer and the corresponding risk mitigation instruments, we need to take into account the type of the value object transferred:

- Financial flows include money, shares, and payment obligations.
- Human resource flows refer to consultancy and body leasing models where one partner misses a specific expertise or wants to transfer an operational risk to an external party. Usually these flows are of a time-limited nature.
- Information flows refer to the exchange of valuable information between business partners.
- Physical object flows involve such activities as assembly, packing, transportation and storage.

We shall concentrate in this article on financial and physical object flows only, however, we believe that other types of flows do follow the same principles, and that we are able to identify a finite number of such flows (and bonded risks) for some practical domains. The supply chain management has already been thoroughly investigated, with variants in the e-Commerce field, as in [3] or [4], for instance.

The following is a list of some typical risks that occur in financial and physical object flows:

1) Risks associated with physical value flows

The main risks associated with the transfer of physical goods are [6] the following:

- The risk of loss or damage of the goods.
- The risk of delay of the delivery.
- The risk of non-acceptance by the customer due to non-conformance with his requirements or due to poor product quality.
- The risk of a liquidity squeeze due to the capital commitment, especially with long payment terms.

The business partners involved in the value exchange need to come to an agreement as to who bears the risks above and who will be responsible for any resulting costs. Note that the sales price for the goods changes depending on who will take which risk in the value transfer.

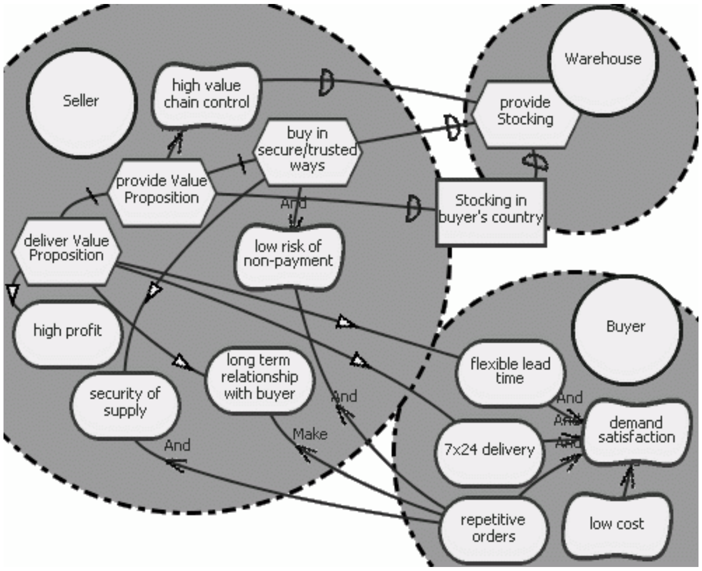


Fig 2: Goals map, covering value exchanges characteristics and business context

2) Risks associated with financial value transfers [6]

As for the financial flows, the following risks may occur:

- Non-payment or partial payment by the buyer as well as payment delays.
- Currency fluctuation, for international flows, especially when the agreed payment terms allow for a late payment.
- Risk (and costs) associated with appeal and collection activities.

B. Risk goals impact on the order of value transfers

In an ideal case, a value transfer is characterized by a bi-directional exchange of objects of economic value between two actors. In a real world business case, a value transfer may not always correspond to this description: a business partner may refuse the object of value; he or she may refuse to pay or pay late. In order to mitigate such and other risks when implementing a value transfer in a real-word scenario, two basic economic situations are thinkable:

- One of the two parties has a strong market position as compared to the other and may hence impose the terms of exchange. The same holds true for non-competitive markets or markets with a high degree of government regulation.
- Both parties are free to negotiate terms and conditions.

In the first case the order of value transfers between the business partners may be imposed by one of the partners, whereas the other partner must accept this, or otherwise decide not engage in the commercial activity. In the second case, each of the business parties may formulate his or her own goals for the order of the value transfers and will try to negotiate for his preferences with the other party.

The goal-setting process involves an assessment of the risks involved in terms of their probability and impact. If a risk related with a value transfer is considered as easily manageable or as minor important, the order of the exchanges does not matter; after all risk management activities impact on effort, time and cost and hence are likely to reduce the profit margin. This is especially the case if

- The counter party is well known, or both parties do trust each other.
- The business relationship is of a long-term nature or if the risks can be spread on several contracts so that the overall risk exposure becomes much lower as compared to the risk associated with a single business contract.
- The impact in the event of risk is relatively low and financial coverage is given.
- The probability of the risk event is considered low or unlikely.

If the risk is considered as severe, both partners may want to negotiate with each other for the terms and conditions of the value transfer, to cover their own risks. For such a situation, the market offers a variety of risk mitigation instruments that bear the advantage that they provide legal security to the business parties involved, in that they define the rights and obligations as well as standardized choices for the terms and conditions of the exchange. The following is a list of some risk mitigation instruments for the risks associated with financial value transfers:

- *Payment in advance*. This option excludes all financial risks on the seller side but leaves the buyer with the risks related to the quality and transportation of the physical goods. It requires a high level of trust from the buyer and can only be imposed by the seller where his negotiation power is strong. Obviously the safest method for the exporter, this is generally unavailable in competitive markets. A partial down payment (20-30%) may be more acceptable to the buyer and therefore be more realistic, but leaves the seller exposed to a risk on the balance.
- *Payment on open account.* The seller delivers the goods to the buyer together with an invoice. This is the least safe method from the seller's perspective and is only used when the buyer is fully trusted and creditworthy. This option does not address the risk of currency fluctuation.
- *Payment in local currency*. The seller issues an invoice in his home currency so that the risk of a currency fluctuation is eliminated.
- *Down payment*. The buyer pays a percentage of the order value in advance and the remaining sum when he receives the goods.

For the transfer of physical goods, typical market instruments for risk mitigation are the following:

- *Fixing of a transport insurance and agreement on the place of the transfer of title*. Depending on the INCOTERM [7] (and hence the price agreement for the financial compensation) agreed upon for the value transfer, the transfer of title for the goods may occur at the seller's premises or the transfer may cover for transportation, customs and delivery to the final destination.
- *Factoring, forfeiting*. The seller excludes the risk of a liquidity squeeze by selling his receivables from the goods delivery to a third party who takes over all financial risks associated.

A market instrument that covers both, the risks of the financial and those of the physical flow of goods, but which is relatively expensive is the

- *Documentary credit*, also known as letter of credit.

Note that most of the risk mitigation instruments discussed require additional business partners (trusted parties or intermediaries) to be added to the business model in order to facilitate the value transfer and to manage the risks involved. What is similar in all cases is that risk management introduces an additional, value creating business activity and hence introduces additional effort and cost that reduces the profit margin for all of the business parties. Therefore, each company or individual involved in a business model will need to balance his risk management goals with those associated with his target profit margin and any other business goal that may be of relevance for the business context. If for instance a company has enough financial coverage to meet a financial risk and if the market it wants to penetrate is highly competitive, it may decide to offer its products at the lowest price possible and therefore not to respond to its financial risks.

C. Risk mitigation instruments help to achieve goals

We may easily identify the appropriateness of the risk mitigation instruments available with regards to the goals expressed by the participating actors, shown in the goal map of section II.C. For example, the risk mitigation instrument of a *payment on open account* can be excluded from the list of suitable instruments because the seller wants to secure the risk of non-payment and has no trust in the buyer, as illustrated in .

A *Letter of Credit*, which is relatively complex from an administrative point of view and which adds one or two intermediaries to the value chain, would hurt the goal of achieving a high profit margin, which presupposes a low cost infrastructure; moreover, it seems not suited for a business of a repetitive nature, as the costly and time-consuming settlement process would need to be carried out frequently. Therefore, a *Payment in advance* mechanism seems to be the best choice for the business scenario and the goals of the actors involved.

In the next section, we shall look at how risk management goals may impacts on the business process implementation of a business model.

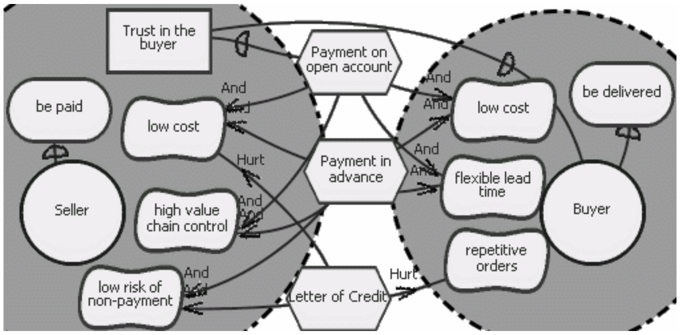


Fig. 3: Goals achieved by risk mitigation instruments

IV. Risk mitigation instruments, process patterns and complete business processes.

Risk mitigation instruments describe alternative ways of achieving risk related objectives. Let us consider the case of the *shipment of goods* in our sample distribution model (see Fig. 1). This shipment might either be handled by the seller, assuming that he's got the logistics capability and the resources needed, or it may be effected by the warehouse, under the same conditions, or even by a third-party carrier contracted by either of the two parties, against financial compensation. Fig. 4 depicts the various flows of goods possible for these alternatives.

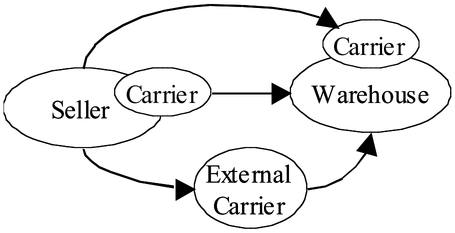


Fig. 4: Risk mitigation instruments for the shipment from the Seller to the Warehouse

Each of the alternative flows bears a different risk level for the business actors involved. When the seller is at the same the carrier he would assume the full responsibility for the goods in delivery, which includes the risk of loss or damage as well as the risk of delayed arrival. If he doesn't want to take this responsibility himself or when he perceives the risks as high, he can either choose a different scenario or he could secure (mitigate, transfer) a part or all of the risks associated, for instance by purchasing appropriate insurance.

Let us assume that the actors agree that the seller takes on responsibility for the shipment of goods from his own premises to the warehouse. From this business choice, we may deduce the flow of information that corresponds to that choice in a standardized manner, depicted by the dashed arrows in Fig. 5. What is interesting to note is that for the various scenarios, the market makes available standardised business processes and transactions involving automated data interchange and standard message types such as the UN/CEFACT Shipment advice message from the seller to the warehouse (DESADV).

It turns out that each of the different risk mitigation instruments corresponds to a fairly standardized business process pattern describing the flow, the content and the business rules governing the information exchange between the business partners involved.

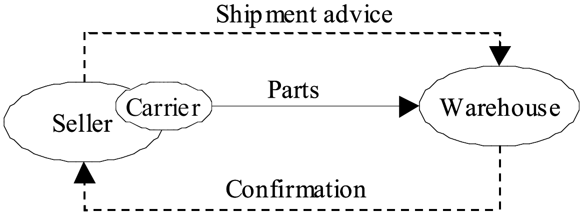


Fig. 5: information flow for one alternative

The description of business process patterns, at this point, is detailed enough to be implemented by classical business process theories and tools (as explained by [1]). There are several frameworks available for the design and the validation of multi-partner business processes, as for example the EFFCIENT toolset [11]. Based upon a shared understanding of the core of the business model, and a description of the roles and responsibilities of each of the actors involved, the EFFICIENT toolset allows the business experts to design and to agree upon an actual business process that implements the value exchanges of the value web. The validation is supported in a distributed simulation environment.

A. The Combination of market risk patterns in a complete business process

The choice of business process patterns (risk mitigation instruments) for a value web does not only depend on the risk goals. There are various interrelationships between the use of risk instruments and the overall goals of the actors concerned. If, for instance, the business maxim for an actor in a business model is to maximize time efficiency, the selection of a time-costly risk instrument in one part of a transaction may prevent the choice of other time-consuming risk mitigation instruments in other parts (value object exchanges) of the same transaction.

Therefore, if we map the risk mitigation instruments available with their respective characteristics and impacts onto the business goals of the business model, we may improve the alignment of business model and process model. This, we think, will allow us to use theories and tools that exist in both the domains of requirements engineering techniques [15] and optimisation [12] to guide the business actors of a value web in their choice of one right process model that fits their business needs.

B. Dynamic creation of new patterns

In the context of a commercial transaction, the main steps involved in a value object transfer are well understood [7] [8]. Also, the various alternative business process patterns that exist for a value web are rather standardised [7] [8]. We are therefore confident that we shall be able to characterise and describe most of these alternative patterns in a reusable way.

However, where there are no risk mitigation instruments available or where there is no standard pattern that fits the business goals of the actors involved, , we should be able to define new process patterns. Each pattern must be described and characterized with regards to the risk management goals associated (as explained in section 0), and its characteristics must be mapped onto the pattern description parameters in order for it to be re-usable in similar business situations (see Appendix 1 for more details).

C. Methodology for defining new patterns is required

There is some research on the combining of basic transaction patterns into more complex sequences in accordance with the business objectives. Among those, we wish to highlight the UMM methodology [10] of UN/CEFACT, that proposes 6 binary communication schemes (called *business transaction patterns*) that can be combined to build up more complex transaction patterns (called *choreography*) involving more than two actors.

As described in Appendix 2, each of these schemes has inherent characteristics that mainly concern the risks associated with the flow of information between two business partners.

The UMM suggests a methodology to break down and detail step by step a high-level, managerial view of a business transaction (called *business operation* map) into a fine-grained, operational sequence of process patterns. As such, UMM might be employed to help business experts define new patterns, starting with a description of their business needs and refining each their needs on a process level.

Another research framework that involves the use of patterns for the description of business transactions is proposed by Jayaweera in [9]. In his BP3 framework, he proposes a method to decompose the description of a transaction, considered as a speech act, into its core activities. The application of Speech-Act based theory considers 13 atomic speech actions (called *pragmatic actions*) that fall into 5 different types (or purposes, illocutionary forces

Using the BP3 framework, any business discourse, considered as the human activity of the conception of a business transaction, can be decomposed into a combination of pragmatic actions. A semi-automated analysis of the relationships between these actions, for instance in terms of their sequence or resource dependencies, may allow us to derive a partial order of the exchanges in a first step. A final effort to assign responsibilities to each of the may then help us to obtain a complete and executable business process, that corresponds to and formally implements the linguistic description of the business discourse.

Finally, another way of defining new patterns would consist in the adaptation of existing patterns. Possible adaptations include the addition of the removal of trust mechanisms (including acknowledgment, or repudiation) as well as the addition of further actors as trusted intermediaries or for transferring a part of the value creation to an external party such as an insurance company that one may want to transfer part of the risks to. The mechanisms for reuse need to be further investigated in order to ensure an easy and integrated evolution of the library of patterns.

Each new pattern defined will need to be described and aligned to the classification grid for risk management, as summarised in Appendix 1.

V. Further work and conclusion

This position paper introduced a means of designing a business process that matches the business goals of its underlying business model by detailing the risks inherent to a business value proposition, and identifying process pieces (instrument) that would satisfy to those goals.

Further work includes the design of a complete classification framework of business risks and the associated risk mitigation instruments to guide business experts in selecting the right instrument for their business goals and requirements. Also, we shall investigate on possible implementation languages to represent the goals of the business actors in a value web and to design a BPM that is aligned with the corresponding value web.

References

[1] Gordijn, Akkermans and van Vliet, *Business Modelling is not Process Modelling*. in: Conceptual Modeling for E-Business and the Web, ECOMO 2000.

[2] Porter, *Competitive Advantage*. New York, Free Press, 1985.

[3] Weill and Vitale, *Place to Space: Migrating to Ebusiness Models*, Harvard Business School Press, 2001.

[4] CompTIA clickable business models, Online guide. Available: http://eidx.comptia.org/business_process/bm_clickable.aspx

[5] Bergholtz, Jayaweera, Johannesson, Wohed, A pattern and dependency based approach to the design of process models, International Conference on Conceptial Modelling, ER 2004, China.

[6] Altmann, Aussenwirtschaft für Unternehmen, 2001, Lucius & Lucius, Stuttgart.

[7] Chambre de commerce international, Incoterms 2000, Paris.

[8] Chambre de commerce international, Guide to Export Import Basics, ISBN 92.842.1309.6

[9] Prasad M. Jayaweera, A Unified Framework for e-Commerce Systems Development: Business Process Pattern Perspective

[10] UN/CEFACT Modeling Methodology (UMM-N090 Revision 10), Valid on 20040419, http://webster.disa.org/cefact-groups/tmg/doc_bpwg.html

[11] Amel Mammar, Sophie Ramel, Bertrand Grégoire, Michael Schmitt, Nicolas Guelfi. Efficient: A Toolset for Building Trusted B2B Transactions. Accepted for 17th Conference on Advanced Information Systems Engineering (CAISE'05), Porto, June 2005.

[12] Handbook on Constraint Programming, Mark Wallace, September 1995

[13] R.J. Wieringa and J. Gordijn, “Value-Oriented Design of Service Coordination Processes: Correctness and Trust”, Accepted by the ACM Symposium on Applied Computing, Organizational Engineering Track, 2005

[14] Osterwalder, Business Model Ontology, PhD Thesis 2004, HEC Lausanne.

[15] E. Yu, Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering' Proceedings of the 3rd IEEE Int. Symp. on Requirements Engineering (RE'97) Jan. 6-8, 1997, Washington D.C., USA. pp. 226-235.

Appendix 1 PATTERN DESCRIPTION FRAMEWORK

1. *Miscellaneous information*: pattern name, author, version

2. *Handled risks*: some risks the pattern deal with, ordered by the nature of its underlying flow(s). Those risk are chosen from the list below that is not exhaustive, and must be attributed to a particular role of the pattern. A risk is associated with one actor of the pattern.

    a. physical value flows [4]
    i. loss or damage of the good.
    ii. delay with the delivery.
    iii. non-acceptance by the consumer.
    iv. liquidity squeeze.

    b. financial value flows [4]
    i. non-payment or partial payment.
    ii. payment delays.
    iii. currency fluctuation.
    iv. risk (and costs) associated with appeal and collection activities.

    c. human resource flows
    i. competency
    ii. quality of work
    iii. information flows
    iv. privacy.
    v. confidentiality.
    vi. Repudiation of origin or content.

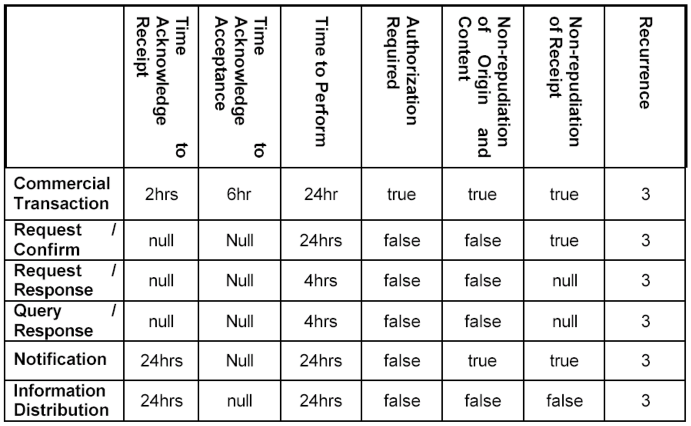
3. *Characteristics*: Features of the pattern that are not directly bound with one of the former risks.
    i) complexity, required capabilities and know-how.
    ii) cost
    iii) time
    iv) level of confidence required in each partner
    v) efficiency
    vi) flexibility

Appendix 2 UMM BUSINESS TRANSACTION PATTERNS

The 6 (atomic) business transaction patterns of UMM [8] are:

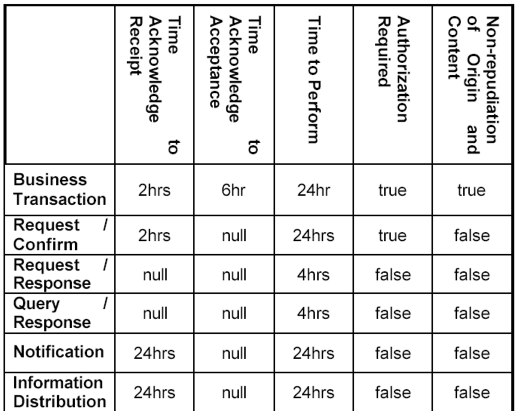
- Commercial Transaction
- Request / Confirm
- Query / Response
- Request / Response
- Notification
- Information Distribution

These patterns have different characteristics regarding the requesting role, described in Fig 6, and the responding one, as detailed in Fig 7.



	Time Acknowledge to Receipt	Time Acknowledge to Acceptance	Time to Perform	Authorization Required	Non-repudiation of Origin and Content	Non-repudiation of Receipt	Recurrence
Commercial Transaction	2hrs	6hr	24hr	true	true	true	3
Request / Confirm	null	Null	24hrs	false	false	true	3
Request / Response	null	Null	4hrs	false	false	null	3
Query / Response	null	Null	4hrs	false	false	null	3
Notification	24hrs	Null	24hrs	false	true	true	3
Information Distribution	24hrs	null	24hrs	false	false	false	3

Fig 6: Pattern characteristics with regards to the requesting role



	Time Acknowledge to Receipt	Time Acknowledge to Acceptance	Time to Perform	Authorization Required	Non-repudiation of Origin and Content
Business Transaction	2hrs	6hr	24hr	true	true
Request / Confirm	2hrs	null	24hrs	true	false
Request / Response	null	null	4hrs	false	false
Query / Response	null	null	4hrs	false	false
Notification	24hrs	null	24hrs	false	false
Information Distribution	24hrs	null	24hrs	false	false

Fig 7: Pattern characteristics with regards to the responding role

Goal-oriented RE for Handling Change Requirements: An Explanation of What Stakeholders Try to Avoid and What They Try to Achieve

Johan F. Hoorn
Vrije Universiteit
De Boelelaan 1081a
1081 HV
Amsterdam, NL
T: +31 20 598 7614
F: +31 20 598 7728
jfhoorn@cs.vu.nl

Elly A. Konijn
Vrije Universiteit
De Boelelaan 1081
1081 HV
Amsterdam, NL
ea.konijn@fsw.vu.nl

Hans van Vliet
Vrije Universiteit
De Boelelaan 1081a
1081 HV
Amsterdam, NL
hans@cs.vu.nl

Gerrit van der Veer
Vrije Universiteit
De Boelelaan 1081a
1081 HV
Amsterdam, NL
gerrit@cs.vu.nl

ABSTRACT

One of the reasons why requirements engineering (RE) is so difficult is that requirements change 'on the fly.' To investigate the sources of requirements change, 18 managerial supervisors of a logistic warehouse management system filled out a structured requirements-engineering questionnaire, the RE*quest*, which assessed the level of agreement to the current system, the future system, and the stakeholders' needs. The results confirmed the assumption in goal-oriented RE that requirements are tightly connected to goals. More importantly, however, we discovered a mechanism that rules the level of agreement to requirements, which we coin the *goal-to-requirements chiasm* or the *χ-effect*: Variance in what the system won't have is for 70% explained by goals stakeholders want to *achieve* with the system. Variance in what the system must have is for 90% explained by goal states that stakeholders want to *avoid*. Moreover, we found evidence for an emotional component (i.e. valence) in the requirements evaluation that has a moderating effect on agreement to requirements. The *χ-effect* emphasizes that won't requirements and goals to avoid are as important to requirements change as must requirements and goals to achieve with the system. In this light, structured questionnaire design is a systematic and controllable addition to common requirements-validation methods.

Categories & Subject Descriptors

H.1.2 [Models and Principles]: *User/Machine Systems–Human information processing*; K.6.3 [Management of Computing and Information Systems]: *Software Management–Software development*.

General Terms

Requirements Engineering, Human Factors, Theory.

Keywords

Requirements validation, requirements change, empirical software engineering, stakeholders' view, structured questionnaire.

1. INTRODUCTION

A major problem in developing a system is to know what functionality a system should offer, what goals it should support or what business processes it should facilitate. Requirements engineering (RE) is a series of organized activities to obtain and document such knowledge for system engineers as well as for other stakeholders who are involved in developing or using the system (e.g., the client, managers, end-users, and maintenance personnel).

The problem gets worse when stakeholders change their minds about what they want from the system. Particularly when a system is under development, a change request can have serious impact on the design of a system (cf. [11]). Moreover, the business situation sometimes changes so quickly that change requests repeatedly occur during the course of development. Redesign, however, is expensive, time-consuming, and often frustrating.

It is therefore important that we can anticipate a change request. If we know the sources of requirements change and the mechanisms that govern a change request, it might be possible to detect 'the danger zones' – the requirements most susceptible to change – in the early stages of requirements elicitation and gathering.

However, we are dealing with rapid changes. Therefore, we not only need to know which requirements on a specific system in a specific business case are changing and why, we also need generic knowledge on requirements change. With this type of knowledge, we – hopefully – can anticipate change requests while being less dependent of the particular system under construction and less vulnerable to the time aspect.

2. THEORY

2.1 The Type of Goals

In the area of goal-oriented RE (e.g., [26]), the cause of requirements change, requirements evolution [1], or requirements development [33] is sought in the goals that stakeholders want to achieve with the system or the concerns they may have with it. "Goals are ... essential elements for managing requirements evolution" [27]. Goals can range from high-level strategic mission statements to low-level operational targets that should be achieved with the system [27]. Goals are supposed to be more stable than the requirements that help reaching them [25]. Moreover, the higher-level a goal is (e.g., a strategic business goal), the more stable the respective requirements will be [3] [1]. Thus, the reasons for requirements change should be sought in a change of lower level goals, such as improving a work process (e.g., higher efficiency, less costs), or advancing system performance, security, and reliability.

2.2 Valence

When stakeholders are involved in developing a system, they are – whether intentionally or not – also busy designing the future situation of their business or work environment. Therefore, they make evaluations of how much a requirement, once implemented as a feature of the system, will impact their goals.

In goal-driven RE, system development is centered on the stakeholders' concerns [21] [4] [2]. In the line of Frijda [16], we think that the requirements on the new system are judged for their usefulness or relevance to potentially satisfy or harm the stakeholder's concerns, goals, or motives. Positive expectations about the future situation result from requirements that promise a match, the actual or expected satisfaction of concerns. Negative expectations result from requirements that promise a mismatch, the actual or expected obstruction of realization of goals and concerns ([16] p. 277). Frijda ([16] p. 207) points out that valence refers to the implied outcome of the event: The intrinsic attractiveness or repulsiveness. In other words, valence (also [36]) refers to the expected match or mismatch between the potential gratification for or obstruction of stakeholder concerns and the possibilities or impossibilities offered by the new situation.

Stakeholders expect positive or negative consequences of the system for achieving their goals (cf. Technology Acceptance Model [13]). Whether stakeholders expect that a proposed feature will support or obstruct their goals may have an impact on the level of agreement or disagreement to a requirement. When the business environment changes, the direction of valence towards the future system may change accordingly, thus triggering a change request.

2.3 Not Only Must Haves

Although practitioners often work from a MuSCoW list,[1] the won't requirements are often put aside as irrelevant for further analysis. The focus is on the must haves, understandably, to help achieve the stakeholders' goals. However, whereas goals specify desired situations, so called "obstacles" designate goal states that are undesirable but yet possible [30] [25]. Apart from achieving goals, there is also an "avoid-mode" [33]. Thus, must haves may be important to achieve goals stakeholders want to approach, yet, won't haves are important to construe what stakeholders want to avoid with the system (e.g., instability, complexity). When a business model changes, a change request can concern the won't requirements just as well as the must requirements.

2.4 Variability in Agreement

When business goals change and the requirements change accordingly, the once agreed-upon requirements are disagreed-upon in the new situation. If we know which goals have changed it should be possible to predict the level of agreement to the related requirements from the level of agreement to the (changed) goals. We suspected that requirements that raise the most conflicts among stakeholders are most vulnerable to change. Such requirements should show more variability in the level of agreement (from agree to disagree) than requirements that raise no conflicts (a ceiling effect of either agree or disagree). Thus, we wished to investigate which type of goals (those to approach or those to avoid) best predicted the variability in the level of agreement to must or won't requirements. Our best guess was that (H1) goals to approach would predict agreement to the must requirements through the mediation of positive outcome expectancies (valence support). In opposition, we assumed (H2) that goals to avoid would predict (dis)agreement to won't requirements, mediated through negative valence (valence obstruct).

The remainder of this paper is organized as follows. Section 3 describes the methods and tools we employed, i.e. the Requirements Engineering questionnaire RE*quest*, to gather the data for testing our hypotheses. Section 4 supplies the necessary statistical analyses and empirical results, which are discussed in Section 5. In Section 6, we relate our findings to some prominent studies in the goal-driven RE domain. Section 7 rounds off our paper by offering an outlook on future explorations.

3. METHOD

3.1 Participants

Managers (N= 18; 11 male, 7 female; age M= 46.4, SD= 10.9; years in service M= 14.4, SD= 11.7) from a provincial governmental institution in The Netherlands participated in a questionnaire study that concerned the (re)design of a logistic warehouse management system. These participants ranged from various services, sectors, and functions within the organization.

3.2 System

The state of the warehouse management system at the time of measurement was a mainly manually and personally driven order and delivery system without intensive automation. Errors occurred regularly but were corrected effectively although not fast. (Re)designing this system was directed at higher efficiency, cost-effectiveness, and fewer behavioral rules while maintaining the current flexibility. The future system aimed at introducing Intranet and e-mail facilities to handle orders and deliveries while reducing the number of human transactions [32].

3.3 Procedure

As part of an internship with the said provincial government [32], rapid ethnography [22] [28] in the early stages of design established a list of features of the current system, a list of requirements of the future system as well as a list of goals of the managers of the organization (not necessarily the same people who participated in the questionnaire study). Based upon these observations, a structured questionnaire, the RE*quest* [20], of 64 items was created (in Dutch), divided into 5 blocks. Three blocks were created for the purposes of the IT practitioner who performed the internship, one block was created for hypothesis testing, and one block concerned demographic information of the participants. The block for hypothesis testing was put in between the practitioner's blocks and the demographic block of items was put in last. Items were pseudo-randomly distributed over blocks. Thirty-five participants were asked to print and fill out this paper-and-pencil questionnaire, which was sent to them over the e-mail. After a few reminders, eighteen questionnaires were completed and returned, which took about a fortnight.

[1] Requirements that Must be, Should be, Could be, or Won't be on the system) [15]. 'Could' requirements are comparable to Kano's "attractive" requirements ([6], p. 4). They are not necessary but they can increase customer satisfaction.

3.4 Measurements

3.4.1 Scale construction

For those who are unacquainted with structured questionnaire design [14], we want to introduce the notions of scales, indicative and contra-indicative items, and faceted scales [18] [19]. In Section 3.4.2 we explain how our measurements were done.

Scales measure a concept or construct that is not immediately visible in the concrete world (e.g., stakeholder goals). Scales consist of multiple items that more-or-less cover a variety of aspects of 'stakeholder goals' (e.g., efficiency, cost-effectiveness, etc.). The items approach the abstract concept of stakeholder goals not only from the positive side ("E-mail is fast") but also from the negative side ("E-mail is slow"). These statements form the indicative and contra-indicative items on the scale, respectively. Each item is scored for agreement. Taken together, the various items on a scale control for different interpretations of what 'stakeholder goals' might mean. Faceted scales [18] [19] systematically combine more single (sub) scales (e.g., requirements plus valence plus goals). A statement from a faceted scale can be formulated as a requirements statement (e.g., "Automated input helps me to do my work properly"). Each item is part of a larger set of statements that systematically combine, for example, the positive and negative aspects of the respective sub scales to see their different impact on agreement.

Albeit in different forms, the notion of indicative and contra-indicative items can sometimes be found in the RE literature but is hardly ever employed to construct scales with. Usually, requirements engineers confine themselves to indicative items. However, this may lead the stakeholder into an affirmative answering tendency [14]. Therefore, contra-indicative items are recommended to neutralize this tendency evoked by a measurement scale. In the present study, the need for contra-indications was also theoretically based. Our assumption was that features a system should *not* have are as important to assess the stakeholders' needs as the features that the system must have. A similar thought can be found with Kano (in [6], p. 5), who speaks of "functional" versus "disfunctional" forms of questions.

3.4.2 Scale construction in the case study

In helping to validate the MuSCoW list created by the practitioner involved in the internship, we developed two scales (Agreed-upon Requirements and Current System) as well as 5 single survey items. Agreed-upon Requirements consisted of 7 indicative and 7 contra-indicative items that pertained to ordering procedures, order handling, and checking available warehouse space. Current System consisted of 4 indicative and 4 contra-indicative items that pertained to the current way of handling orders, focusing on flexibility and efficiency. The 5 single survey items controlled for the level of acquaintance with the fact that after using 8m^2 of warehouse space, users should pay a fee, which need not concern us here. All (5+14+8=) 27 items were presented in the form of statements about the system followed by a 6-point rating scale (0= completely disagree, 5= completely agree).

In addition to the scales that helped to validate the MuSCoW list, we also created a faceted scale [18] [19] for hypotheses testing, called Stakeholders' Needs. It consisted of three sub scales: Requirements, Goals, and Valence towards proposed features of the new system. The sub scale Requirements consisted of the same items as Agreed-upon Requirements but based on the ethnographical study during the internship, these items were categorized as either must have or won't have. Must have requirements covered aspects of automation and digitalization of operations whereas won't have requirements keyed manual aspects and human interference that was typical for the old system. Goals were divided into goals to approach (achieve) or goals to avoid. Goals were related to the work of the managers and included aspects of time efficiency, error reduction, and cost-effectiveness. Valence was operationalized as keying support or obstruction of goals by the proposed feature.

Together, items on the faceted scale Stakeholders' Needs combined a requirement with a certain valence to a goal. Items on the scale Stakeholders' Needs followed the structure:

<Requirement (must or won't have)> has <Valence (supports or obstructs)> towards a <Goal (that you want to approach or want to avoid) >

By systematically combining the three sub scales, we produced eight categories of items. For each type, 3 variants were prepared, resulting into 24 items on the scale Stakeholders' Needs.

1. Must requirement – supports – goal to approach (× 3)
2. Must requirement – supports – goal to avoid (× 3)
3. Must requirement – obstructs – goal to approach (× 3)
4. Must requirement – obstructs – goal to avoid (× 3)
5. Won't requirement – supports – goal to approach (× 3)
6. Won't requirement – supports – goal to avoid (× 3)
7. Won't requirement – obstructs – goal to approach (× 3)
8. Won't requirement – obstructs – goal to avoid (× 3)

An example of a category 1 item is "Notification by e-mail that an order will be delivered facilitates a good planning." "Notification by e-mail that an order will be delivered" was a must requirement, "facilitates" supposedly induced positive valence (is in support of), and "a good planning" was a lower-level business goal (that managers wanted to approach in their work).

Moreover, upon request of the IT practitioner, two more indicative and two contra-indicative filler items were inserted. This made a total of (24+4=) 28 items on the scale Stakeholders' Needs, which entered the final questionnaire in a pseudo-random order [20]. Items were followed by a 6-point rating scale (0= completely disagree, 5= completely agree). Further, demographic information was sampled, such as sex, age, service, sector, function, and number of years in function.

Two staff members who were not involved in the actual test checked the items for readability and understandability. Given the time frame of system development and the duration of the internship, it was impossible to pretest the questionnaire on psychometric quality. Therefore, controls had to be performed post hoc.

4. ANALYSIS AND RESULTS

After the completed questionnaires were returned, the data were entered in an SPSS 11.0 data matrix for statistical analysis.[2] In depth details about the statistical procedures followed and intermediate results can be found in [20]. In Section 4.1, we evaluated the scales Agreed-upon Requirements, Current System, and Stakeholders' Needs for psychometric quality. In

[2] Statistical Package for the Social Sciences, SPSS Inc.

Section 4.1.3, manipulation checks and some preliminary hypotheses testing was performed with multivariate analyses of variance.[3] In Section 4.1.4, we explored the structure of the different variables on the Stakeholders' Needs scale with multiple regression analyses to test H1 and H2.

4.1 Scale Analysis

Two types of scales were analyzed for psychometric qualities: Agreed-upon Requirements and Current System on the one hand and Stakeholders' Needs on the other. We regarded Agreed-upon Requirements and Current System as conventional bipolar scales. That is, we summated the indicative and contra-indicative items and treated them as one scale with two opposite extremes or poles. Stakeholders' Needs was a faceted scale, needed to explicitly connect a system feature to an outcome-expectancy towards goals. For theoretical as well as methodological reasons (Section 4.1.2), we treated Stakeholders' Needs as a set of 6 unipolar sub scales. Here, the indicative items of one variable (e.g., goals) are considered a sub scale of their own (e.g., sub scale Goals to Approach), which is relatively independent of the sub scale formed by the respective contra-indicative items (e.g., sub scale Goals to Avoid).

4.1.1 Agreed-upon Requirements and Current System

The contra-indicative items of Agreed-upon Requirements and Current System were reverse-scaled: A score of 0 was transformed to a 5, 1 to 4, etc. We then tested whether items correlated with their own scale by means of Corrected Item-Total Correlations and Cronbach's alpha (indicating reliability). The degree to which items did not correlate with other scales was tested with Pearson correlations.

We conducted item analyses on the 14 items hypothesized to assess Agreed-upon Requirements and the 8 items to assess Current System. Initially, each item was correlated with its own scale (with the item removed) and with the other scale. In certain cases, items were more highly correlated with the other scale than with their own scale. Based on these results and additional item analyses, the psychometrically weak items were eliminated from their scales.

For these shortened scales, each item was again correlated with its own scale (with the item removed) and with the other scale. The results of these analyses are shown in Table 1. In support of the measure's validity, items always were more highly correlated with their own scale than with the other scale. Cronbach's alphas were computed to obtain internal consistency estimates of reliability for these two scales. The standardized item alphas for the Agreed-upon Requirements and Current System scales were .70 and .65, respectively, which is sufficient.

Table 1. Reliability of revised scales and correlations of each item with its own scale (in bold type) and with the other scale

	Scales	
Items	*Agreed-upon Require-ments*	*Current System*
Agreed-upon Requirements		
Direct ordering at warehouse	**.48**	-.12
Order (re)directed by computer	**.54**	-.25
Computer access to order status	**.53**	.03
Reply e-mail for delivery notification	**.41**	-.02
E-mail warning when ordering problems occur	**.23**	-.03
Check available storage room on ATRIUM intranet	**.33**	-.26
Access to order status via secretary	**.34**	-.15
Current System		
Current way of doing orders ignores my wishes	-.55	**.67**
Present flexibility of handling orders is bad	-.34	**.48**
The efficiency of currently doing orders is low	-.12	**.33**
Automatic signaling that my storage room is full is useless in saving time	.05	**.22**
Cronbach's alpha	.70	.62
Standardized Cronbach's alpha	.70	.65

4.1.2 Stakeholders' Needs

The 28 items on the faceted scale of Stakeholders' Needs consisted of a requirement (1) and the valence (2) towards that requirement in view of a goal (3) as related to the work. Ample empirical literature exists [8], [9], [24], [31], [10], providing evidence that concepts related to valence should preferably be treated as unipolar scales rather than bipolar. Therefore, the sub scales Requirements, Valence, and Goals were subdivided according to their item types (indicative vs. contra-indicative). This resulted in six unipolar sub scales of Requirements Must, Requirements Won't, Valence Support, Valence Obstruct, Goals Approach, and Goals Avoid. These 6 sub scales had systematically differing combinations of the items on the Stakeholders' Needs scale.

First, we correlated each item with its own sub scale (with the item removed) and with the other sub scales. In many cases, items were more highly correlated with another sub scale than with their own sub scale. Probably, this is because the items on the Stakeholders' Needs scale explicitly related requirements, valencies, and goals, which may explain the relatively strong interdependency of sub scales. Based on these results and additional item analyses, the psychometrically weak items were eliminated from their sub scales.

Each item on the shortened scales was again correlated with its own sub scale (with the item removed) and with the other sub scales. The results of these analyses are displayed in Table 2.

The measures' reliabilities were not extremely good. The 3 best items on a sub scale were not always more highly correlated with their own sub scale than with the other sub scales.

[3] Note that the GLM > Repeated measures option in the new releases of SPSS is more-or-less similar to the MANOVA procedures in the syntax editor. The latter option was used in this study.

Cronbach’s alphas were calculated to attain internal consistency estimates of reliability for the 3-item sub scales (Table 2). Standardized item alphas were between .48 and .78, which is weak to good. However, scales with alpha > .60 are actually needed only for placing individuals on a standardized scale. With the necessary precaution, alpha around .60 may be acceptable [17] for grouped individuals like our managers.

Table 2. Reliability of the 6 sub scales of Stakeholders’ Needs and correlations of each item with its own sub scale (in bold type) and with the other sub scales. Suspect items have an asterisk

Items	Sub scales					
	Require-ments		*Valence*		*Goals*	
Requirements Must	*must*	*won’t*	*sup-port*	*ob-struct*	*ap-proach*	*avoid*
Direct transaction	**.61**	-.30	.11	-.53	.14	-.41
*Order (re)directed	**.18**	-.28	.04	-.53	.21	-.34
E-mail announcement	**.66**	-.11	.56	.08	.25	-.52
Requirements Won’t	*must*	*won't*	*sup-port*	*ob-struct*	*ap-proach*	*avoid*
*Knowing exactly where…	-.23	**.25**	-.49	.20	-.53	.28
*Checking on problems…	-.11	**.37**	-.23	.69	-.46	.20
*Checking free storage…	-.14	**.27**	.02	.52	-.37	.06
Valence Support	*must*	*won't*	*sup-port*	*ob-struct*	*ap-proach*	*avoid*
*Check available storage…	.55	-.78	**.41**	.05	.68	-.35
Direct transaction	.57	-.25	**.76**	.32	.27	-.39
Checking on problems…	.48	-.21	**.72**	.26	.21	-.32
Valence Obstruct	*must*	*won't*	*sup-port*	*ob-struct*	*ap-proach*	*avoid*
*Checking free storage…	-.14	.36	.02	**.11**	-.37	.06
Order (re)directed ...	-.21	.28	-.04	**.42**	-.21	.34
Administration of …	-.22	.35	-.08	**.48**	-.38	.02
Goals Approach	*must*	*won't*	*sup-port*	*ob-struct*	*ap-proach*	*avoid*
*Knowing exactly where…	.23	-.67	.49	-.20	**.33**	-.28
*Checking on problems…	.11	-.43	.23	-.69	**.57**	-.20
E-mail warnings …	.16	-.40	.21	-.60	**.70**	-.08
Goals Avoid	*must*	*won't*	*sup-port*	*ob-struct*	*ap-proach*	*avoid*
*Delivery notification …	-.28	.12	-.38	.03	-.05	**.16**
Direct transaction …	-.13	.30	-.11	.53	-.14	**.67**
Order (re)direction	-.21	.28	-.04	.53	-.21	**.62**
	must	*won't*	*sup-port*	*ob-struct*	*ap-proach*	*avoid*
Cronbach’s alpha	.64	.48	.78	.50	.69	.64
Standardized Cronbach’s alpha	.63	.48	.78	.50	.72	.61

4.1.3 MANOVA on Stakeholders’ Needs

We treated the faceted scale of Stakeholders’ Needs as a nested factorial design (within-subjects) of the 3-leveled factor Scales (requirements vs. valence vs. goals) and the 2-leveled factor Item Type (indicative vs. contra-indicative). In view of this setting, 6 within-subjects (dependent) variables were calculated from the 3 items per sub scale (Table 2): The grand mean average level of agreement to Requirements (must vs. won’t have) vs. Valence (support vs. obstruct) vs. Goals (to approach vs. to avoid). Moreover, we calculated the grand mean averages over the items on the revised scales Agreed-upon Requirements and Current System. As a preliminary test, a One-Way MANOVA was run to see the effects of the fixed factors Service (4), Sector (7), and Sex (2) on the grand means of the 6 within-subjects (dependent) variables. The effects of Age (28-58), Number of Years in Service (1-36), Agreed-upon Requirements, and Current System were controlled for by treating them as covariates. Function (14) was not analyzed because each function had but one or two managers. Multivariate tests according to Pillai showed that none of the fixed or covariate factors were significant (.36 < F < 1.59; .479 ≤ p ≤ .700) for either of the dependents.

In addition, the main test consisted of a 3*2 MANOVA of Scales (Requirements vs. Valence vs. Goals) (within-subjects) and Item Type (indicative vs. contra-indicative) (within-subjects) on the grand mean average agreement to the 6 sub scales. Results can be found in Figure 1 and Table 3.

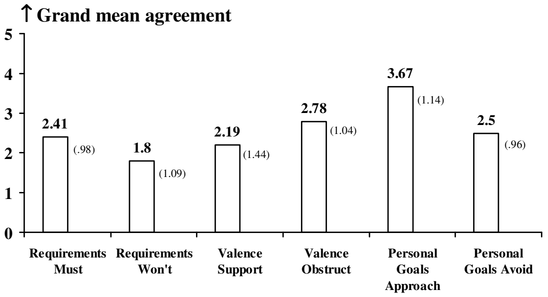


Figure 1. Grand mean average agreement to the 6 sub scales of Stakeholders’ Needs (N= 18). Standard deviations are between parentheses

Table 3. Summary of results of MANOVA on Requirements Must, Requirements Won’t, Valence Support, Valence Obstruct, Goals Approach, and Goals Avoid

Main effect of Item Type (indicative vs. contra-indicative)

F(1,17)= 1.44, p= .246

Main effect of Scales (Requirements vs. Valence vs. Goals)

Pillai’s Trace = .44, F(2,16)= 6.40, p= .009

Parameter (Requirements vs. Valence)
Coefficient= -.76, t= -1.55, p= .139, η_p^2= .12
Parameter (Requirements vs. Goals)
Coefficient= -1.96, t= -3.57, p= .002, η_p^2= .44
Parameter (Valence vs. Goals)
Coefficient= -1.20, t= -2.34, p= .032, η_p^2= .24

Interaction Item Type (indicative vs. contra-indicative) and Scales (Requirements vs. Valence vs. Goals)

Pillai’s Trace = .51, F(2,16)= 8.40, p= .003

Parameter Item Type * (Requirements vs. Valence)
Coefficient= 1.20, t= 2.51, p= .022, η_p^2= .27
Parameter Item Type * (Requirements vs. Goals)
Coefficient= -.56, t= -4.04, p= .001, η_p^2= .49
Parameter Item Type * (Valence vs. Goals)
Coefficient= -1.76, t= -3.25, p= .005, η_p^2= .38

The most important result of Figure 1 in combination with Table 3 is the significant interaction between Item Type (indicative vs. contra-indicative) and Scales (Requirements vs. Valence vs. Goals) (Pillai's Trace = .51, F(2,16)= 8.40, p= .003).

To start with the strongest significant contrast, parameter estimates showed that indicative items of Requirements (M_{Must}= 2.41) evoked higher levels of agreement than contra-indicative items ($M_{Won't}$= 1.80), which may be expected. This difference was larger, however, for Goals. Indicative items of Goals ($M_{Approach}$= 3.67) evoked the highest level of agreement in this study, more than contra-indicative items (M_{Avoid}= 2.50) (parameter coefficient= -.56, t= -4.04, p= .001, η_p^2= .49).

A less strong but also significant contrast was found for the indicative items of Valence ($M_{Support}$= 2.19), which surprisingly, elicited *lower* levels of agreement than the contra-indicative items ($M_{Obstruct}$= 2.78). As visible in the previous paragraph, the opposite happened for Goals (parameter coefficient= -1.76, t= -3.25, p= .005, η_p^2= .38).

The third contrast was only marginally significant according to Bonferroni (α= .05/3≈ .017) and should be considered merely a trend. Parameter estimates showed that the level of agreement to indicative and contra-indicative items in Requirements had an inverse pattern as compared to Valence (parameter coefficient= 1.20, t= 2.51, p= .022, η_p^2= .27).

These interactions were sustained by a significant main effect of Scales (Pillai's Trace = .44, F(2,16)= 6.40, p= .009), which was mainly based on the contrast between Requirements and Goals (parameter coefficient= -1.96, t= -3.57, p= .002, η_p^2= .44). The difference between Valence and Goals was much smaller and only marginally significant (parameter coefficient= -1.20, t= -2.34, p= .032, η_p^2= .24) according to Bonferroni (.05/3≈ .017). In other words, the strongest interactions and main effects were produced by Goals in combination with Requirements, whereas the weaker interactions and main effects were generated by Valence in combination with Goals.

The following observations can be done from these results. First, the three variables Requirements, Valence, and Goals could be successfully applied during the requirements engineering of a logistic warehouse management system. Requirements, Goals, and Valence all produced significant (interaction) effects on how much the managers agreed to a requirements statement about the (planned) system. Goals Approach had the strongest positive effect on agreement whereas Requirements Won't had the most negative effect. Moreover, Requirements, Goals, and Valence were not independent but affected one another (significant interactions). Missing out on one weakens the explanation why requirements are (dis)agreed upon.

Second, goals (i.e. those that the managers wanted to achieve) played a leading role here, inducing the largest effects. This implies that the stakeholders' concerns [21] are indispensable for requirements validation. Interestingly, the goals these managers pursued in their work all pertained to efficiency and not, for example, cost-effectiveness. The sub scale of Goals Approach pertained to the goals 'quick order processing' (a speed aspect), 'accurate order handling' (an accuracy aspect), and 'efficient work' (high speed and high accuracy combined).

The third observation concerns the valence towards the warehouse management system. That is, the manager's expectancy whether a system feature would support or frustrate certain of his or her goals and concerns. In this study, the managers felt that the proposed features would obstruct their goals rather than support them. The finding that Valence Support had less effect on the level of agreement than Valence Obstruct, moreover, counters a possible bias towards positive answering tendencies.

Fourth, from a more general point of view, we see that goals (i.e. those to be achieved) have the largest impact on agreement, followed by the emotional component of valence, and only then by the proposed requirements. This underscores that RE should indeed be goal-driven. Moreover, the results suggest that requirements engineers should look into the motivational aspects of stakeholders to gain more insight in why requirements are agreed upon or not. This can be done by explicitly connecting a proposed system feature to a (lower-level) goal and asking what positive or negative outcome stakeholders expect (valence) with regard to achieving their goals with the system.

4.1.4 Regression on Agreed-upon Requirements, Current System, and Stakeholders' Needs

H1 and H2 predicted that requirements are explained by valence (as a mediator), which in turn is directed by lower-level business goals. Yet, certain constellations could counter those predictions, such as (dissatisfaction with) the current system or direct contributions of goals to requirements without interference of the emotional component of valence. Due to the small number of respondents (N= 18) a Structural Equation Model could not be performed. Instead, the analysis was restricted to a set of multiple regressions. The research question (RQ) ran as follows:

RQ1. How well do valence and goals predict agreement to requirements, controlling for agreement to the current system?

To execute a first multiple regression analysis (Method Enter), RQ1 was restated as:

RQ1a. How well do Valence Support, Valence Obstruct, Goals Approach, and Goals Avoid predict Agreed-upon Requirements, controlling for agreement to the Current System, Requirements Must, and Requirements Won't?

Agreed-upon Requirements acted as the dependent variable in the regression with four ordered sets of predictors, using the items as displayed in Table 1 and Table 2. Current System was entered in the first step as categorical independent variable, Requirements Must and Requirements Won't were entered in the second step, Valence Support and Valence Obstruct in the third step, and Goals Approach and Goals Avoid in the fourth.

None of the (sets of) predictors accounted for a significant amount of the variability of Agreed-upon Requirements [20].

A second multiple regression analysis followed the research question:

RQ2a. How well do Valence Support and Goals Approach, Valence Obstruct and Goals Avoid predict Requirements Must, controlling for agreement to the Current System, Agreed-upon Requirements, and Requirements Won't?

RQ2b. How well do Valence Obstruct and Goals Avoid, Valence Support and Goals Approach predict Requirements Won't, controlling for agreement to the Current System, Agreed-upon Requirements, and Requirements Must?

With regard to RQ2a, Goals Avoid and Valence Obstruct together accounted for a significant amount (90%) of the Requirements Must variability, R^2= .93, R^2_{adj}= .90, F(5,12)=

30.30, p= .000. Goals Approach and Valence Support did not significantly increment the percent of explained variance of Requirements Must, R^2_{change}= .01, F(2,10)= .33, p= .728. We also assessed the relative importance of Goals Avoid and Valence Obstruct in predicting Requirements Must. It seemed that Goals Avoid was most strongly related to the Requirements Must (standardized β= -.97, t= -9.48, p= .000). Supporting this conclusion is the height of the standardized Beta coefficient and the strength of the correlation between Goals Avoid and Requirements Must, partialling out the effects of all other predictors ($r_{partial}$= -.94, r_{part}= -.74). Valence Obstruct offered little or no additional predictive power beyond that contributed by the Goals Avoid measure.

Regarding RQ2b, Goals Approach and Valence Support, accounted for a significant amount (70%) of the Requirements Won't variability, R^2= .79, R^2_{adj}= .70, F(5,12)= 9.01, p= .001. Goals Avoid and Valence Obstruct did not increment the percent of explained variance of Requirements Won't, R^2_{change}= .07, F(2,10)= 2.28, p= .153. We also assessed the relative importance of Goals Approach and Valence Support in predicting Requirements Won't. It seemed that Goals Approach was most strongly related to Requirements Won't, standardized β= -.96, t= -5.31, p= .000. Supporting this conclusion is the height of the standardized Beta coefficient and the strength of the correlation between Goals Approach and Requirements Won't, partialling out the effects of all other predictors ($r_{partial}$= -.84, r_{part}= -.70). Valence Support offered little or no additional predictive power beyond that contributed by the Goals Approach measure.

H1 and H2 further predict that valence is explained by goals. Therefore, RQ3a and RQ3b ran as follows:

RQ3a. How well do Goals Approach predict Valence Support, controlling for Goals Avoid and Valence Obstruct?

RQ3b. How well do Goals Avoid predict Valence Obstruct, controlling for Goals Approach and Valence Support?

No significant results were obtained in the respective regression analyses [20].

Based on the series of multiple regression analyses, the first remark that can be made is on the bipolarity of variables. Regression on the bipolar Agreed-upon Requirements scale yielded no significant results what so ever, whereas regression on the unipolar sub scales Requirements Must and Requirements Won't did. Second, in a bipolar conception important information is lost: H1 was refuted because the level of agreement to must requirements was best explained by goals stakeholders wanted to avoid (!) and H2 was refuted because won't requirements were best explained by goals stakeholders wanted to approach (!). Third, these findings are in line with the literature on attitudinal ambivalence [8] [9] [24] [31] [10].

The findings in the regression analyses on Requirements Must and Requirements Won't can be summarized and interpreted as follows. Agreed-upon Requirements and Current System did not explain agreement to Requirements of either sort. This teaches us two things. *It is better to explicitly connect a requirement to a (lower-level business) goal and state the expected outcome valence than to have an agreement score to a requirement (or goal) without more. In addition, (dis)agreement with the current system does not predict agreement to the requirements of a future system.*

The variables that did explain Requirements Must and Requirements Won't formed another constellation than expected. H1 expected that requirements the system must meet are explained by a positive outcome valence of the proposed features towards goals the stakeholder wants to achieve in his or her work. The opposite was the case, however. Goals Avoid significantly accounted for 90% of the variability in agreement to Requirements Must. A similar structure was found for the requirements of features the system won't have. H2 anticipated that what the system won't have is predicted by a negative outcome valence of the proposed features towards states and situations the stakeholder wants to avoid in his or her work. Again the reverse happened, because Goals Approach significantly accounted for 70% of the variability in agreement to Requirements Won't. *Probably, requirements the system must meet had a baseline agreement that was pushed down by the disagreement of the stakeholder to an undesired future situation. Mirroring this, requirements of things the system won't have, evoked a baseline disagreement that was pulled up by the agreement of the stakeholder to a desired future situation.* This is why goals to avoid predicted 'must haves' better than goals to approach did. It is the same reason why goals to approach predicted 'won't haves' better than goals to avoid. These findings for the future – for these managers still somewhat fictional – system corresponds to what [31] called 'subjective ambivalence,' that is, a conflict between simultaneously occurring positive and negative attitudes towards a feature or object (also called evaluative tension or attitudinal ambivalence). Similar positive-negative asymmetry effects are also repeatedly confirmed in the field of impression formation, e.g., [34].

As another matter, H1 and H2 assumed that valence was a mediator between agreement to requirements and goals. This was not demonstrated by the regression results, however. The relative importance of Goals Avoid to Requirements Must was significantly higher than for all other predictors, including Valence (standardized β= -.97, t= -9.48, p= .000, $r_{partial}$= -.94, r_{part}= -.74). Likewise, the relative importance of Goals Approach to Requirements Won't also was significantly higher than for all other predictors, including Valence (standardized β= -.96, t= -5.31, p= .000, $r_{partial}$= -.84, r_{part}= -.70). *This means that there is a direct link between the situation a stakeholder wants to avoid and the requirements that the system must have to achieve that. In addition, there is a direct link between the goals a stakeholder wants to approach and the requirements that should be left out from the system. Valence, expectations of support or frustration of goals by the proposed features, plays a moderating role in explaining agreement to requirements.*

Valence moderates the relational strength between goals and requirements. On the one hand, MANOVA (Table 3) showed that valence was involved in a significant interaction with goals on agreement. On the other hand, valence had no significant main effect according to Bonferroni. Additional multiple regressions indicated that Goals Approach did not significantly predict Valence Support and that Goals Avoid did not significantly predict Valence Obstruct. Therefore, valence should be regarded a moderating rather than a mediating variable.

5. CONCLUSION/DISCUSSION

The needs of the stakeholder should be modeled as a unipolar constellation. Situations a stakeholder does not want to get into, directly and to a large extent explain what the system must offer. This mirrors the finding that situations a stakeholder *does* want to reach, directly and to a large extent explain what the system must *not* offer. Valence, the expectation of the stakeholder whether a proposed feature might harm or sustain a goal at work, appears not to be a necessary step in the initial stage of RE. It does, however, have a moderating effect, increasing or decreasing the level of agreement to a requirements statement.

The most important information an IT practitioner could extract from a system's stakeholders are covered by four questions, then. What are the things in life or work that you do not want? What can the system offer to avoid those things? What are the things in life or work that you do want? What should the system *not* have in order to support that? In view of the relative importance of features the future system should not have, it seems that analysis of the won't requirements is underestimated in industrial practice.

"To identify possible inconsistencies between what is wanted and what is possible to meet" [2], we analyzed the matching between requirements and managerial goals. We did so by querying the 'subjective judgments' [2] of a group of managers with regard to the positive or negative valence they attached to the requirements in view of their lower-level business goals. In so doing, we succeeded in our RE*quest* '… to align system function with stakeholder values...' [4].

The structured requirements-engineering questionnaire RE*quest* assessed the actual level of agreement to requirements that were supposedly agreed upon in earlier negotiations (Agreed-upon Requirements). Moreover, the agreement to the Current System was assessed as well as the Stakeholders' Needs. The latter scale was subdivided into items that measured the positive and negative outcome expectancies (valence) the managers had of requirements to goals.

The results revealed that Requirements, Valence, and Goals had a significant impact on the level of agreement. These variables are affecting one another so that combining these three variables into one scale of Stakeholders' Needs seems to be an addition to common RE methods (e.g., [12] [29]). The goals had the strongest impact on the level of agreement. Therefore, requirement engineers are recommended to always take these into account.

With respect to valence, the managers that evaluated the requirements of the future system thought that rigorous automation and fewer behavioral rules would harm their goals on the workfloor rather than sustain them. A finding like this is most informative for the management of change. It suggests that in this group of managers implementing the features as agreed upon in earlier negotiations will lead to non-acceptance of the technology. In this light, it is plausible that the sub scale of Requirements within the Stakeholders' Needs scale had the weakest effects on the level of agreement. Putting a score to a requirement without more (e.g., Kano in [6], p. 5) apparently is not the most informative way to do requirements engineering. Requirements should be coupled to a goal while explicitly asking for the direction of the stakeholders' expectations (valence). In addition, (dis)satisfaction with the current situation is not a good predictor of the level of agreement to requirements in a future situation.

A sequence of multiple regressions shed further light on the structure of requirements change. It turned out that the sources of change should be conceived of as unipolar dimensions. That is, requirements should be treated separately as 'must have' versus 'won't have' because these are explained differently from the underlying goals and concerns of the stakeholders. To arrive at such an explanation, valence and goals also should be treated as unipolar. In fact, we have found two sub models of requirements change: Variance in agreement to must requirements is best explained by goals stakeholders want to avoid (sub model 1) and variance in won't requirements are best explained by goals stakeholders want to approach (sub model 2). In line with the literature on emotional biases and action tendencies, stakeholders maintain a baseline agreement to must requirements, which is regulated by the 'threat' to goals in the future ('cover your ass'). In opposition, won't have requirements evoke a baseline disagreement that is governed by agreement to possible support of desirable goals in the future ('make life easier').[4] We coin this mechanism the *goal-to-requirements chiasm* or *χ-effect* (CHI-effect) on the stakeholders' agreement to requirement statements. The direct explanatory relation between positive or negative requirements and their respective inverse counterparts in goals is moderated by valence (positive or negative expectations). Valence can increase or decrease the influence of goals on requirements. In Figure 2, a graphical display of the two sub models of requirements change is exhibited as they emerge from the empirical findings.

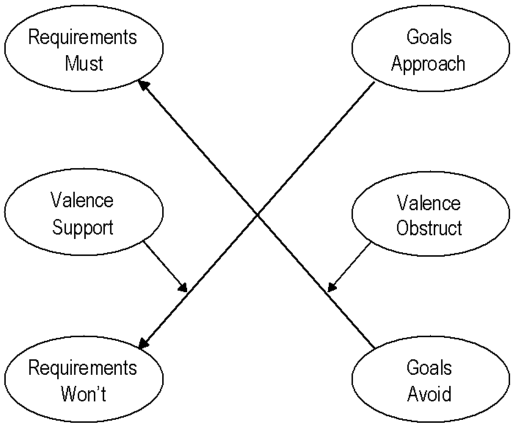


Figure 2. The goals-to-requirements chiasm or χ-effect as derived from the empirical data

6. RELATED WORK

In system design, requirements change as the situation in which these systems function evolves [1]. Situations change as a result of certain events, a change of tasks, adopting another business model or a change in (organizational) culture [12]. Stakeholders call for or dismiss requirements and errors should be repaired [1]. However, different stakeholders may have conflicting requirements [35], which points at opposing goals or different means of achieving them in the new situation. While situations, and subsequently, requirements develop, uncertainty can be managed and the new situation controlled as soon as requirements are again agreed-upon [1]. To manage a change request, goals are fundamental for discovering conflicts among (the new) requirements [26]. "Goals provide the rationale for requirements i.e. requirements represent one particular way to achieve high-level goals" [1] (e.g., strategic business goals).

To manage conflicting requirements and requirements change, system developers need to comprehend the sources of conflict and the mechanisms by which conflicts emerge [2]. We showed that constructing a measurement tool that systematically uses conflicting goals, contradictory requirements (and opposite valences for that matter) can be successful in pinpointing the

[4] We owe the "make life easier, while covering your ass" interpretation to Jo Geraedts, Industrial Design Dept., Océ-Technologies.

sources of conflict. In system development, questionnaires have already been applied (e.g., [23], [7], [12]) but these mainly worked from single one-response survey items.[5] However, constructing items on a scale that indicate and contra-indicate a certain concept as well as statistically assessing the psychometric quality of items is a more reliable and valid approach [14] to requirements-questionnaire design.

In our questionnaire RE*quest*, we followed the strategy of [2] to deal with requirements as alternatives to operationalize goals. The results indicated that variance in the level of agreement to goals of stakeholders is one of the main sources of requirements change. It predicted the largest part of variance in agreement to requirements. The strength of this relation (70% and 90%) indeed supports the assumption that requirements are refinements of goals [1]. The results of our study indicate that stakeholders evaluate the risks and benefits (Kano in [6], p. 9) of the new system in terms of emotional valence towards proposed features. Assessing the valence of requirements and features towards goals is important because valence modulates the level of agreement to a list of requirements.

The empirical results of running the RE*quest* also made us identify the mechanism by which conflicts in requirements emerge (cf. [2]). The χ-effect suggests that stakeholders have a baseline agreement to requirements that the system must meet ("Of course, my system is UNIX-based because I want it to be reliable"). This finding links up with the work of Kano (in [6], p. 4), who states that customers have so called "must be" requirements on a product. Customer satisfaction decreases if the product does not satisfy the must be requirements (e.g., breaks on a car) but remains neutral if the respective functionality improves (e.g., breaks with ABS). What we can add to Kano's proposal, then, is that changes in agreement to requirements are directed by goal states the stakeholders want to avoid with the system ("On the other hand, the UNIX system should not be all too difficult to operate"). Furthermore, and as a counterpart of Kano's "must be" requirements, stakeholders have a baseline disagreement to proposed features the system should not include ("Of course, my system is not Windows-based because I hate its instability"). Following Kano, we could coin these features the "won't be" requirements. Changes in disagreement, then, are predicated by goal states the stakeholders want to achieve with the system ("But I do like to work with an easy-to-handle graphical user interface"). When engineers merely investigate the baseline agreement to must requirements as related to positive goal states (UNIX guarantees reliability) and the baseline disagreement to won't requirements as related to negative goal states (Windows promises unreliability) it seems that there are no conflicts. However, the wish list stakeholders put forth can yet contain conflicting requirements because they want UNIX for reliability but not for usability and they want Windows for usability but not for reliability.

7. FUTURE WORK

The main focus of our research is to repeat our finding of the goal-to-requirements chiasm. We are currently involved with the Dutch police force to do RE on a capacity management system (CMS) for planning and allocating personnel. To date, the Dutch police undergo a major business model change in moving from a public service to a self-supporting business-like organization. We will explore whether the requirements on the CMS can be explained from the officers' goals and concerns in the predicted constellation (avoid-to-must and approach-to-won't). We will do this from two points of view. One group of officers works from a business perspective (requirements as related to business goals) and one group will work from a personal perspective (the same requirements as related to personal goals).

A second replication study is currently administered with interaction designers and software engineers from 6 different countries who are asked to assemble a computer off-the-shelf (COTS). Two types of systems are offered from which they can pick their features. One with software and hardware that is outmoded (e.g., a cathode ray tube monitor and a 5¼" floppy drive) and one that is state-of-the-art (e.g., 63" wide screen plasma monitor and an AMD Athlon 64 processor). Again, the question is whether we can produce the goal-to-requirements chiasm.

Stakeholder participation and psychological involvement foster satisfaction with the system and improves the development of products [5] [33]. It would be interesting to find out if adopting the approach proposed in this paper will actually increase customer satisfaction and whether it ensures a more correct alignment between business and IT. If we can repeat our findings, this is something we intend to investigate in the future.

In this study, we employed theory and methods of psychology, invited a group of managers as participants in our requirements validation test, and used the results to improve the logistic warehouse management system [32]. On our way, we gained more insight into the sources and mechanisms of requirements conflicts and requirements change.

8. ACKNOWLEDGMENTS

This paper is part of the Innovation Oriented research Program (IOP) for Human-Machine Interaction entitled *Integrating Design of Business Processes and Task Analysis,* granted to the first author by the Senter*Novem* Agency of the Dutch Ministry of Economic Affairs in The Hague, grant Mmi99009. Johan F. Hoorn, Hans van Vliet, and Gerrit van der Veer work in the Faculty of Sciences, Dept. of Computer Science, Section Information Management & Software Engineering. Elly A. Konijn works in the Faculty of Social Sciences, Dept. of Communication Science. Sandra Pronk is acknowledged for her help in administering the survey and for sampling the data.

9. REFERENCES

[1] Alves, C., and Finkelstein, A. Challenges in COTS-decision making: A goal-driven requirements engineering perspective. *Workshop on Software Engineering Decision Support* (Ischia IT, 2002).

[2] Alves, C., and Finkelstein, A. Investigating conflicts in COTS decision-making. *International Journal of Software Engineering and Knowledge Engineering, 13* (2003), 1-21.

[3] Anton, A.I., Cracken, W., and Potts, C. Goal decomposition and scenario analysis in business process reengineering, in *Proceedings of CAiSE '94* (Utrecht NL, 1994).

[4] Anton, A.I., Earp, J.B., Potts, C., and Alspaugh, T.A. The role of policy and stakeholder privacy values in requirements engineering, in *Proceedings of RE'01: International Symposium on Requirements Engineering* (Toronto Canada, 2001).

[5] In the Damian et al. study [12], question 9 could be seen as a scale for Perceived Immediate Benefit but was not analyzed that way vide the discussion of their Figure 8.

[5] Barki, H., and Hartwick, J. (1989). Rethinking the concept of user involvement. *MIS Quarterly, 13*, 1 (1989), 53-61.

[6] Berger, C., Blauth, R., Boger, D., Bolster, C., Burchill, G., DuMouchel, W., Pouliot, F., Richter, R., Rubinoff, A., Shen, D., Timko, M., and Walden, D. Kano's methods for understanding customer-defined quality. *Center for Quality of Management Journal, 2,* 4 (1993), 2-36.

[7] Burke, B. *Evolving Architecture Maturity*. META PRACTICE, META Group Inc., 2002.

[8] Cacioppo, J.T., and Berntson, G.G. Relationship between attitudes and evaluative space: A critical review, with emphasis on the separability of positive and negative substrates. *Psychological Bulletin, 115* (1994), 401-423.

[9] Cacioppo, J.T., Gardner, W.L., Berntson, G.G. The affect system has parallel and integrative processing components: Form follows function. *Journal of Personality and Social Psychology, 76* (1999), 839-855.

[10] Carroll, J.M., Yik, M.S. M, Russell, J.A., and Feldman Barrett, L. On the psychometric principles of affect. *Review of General Psychology, 3* (1999), 14-22.

[11] Coakes, E., and Elliman, T. Focus issue on legacy information systems and business process engineering: The role of stakeholders in managing change. *Communications of the AIS, Volume 2,* 1, 4 (July 1999), 1-30.

[12] Damian, D., Zowghi, D., Vaidyanathasamy, L., and Pal, Y. An industrial case study of immediate benefits of requirements engineering process improvement at the Australian Center for Unisys Software. *Empirical Software Engineering, 9,* 1-2 (2004), 45-75.

[13] Davis, F.D., Bagozzi, R.P., and Warshaw, P.R. User acceptance of computer technology: A comparison of two theoretical models. *Management Science, 35* (1989), 982-1003.

[14] Dillman, D.A. *Mail and Internet Surveys: The Tailored Design Method.* Wiley, New York NY, 1999.

[15] eRA. *Program Portal Scope Document v1.4.* (Tech. Rep.). National Institutes of Health, Bethesda MD, 2002. Available at http://era.nih.gov/Docs/Pgm_Por_Scope_4-02.doc

[16] Frijda, N.H. *The Emotions.* Cambridge University Press, New York NY, 1986.

[17] Garson, D. *Scales and Standard Measures.* Available at http://www2.chass.ncsu.edu/garson/pa765/standard.htm

[18] Guttman, L. An outline of some new methodology for social research. *Public Opinion Quarterly, 18* (1954), 395-404.

[19] Guttman, L. A faceted definition of intelligence. *Studies in Psychology, 14* (1965), 166-181.

[20] Hoorn, J.F. *Validating Stakeholders' Needs of a Logistic Warehouse Management System.* (Tech. Rep.). Free University, Amsterdam NH, 2004.

[21] IEEE Standard 1471. *IEEE Recommended Practice for Architecture Description*, 2000.

[22] Jordan, B. Ethnographic worksplace studies and computer supported cooperative work, in D. Shapiro et al. (eds.) *The Design of Computer-supported Cooperative Work and Groupware Systems.* North Holland/Elsevier, Amsterdam NL (1996), 17-42.

[23] Juristo, N., Moreno, A.M., and Vegas, S. Reviewing 25 years of testing technique experiments. *Empirical Software Engineering, 9,* 1-2 (2004), 7-44.

[24] Kaplan, K.J. On the ambivalence-indifference problem in attitude theory and measurement: A suggested modification of the semantic differential technique. *Psychological Bulletin, 77* (1972), 361-372.

[25] Van Lamsweerde, A., and Letier, E. Handling obstacles in goal-oriented requirements engineering. *Software Engineering, 26*, 10 (2000), 978-1005.

[26] Van Lamsweerde, A. Goal-Oriented Requirements Engineering: A Roundtrip from Research to Practice, in *Proceedings of the Requirements Engineering Conference, 12th IEEE International (RE'04)* (Kyoto JP, 2004), Volume 00, 4-7.

[27] Letier, E., and Van Lamsweerde, A. Agent-based tactics for goal-oriented requirements elaboration, in *Proceedings of 24th International Conference on Software Engineering* (Orlando FL, 2001).

[28] Norman, D. *The Invisible Computer.* MIT Press: Cambridge MI, 1998.

[29] Nurmuliani, N., Zowghi D., and Williams S.. Using card sorting technique to classify requirements change, in *Proceedings of the 12th IEEE International Requirements Engineering Conference (RE'04)* (Kyoto JP, 2004).

[30] Potts, C. Using schematic scenarios to understand user needs, in *Proceedings of DIS'95 - ACM Symposium on Designing Interactive Systems: Processes, Practices and Techniques* (Michigan MI, 1995).

[31] Priester, J.R. and Petty, R.E. Extending the bases of subjective attitudinal ambivalence: Interpersonal and intrapersonal antecedents of evaluative tension. *Journal of Personality and Social Psychology, 80* (2001), 19-34.

[32] Pronk, S. *From Static Store to Organized Warehouse.* (Tech. Rep.). Free University, Amsterdam NH, 2004.

[33] Robinson, W.N., and Pawlowski, S. Managing requirements inconsistency with development goal monitors. *IEEE Transactions on Software Engineering,* (Nov/Dec 1999).

[34] Skowronski, J.J., and Carlston, D.E. Negativity and extremity biases in impression formation: A review of explanations. *Psychological Bulletin, 105* (1989), 131-142.

[35] Spanoudakis, G., Finkelstein, A., and Till, D. Overlaps in requirements engineering. *Automated Software Engineering, 6*, 2 (1999), 171-198.

[36] Sutcliffe, A. *User-centred Requirements Engineering. Theory and Practice.* Springer, London UK, 2002.

Applying Value Concepts to Organisational Modelling

Eduardo Henrique Ferro Freire, Zair Abdelouahab and Simara Vieira da Rocha
Department of Electricity Engineering Federal University of Maranhão,Brazil
eduardo.freire@pop.com.br, zair@dee.ufma.br, simararocha@uol.com.br
Tel: +55 98 3217 8243
Fax: +55 98 3217 8241

Abstract

Economic value concepts are of addition importance for the viability study of e-commerce systems. Understanding, modelling, and exposing possible value exchanges in diagrams, in e-commerce, facilitates to stakeholders in order to plan and construct such systems. On the other hand, requirement engineers have demonstrated a growing need by getting a deeper comprehension about the organisation, its objectives, its goals, and its business strategies. Besides, Requirements Engineering results become more complete when organisational aspects are modeled in order to understand better organisational intentions and motivations that incorporating the desire to develop software. These aspects are obtained with the organisational requirements modelling accomplishment. Applying value concepts to organisational requirements representation, in early requirements more specifically, turns the requirements elicitation more understandable when it leads of e-commerce systems, because the value exchanges will be modeled together with organisational goals. This paper proposes applying economic value concepts to organisational modelling in order to carry out requirements elicitation of e-commerce systems. With this integration, there will be a better specification requirements elicitation documentation, and also there will be initial information of the financial viability of the e-commerce solution to be elicited. It defines a value modelling starting from the union of the value concepts with organisational modelling, and it exposes some guidelines to elaborate value organisational models.

1 Introduction

Electronic Commerce (e-commerce) may be defined as a kind of commerce where a product is known, demonstrated, and sold through electronic ways. It is needed two or more parts using this electronic way, usually the Internet, to make business transactions involving value exchanges such as goods, services, information, and money [1]. E-commerce is also considered a computer science area that has been growing a lot the past years.

We notice that to elaborate systems of this nature involves a requirements elicitation more specifically, an information capture, which includes all of economic value concepts related to e-commerce area. Requirement engineers need to know the domain and the environment where that kind of electronic business is exposed. Much information related to value concepts is important, like economic value objects, value exchanges, value offerings, and involved actors.

On the other hand, we have noticed that a growing need of requirement engineering professionals in getting a deeper comprehension about the organisation, its objectives, goals, and business strategies [6]. Also, we verify in the literature on Requirements Engineering that it turns more complete when we model organisational aspects in order to understand better organisational intentions and motivations that incorporate the desire to develop software [4]. We obtain these aspects with the organisational requirements modelling accomplishment.

Organisational modelling has the objective to supply resources in order to allow modelling intentions, relationships and motivations among members of an organisation, as well as it describes organisational goals that can originate and orientate the software system development. With these models, we can understand better organisational environment, as well as the human and work relationships, among the organisation participants. With this information, the requirements of a computational solution for organisational processes can be better elicited and specified [4].

Inside of this context, we have been motivated to unite these two important theories, the Value Theory related to economics and Organisational Modelling, on behalf of providing to requirements engineers a way more specific and complete of early requirements elicitation when we lead with e-commerce systems.

We still were motivated by the fact that the software construction in a short time, with better use of resources and chronograms execution established, depends on a good requirements definition [6]. Thus, we should have two stages to define the requirements of an e-commerce system. Firstly, we would have to catch information on organisational aspects of the software to be developed; then, we should have to make an approach about the financial viability aspects related to the value concepts which are embedded in e-commerce. Consequently, we believe that joining value concepts with Organisational Modelling in the requirements definition time turns these two stages become only one, where organisational and value aspects are seen together. Theoretically, it turns the requirements definition faster and more correct.

The main achievement of this work is to present an application proposal of economic value concepts to organisational modelling in order to carry out requirements elicitation of e-commerce systems. With this integration, we will have a better requirement elicitation specification documentation and an initial information of the financial viability of the e-commerce solution which we want elicitate.

As related work, we can identify studies related to Organisational Modelling, mainly in the University of Toronto [7] [8] (Eric Yu and John Mylopoulos), and the Federal University of Pernambuco [4] (Victor Santander and Jaelson Castro). In the Value Theory area, we give prominence to the works related to value chain [3] (Michael Porter), to the e3-value technique [5] of the University of Vrije (Jaap Gordijn), and to the methodology called Process Pattern Perspective [2] of the University of Stockholm (Prasad Jayaweera). Another important study is the work which integrates different techniques and, in this case, we put in evidence the integration work of Organisational Modelling with the Knowledge Management [6] of the Federal University of Pernambuco (Francisco Carvalho and Jaelson Castro).

We structure this work in five sections. In section 1, we present an introduction to the application of the value theory to Organisational Modelling. We describe the approach of Organisational Modelling in the section 2, relating it to Goal-driven Requirements Elicitation and defining the *i** technique for our Organisational Modelling elaboration. Section 3 shows some definitions about e-commerce systems and Value Concepts that are important when we want to determine the financial viability of such systems. In section 4, we expose the main achievement of our work that is the application of the Value Concepts to Organisational Modelling, where we propose our Value Diagram for value modelling and some guidelines to help to elaborate such diagrams. Section 5 gives the conclusions of our work.

2 Organisational Modelling

We present in this section Organisational Modelling definitions. Firstly, we relate this modelling with Goal-driven Requirements Elicitation, and, after that, we determine the use of the *i** technique, explaining the benefits of its use in Organisational Modelling.

2.1 Goal-driven Requirements Elicitation and Organisational Modelling

We found in the literature that Requirements Elicitation is the first activity to be developed in Requirements Engineering. During the phase of making the elicitation, we look to discover system requirements, usually obscure, wandered, and confusing at the beginning of a software system development [10], with the purpose to obtain relevant knowledge for the problem to be solved [6]. In general, we can affirm that Requirements Elicitation is system requirements discovery process, through the communication among the involved stakeholders, which should consider both organisational aspects and processes, and also the application domain in order to identify the users' needs [6].

We notice that scenarios have been used so much in requirements elicitation because they minimize and outline some of the great difficulties of the Requirements Engineering that are working with several users and great amount of information. In spite of recognition that scenarios are quite important in the process of Requirements Engineering, scenarios technique presents some lacks, mainly when we talk about the inclusion of aspects into the environment organisational that the software is placed [4]. Carvalho [6] exposes that aspects in organisational dynamics and interaction impacts between the organisations and the external environment have been getting a lot of importance in the last years, because the changes in the social, economic, legal, organisational, and humans' aspects, amongst others, are enlarging the traditional vision on requirements. The modern vision about requirements does not just worry about the "how" system should do, but with "what" system should do, associated with the "why" it should do, understanding organisational facts rationales [7] [8].

We noticed, therefore, for accomplishing requirement elicitation we should use goal-driven approaches together with scenarios. Goal-driven approaches focus on why systems are constructed, expressing the rationale and justification for the proposed system [9]. Furthermore, Antón [9] say that, working with goals instead of specific requirements, we communicate with stakeholders using a language based in concepts with which they are both comfortable and familiar.

In order to model aspects into the environment organisational, and suppressing the scenarios lacks, we can achieve organisational requirements modelling. With organisational modelling, we look forward to have a better comprehension about organisational intentions and motivations that incorporate the software development desire, and get a deeper understanding about the organisation, their objectives, goals and business strategies. This kind of modelling aims at supplying resources to allow the intentions, relationships and motivations modelling among the organisation members, as well as also describing organisational goals that can originate and orientate the software system development.

2.2 The *i** technique for Organisational Aspects Modelling

In this work, we advise using the framework *i** technique for organisational aspects modelling, because this technique allows us a better understanding of organisational requirements that will get impact in the systems, and also identifying alternatives for several processes of the organisation [7].

We found in Organisational Modelling studies that an organisational model is a representation of the structure, activities, organisational processes, information, resources, goals and government restrictions (legal or other nature), that help us to understand the complex interactions between organisations and people. Thus, we notice that *i** technique allows us understanding organisational requirements that will get impacts on the system to be developed, as well as it aids us to identify it alternatives for organisational processes [7] [8].

Despite *i** technique does not express any time order, it propitiates us an initial understanding of the problem to be solved in the organisation, as well as how computational systems could collaborate in the solution of this problem [7]. It gives us mechanisms that allow expressing tasks, goals, softgoals, and resources, associated to actors needs and intentions in organisational environment. When we need to place sequence on events, we must extend this technique. But, in general, the *i** technique is considered easy to understand by stakeholders, using close domain concepts of organisational actors knowledge [4].

It is stated that *i** was developed to model intentions in the strategic actors relationships [7]. The *i** technique is based on actors dependence model, where these actors dependences are analyzed so that goals could be achieved, the tasks could be performed, and resources could be supplied. Those dependences are intentional and based on concepts of goals, abilities, beliefs, compromises and so on [7] [8].

In the framework of actors' dependencies analysis in *i**, we find two models defined: the Strategic Dependence (SD) model and the Strategic Rationale (SR) model. The Strategic Dependence model is used to describe the actors dependence relationships in organisational context. On the other hand, the Strategic Rationale model is used to describe the interests and concerns of stakeholders and how these feelings can be led in various systems and environments configurations. This model offers us a more detailed level of modelling by looking "inside" actors to model internal intentional relationships [7] [8]. We can conclude that SD is a general model and SR is more specific model.

Models are graphs composed by nodes and their links. Each node represents an actor, and each link between two actors indicates that an actor depends on another actor to do something. The dependent actor is called *Depender*, and the actor on who is the dependence is called *Dependee*. The object on which the dependence relationship is centered is called *Dependum*, and it can be a Goal, a Task, a Resource or a Softgoal [7] [8]. *Dependum* gives the name to dependence type.

In *goal dependence*, we see an actor depends on another to make a condition in the world come true, satisfying its intention and achieving its goal. We remember that *i** does not prioritize goals neither distinguish kinds of goal. We have to do it explicitly, if it is necessary. In *task dependence*, an actor depends on another to perform an activity, informing to the other what should be done, without needing to inform "the reason" to do. *Resource dependence* indicates us that an actor depends on another for the availability of a resource which could be something physical or informational, for the accomplishment of other activities in organisational environment. And the last dependence, the *softgoal* is a variant of the first; however the goal evaluation is quite subjective. In other words, the success condition is not defined a priori, and so, we could not be able to affirm the goal could be really satisfied. High performance, low cost, precision, among other are *i** softgoal examples.

3 Value for e-commerce

3.1 Electronic commerce

Electronic Commerce (e-commerce) differs from traditional commerce for the fact of using an electronic way for various stages of a trade. Jayaweera [2] defines e-commerce as the buying and selling of goods and services electronically by consumers or by companies via computerized transactions. Due to this characteristic, we verify this electronic way of trade has been accelerating the demand, the production, the delivery, and the payment for goods and services, and, at the same time, it has reduced marketing, operational, production and inventory costs in such a way that customer will benefit indirectly.

An e-commerce solution involves many areas of a company such as sales, purchases, marketing, and information technology. For selling products through the Internet, it is necessary thinking in the means to give products, services or information to the customers [1].

We have reminded the challenge for any (electronic) commerce application is to do the profitable business where the price for goods/services sold is higher than the production costs. We are able to do that by performing value adding activities at lower cost or performing them in a way that leads to differentiation from similar products so that customers will be ready to pay a premium price [2].

To include all of requirements presented here, we need elaborating a good e-commerce project, and it should start with organisational modelling, where all of the early requirements for the system to be constructed are elicited. System's organisational modelling should be used to develop any other necessary models for e-commerce solution. According to Jayaweera [2], when building e-commerce system, two types of models are fundamental: the business models and the process models.

In order to elicit early requirements, business model is just what matter, because its proposition is describing fundamental business aspects of the e-commerce system to be built. A business model describes which actors are involved, which the actors offer each other, and what activities they perform when producing and consuming offerings. The central concept in a business model is that of value, and the model describes how value is exchanged between actors [2]. Business model is also known as value model because those feature.

3.2 Value Concepts

Value concept is the main foundation for any commerce application, electronic or not. It has been analyzed extensively in economics and marketing literature [2]. Modelling value concepts, like goals, chains, activities, and exchanges, expose the company business strategies.

Consumer value is central for every successful marketing strategy in a market economy. The evaluation of some "object" by some "subject" is called consumer value. In a typical case, the "subject" could be the consumer while the "object" could be a product or a service offered by a company [2].

We find value chain definition as value's creative activities, since basic raw material sources, passing by components suppliers, until the final product is given to consumers hands. In company perspective, which intends finding its e-commerce solution, we see its participation in value chain starts in the suppliers' payment (purchases) until the delivering to consumers (sales). So, we know what matter to the company is maximizing the difference between purchases and sales [3].

Furthermore the concepts we exposed here, value theory is still done by other concepts. It comprehend since value model concepts until specific components related to the value's concepts, like actors, goals, value activities, value objects, value offerings, and value exchanges. We will see all of these concepts below.

3.2.1 Concepts related to the general model

Value Model: Shows how actors create, distribute, and consume objects of economic value. It captures decisions from different stakeholders. In other words, it captures decisions about "who" is offering and exchanging "what" with "whom", and expects "what" in return [5].

Value Chain: Shows how a value is successively added to products until stopping in a final consumer [2]. The value chain is intended to analyze competitive advantage by explaining cost leadership focus, or differentiation strategies. Using linkages between activities dependences between activities can be shown, for instance the way one activity is performed and a cost influence on another activity [3].

Value System: Each value system comprehends multiple companies, where each company in the system that has its own value chain [3]. So, we can say that value system is a set of many related value chains.

Value Viewpoint: Focuses on the (new) way of economic values creation, distribution and consumption. Its contribution to the evaluation of an e-commerce idea is a statement of revenues and expenses, caused by the exchange of valuable object between actors [5].

3.2.2 Concepts related to specific components

Actor: Enterprises (companies) or end consumers are examples of actors. It is perceived by his/her environment as an economically independent entity [5].

Market segment: Actors' group that attributes the same value to objects [5].

Composite Actor and Elementary Actor: (Actors specializations). It indicates when an actor is composed of other actors [5].

Actors Goal: Generally, actor goal is summarized in creating profit, or obtaining products or services that are of economic value for them [5].

Value Activity: Actors need performing value activities when they want exchanging objects of economic value with each other. These activities must yield profit or should increase economic value for the performing actor. There is interesting in the activity which has at least an actor (but hopefully more) believe that she/he can execute the activity profitable [3]. Value Activities can be decomposed into smaller activities. A value activity is executed by only one actor exactly, but each actor can execute more than one activity. The physical creation of the product, and its marketing and delivery to buyer, are some primitive value activities [2].

Objects of Economic Value: This object could be a service, a product, or even an experience, which is of economic value for at least one of the actors involved in a value model. Actors may value differently and subjectively, according to their own valuation preferences. For a value model, value object should be seen as a kind of value object which actors exchange, instead of the actual instance itself [5].

Mixed Bundling: It refers to the mechanism that an actor wants to offer value objects in combination rather than separately, because the actor supposes that different products sold in combination yield more profit than that if they were sold separately [5].

Value Object Instances: It is the reference to an instance of the value object exchanged by actors [5].

Value Offering: It models what an actor offers or requests from his/her environment. It models mixed bundling exchanges and individual objects, and shows the mechanism of Economic Reciprocity [5].

Economic Reciprocity: It refers to rational acting actors. It is supposed that actors are only willing to offer objects to someone else, if they receive adequate compensation in return [5].

Value Exchange: It is the relationship (link) between actors with a value object in the middle. It represents one or more potential trades of value object instances between value offers. The value exchange object instances is atomic, what ensures that if an actor offers something of value to someone else, he/she always gives in return what he/she wants. The value exchange does not represent the number of value exchange instances over time, nor their ordering in the time [5].

Value Transaction: Set of Value Exchanges. Sometimes, it is convenient having a concept that aggregates all value exchanges, which define the value exchange instances that must occur as consequence of how value exchanges are connected [5].

4 Applying Value Concepts to Organisational Modelling

Based on information of organisational aspects modelling and value concepts described in the previous sections, we present our value modelling proposed, which is resulted of organisational modelling together with value concepts.

4.1 Value Diagram

Value Diagram, we propose here, using the *i** technique, applied so much in representation/modelling of organisational aspects involved with processes. We intent extending the *i** framework joining value concepts to it.

The goal of this proposed diagram is doing a discovery of all information about values that can exist in e-commerce system to be elicited, with intention of documenting and verifying financial viability for the e-commerce solution, together with organisational aspects modelling found in the application domain. As final result of this Value Diagram's elaborating, we will have a Conceptual Value Model, referring to the elicited e-commerce system, where the involved actors Value Chain can be seen, as well as the System Value. It is still possible detaching, through the Value Diagram we propose, a lot of Value Viewpoints, whose objective is to indicate profits and costs caused by exchange of value objects among actors.

In value modelling, we use diagrams just based on the Strategic Rationale (SR) model [7] [8] of Organisational Modelling with *i**. These are the components making part of these diagrams: Actors (Elementary Actor, Composite Actor, and Market Segment), Actor Goals, Value Activities, Objects of Economic Value (isolated Objects or Mixed Bundling), Value Offerings and Value Exchanges (single Exchange and Transaction).

Beyond these components, the modelling includes the Softgoal, which is related to an actor and has direct connection with other actor. To this modelling, Softgoal has the same meaning of organisational modelling.

4.1.1 Actor

As in traditional organisational modelling [7] [8], actors are entities who perform actions in order to achieve goals in organisational environment context. The idea of Depender (actor who depends on other) and of Dependee (actor that helps or satisfies the Depender) continues. Figure 1 shows actors' examples.

There are three kinds of Actors: Elementary Actor, (common actor and represents a single entity), Composite Actor (the one that represents a group of elementary actors), and Market Segment (similar to Composite Actor; however, it contains actors' group that attribute the same value to objects, as if it was an elementary actor).

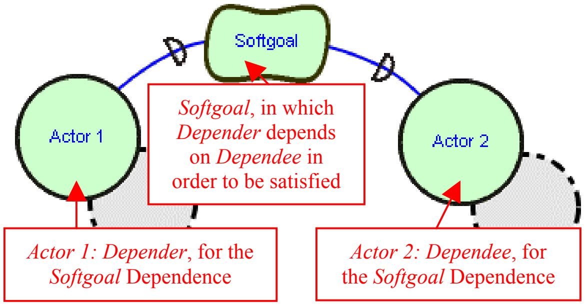

Softgoal
Actor 1
Actor 2
Softgoal, in which *Depender* depends on *Dependee* in order to be satisfied
Actor 1: Depender, for the *Softgoal* Dependence
Actor 2: Dependee, for the *Softgoal* Dependence

Figure 1. Actors' examples in the value modelling, with softgoal between them.

Therefore, the actor has the following properties: **Name**, its identification in the value model; **Description**, which is composed by a soon description about the actor; **Role**, identifying which role, if it is necessary, the actor plays in the value modelling context; **Type**, which indicates if the actor is an elementary one, a composite one, or a market segment; **Aggregated Actors**, listing the composed actors, in the case of dealing with a composite actor or a market segment; **Actor's Goals**, listing achieved goals for the actor; **Actor’s Activities**: listing value activities that the actor performs; **Softgoals and their priorities**, which consist in a list of softgoals, their priorities (lower, medium, or high), and what actor waits for this softgoal.

4.1.2 Goals

Goals for value modelling are similar to the ones of organisational modelling [7] [8], however they lead exclusively with value exchange among actors and they are called Actors Goals. Only *Dependers* contain these goals, in other words, goals should be “inside” of *Dependers* limits. But, it is important to emphasize that different actors goals can act in complement one each other. In the Figure 2, we see an example of actor goal.

Goals properties are **Name**, which identifies the goal in value model; **Description** which describes the goal in full detail; **Related Goals** which consists in a list of goals (and their actors) that are related with the goal; and **Goal’s Activities** which are list of value activities of the goal’s actor that are necessary for the actor achieving the goal, including the execution order, if it is necessary.

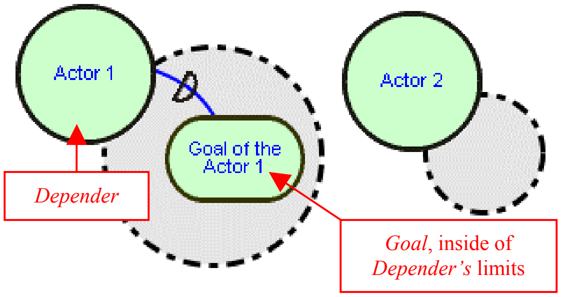


Figure 2. Actor's goal example inside of actor's limits, in value modelling.

4.1.3 Value activity

Tasks in organisational modelling are seen as value activities in value modelling, because they are tasks that lead directly with value exchange among actors. We can see a value activity example in Figure 3.

Activities are performed by actors and, thus, they should be profitable or they should increase the economic value for the actor that executes them [3]. So, we may affirm activities can contain information of how much the exchanged object’s value increases. Activities make part of an actor’s goal, as we see in actors' goal definition. Furthermore, activities can be decomposed in smaller ones [3].

According to what described previously, we define the following properties for the value activity: **Name**, which identifies the activity in the value model; **Description**, describing this activity in full detail; **Value Objects**, which list objects and their value increases that the activity refers to; **Sub-activities**, listing value activities that composes the activity, in case of sub-activities exist.

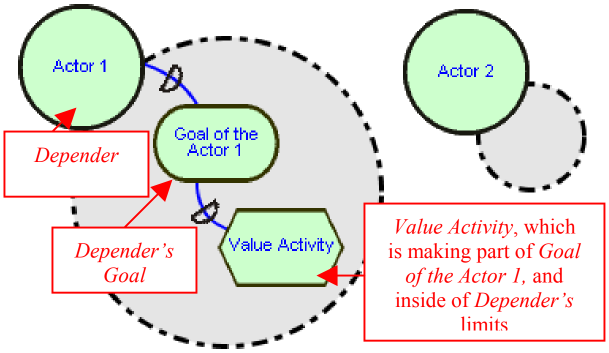

Actor 1
Actor 2
Goal of the Actor 1
Depender
Depender’s Goal
Value Activity
Value Activity, which is making part of *Goal of the Actor 1,* and inside of *Depender’s* limits

Figure 3. Value activity example, related to its goal and inside of actor's limits, in value modelling.

4.1.4 Object of Economic Value

In value modelling, one of the main concepts is the object of economic value, represented by the Resource element in *i** technique. Object of Economic Value is the resource of an actor who uses it to exchange with a value of another actor. Therefore, each actor in the value exchange should possess value objects. Because that, we put them inside of *Dependers* limits, as we see in object of economic value’s example of the Figure 4.

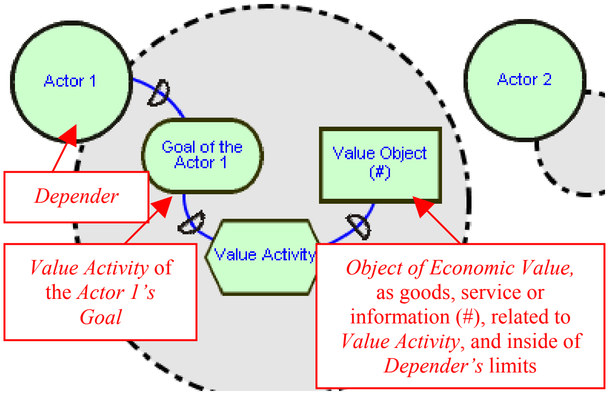

Actor 1
Actor 2
Goal of the Actor 1
Value Object (#)
Depender
Value Activity
Value Activity of the *Actor 1’s Goal*
Object of Economic Value, as goods, service or information (#), related to *Value Activity*, and inside of *Depender’s* limits

Figure 4. Object of economic value’ example, related to the value activity, and inside of the actor's limits, in value modelling.

Value objects are always related to value activities of an actor, which manipulates them. In value modelling proposed, we represented two kinds of values: those that represent monetary values, characterized by “($)” symbol, and those that represent objects (goods, services, or information) characterized by “(#)” symbol.

Mixed bundling is modeled as a common object, with the difference that a mixed bundling has their objects linked to him, according to the example in Figure 5.

Value object properties are **Name**, identifying it in value model; **Description**, which describes the object in full detail; **Kind of Value**, indicating if the object represents a monetary value ($) or an object (#); **Kind of Object**, which indicates if it represents an elementary object or a mixed bundling; **Initial Value**, indicating the object economic initial value in the value model, if possible; **Aggregated Objects**, listing objects that composes the mixed bundling, in that case.

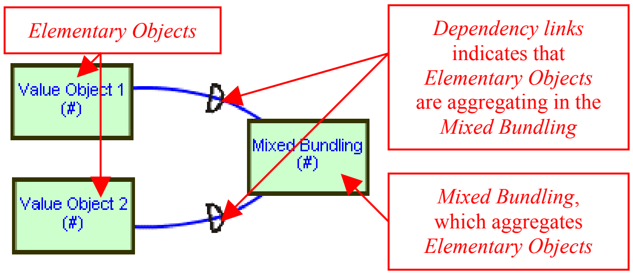

Elementary Objects
Dependency links indicates that Elementary Objects are aggregating in the Mixed Bundling
Value Object 1 (#)
Mixed Bundling (#)
Value Object 2 (#)
Mixed Bundling, which aggregates Elementary Objects

Figure 5. Elementary objects example aggregated by mixed bundling, in value modelling.

4.1.5 Value offering

In value modelling, value offering is represented by a group of relationships (dependency links in *i**) that leaves (out-going offerings) from a *Depender* actor and that enters (in-going offerings) into a *Dependee* actor. We see a value offering example in Figure 6.

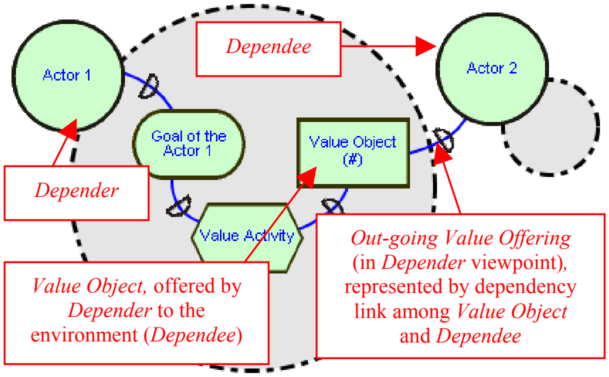

Dependee
Actor 1
Actor 2
Goal of the Actor 1
Value Object (#)
Depender
Value Activity
Value Object, offered by Depender to the environment (Dependee)
Out-going Value Offering (in Depender viewpoint), represented by dependency link among Value Object and Dependee

Figure 6. Value offering example in value modelling.

Value offering models what an actor offers to (an out-going offering) or request from (an in-going offering) his/her environment [5]. Therefore, value offering is each exit from an actor to environment (in other words, to another actor) or each entrance into an actor from the environment.

An actor is able to have only out-going offerings when the actor does not want anything in return. On the other hand, an actor also is able to have only in-going offerings when the actor does not wants paying for that was offered to him/her. However, the most common situation is actor having in-going and out-going offerings, what models the economic reciprocity.

Value offering properties are **Name**, that identifies the value offering in value modelling (we suggest using the "Depender-Object" model, being the actor of the out-going offering); **Description**, describing the value offering in full detail; **Actors and their views**, which indicate the actor who delivery value offering, and that one who receives value offering, with their respective offering views (if it is in-going or out-going); **Object**, indicating the offering's value object; **Reciprocity Offering**, which indicates what offering is reciprocal of the offering or from which offering this is reciprocal, when there is economic reciprocity.

4.1.6 Value exchange

Value exchange in value modelling does not have a specific component. It is modeled as a group of value object exchanged together with the object's relationships (links). In Figure 7, we show a value exchange example that is highlighted.

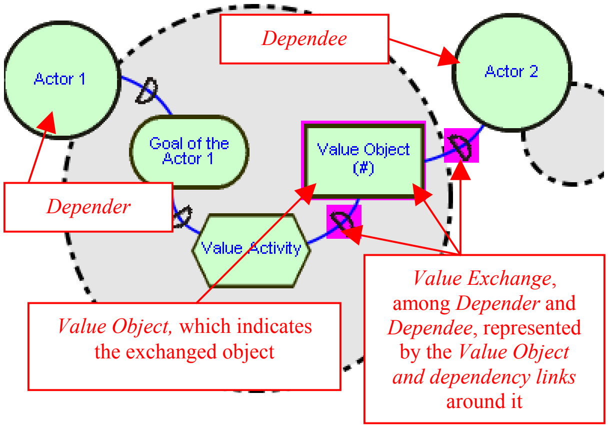

Dependee
Actor 1
Actor 2
Goal of the Actor 1
Value Object (#)
Depender
Value Activity
Value Object, which indicates the exchanged object
Value Exchange, among Depender and Dependee, represented by the Value Object and dependency links around it

Figure 7. Value exchange example among actors, in the value modelling.

Value exchange is an important point in value modelling, because it represents one or more possible trades of offered value objects from an actor to other. The value exchange relevant information consists of identifying which actors are exchanging a value object.

Value Exchange properties are **Name**, which identifies the value exchange in modelling (we suggest naming value exchange following the "Depender-Object-Dependee" model); **Description**, which describes the value exchange in full detail; **Exchanged Object**, indicating the value object of the exchange; **Origination Actor**, the *Depender* (the actor that gives the value exchange's object to other one); **Destination Actor**, the *Dependee* (the actor who receives the value exchange's object from other one); **Transaction Exchanges**, listing value exchanges belonging to transaction which this exchange is part of, case it is true.

4.1.7 Value transaction

As well as value exchange, value transaction in value modelling does not have a specific component. It is modeled as a group of value exchanges, where all of the exchanges in transaction are done successfully or none should be done. We demonstrate in Figure 8 an example of value transaction, whose value exchanges are highlighted.

The main information about value transaction consists of identifying which exchange does part of it, and the exchanges' order in that their exchanges should be done in this value transaction.

Value Transaction properties are **Name**, which identifies the value transaction in modelling; **Description**, describing the value transaction in full detail; **Transaction's Exchanges**, listing value exchanges, in the time order, belonging to the value transaction.

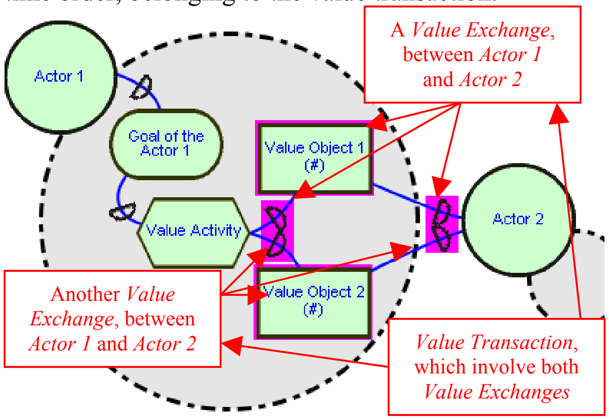

A *Value Exchange*, between *Actor 1* and *Actor 2*
Actor 1
Goal of the Actor 1
Value Object 1 (#)
Value Activity
Actor 2
Value Object 2 (#)
Another *Value Exchange*, between *Actor 1* and *Actor 2*
Value Transaction, which involve both *Value Exchanges*

Figure 8. Value transaction example, which contains a group of two value exchanges among actors, in value modelling.

4.2 Guidelines for Value Diagram

In order to aid requirements engineers by elaborating a Value Diagram, according to specifications defined previously, we suggest using the following group of guidelines:

GUIDELINE 1 (G1): Discovering Actors. Discover all of the actors involved in value model to be elicited, defining properties for each one, including softgoals. After that, put them into Value Diagram.

GUIDELINE 2 (G2): Identifying Actors' Goals. Identify all of the actors' goals for each actor, and fill out the properties of each goal. Put them into Value Diagram, inside actors limit, and connect (dependency link) actors with goals.

GUIDELINE 3 (G3): Defining Value Activities. Define all of the value activities to be executed for each actor goal and, consequently, fill out the properties for each defined activities. In Value Diagram, put value activities inside of the actors limit which performs them, and make the connections between those goals and the value activities.

GUIDELINE 4 (G4): Identifying Value Objects. Together with value activities, identify each value object involved in each actor activities, also defining the objects properties. In Value Diagram, put value objects inside of actors limits, and make the connection between value activities and value objects.

GUIDELINE 5 (G5): Identifying Value Offerings. Identify value offerings among of the actors in value modelling, and describe these offerings properties. Make connections, in Value Diagram, among the actor who offers (*Depender*) and the actor who receives the offering (*Dependee*), making the dependency link from each value object of that *Depender's* offering with *Dependee*. Value Diagram is completed after this guideline. The next guidelines are concluding value modelling.

GUIDELINE 6 (G6): Identifying Value Exchanges. After the Value Diagram is completed, it is necessary identifying all value exchanges among actors, and defining their properties. A suggestion is putting them in a table so that better the visualization.

GUIDELINE 7 (G7): Identifying Value Transactions. Identify the transactions that happen with value exchanges in modelling, defining the properties of each transaction, when they exists in value modelling.

In spite of these guidelines seem indicating a chronological order to be proceeded, we pointed out that the proposed Value Diagram's elaboration process can be done adopting the iterative and incremental software development principle.

4.3 Financial Viability

We can extract initial information for financial viability of the e-commerce solution. We just should make profitability sheets for each actor, where we sum actor outgoing objects values (expenses), sum actor ingoing objects values (revenues). So, if the difference between total revenues and total expenses is positive, the actor had profit; if it is negative, the actor had damage. More about profitability sheets generated from value models, we can found in [5], and about economic value concepts, in [3].

4.4 Example

For demonstration of value modelling application, we present an e-commerce value modelling example.

In this modelling, only two actors are involved: Consumer and Company. Consumer has intention of buying products, and Company, of selling products. Both execute only a task. Company has the product which the Consumer wants to acquire, will give a gift together with the product sold to him. Consumer has money for making the payment. Finally, Company hopes getting the Consumer fidelity, so that Consumer continues acquiring products. The main company strategy is offering a gift in order to get the consumer fidelity. The complete Value Diagram for this modelling is in Figure 9.

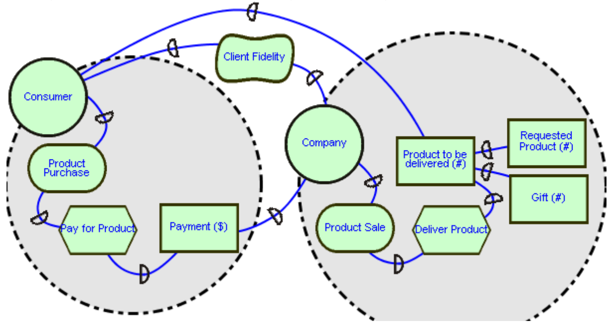


Figure 9. Completed modelling of the Conceptual Value Model for the Example.

We will show how elaborate this Model of Conceptual Value, using guidelines defined in previous section.

G1 – Discovering Actors. There are just two actors in this value modelling: "Consumer" and "Company". Company actor has the properties below in Table 1.

Properties	Description
Name	Company
Description	Company which is owner of the e-commerce system, and it has the interest of selling products to Consumers
Role	Salesperson of products
Type	Elementary actor
Aggregated actors	There are no aggregated actors defined for elementary actor
Actor goals	1. Product Sale
Actor activities	1. Deliver Product
Softgoals and their priorities	1. Client Fidelity (High) – Consumer

Table 1. Company actor's properties.

G2 – Identifying Actors' Goals. For this modelling, each actor has one goal: "Consumer" wants making "Product Purchase", and "Company" intends to achieve "Product Sale". We remember the Product Purchase (Consumer's goal) is complement to the Product Sale (Company's goal), and vice-versa. Product Purchase goal's properties are in Table 2.

Properties	Description
Name	Product Purchase
Description	Products purchase is the main interest of this market segment, that intends using the Company's e-commerce system to acquire products
Related goals	1. Product Sale (Company)
Goal's Activities	1. Pay for product

Table 2. Purchase goal's properties.

G3 – Defining Value Activities. In order to achieve the goal of "Consumer" by doing "Product Purchase", Consumer needs performing a value activity called "Pay for Product", while for "Company", in order to accomplish "Product Sale", it is necessary to "Deliver Product". Pay for Product value activity's properties are seen in Table 3.

Properties	Description
Name	Pay for Product
Description	Consumer needs doing the product payment according to what is defined in e-commerce system (product value, delivery fee, payment way, and so on)
Value Objects	1. Payment – 20% more than the product value for Company
Sub-activities	There are no sub-activities for this activity

Table 3. Pay for Product value activity's properties.

G4 – Identifying Value Objects. Actors' Objects for this modelling are simple: "Consumer" has money for "Payment ($)", and "Company" has "Requested Product (#)". However, "Company" decides that, in product purchase, it should be sent a "Gift (#)" together with "Requested Product (#)", completing the "Product to be delivered (#)" in order to satisfy "Consumer" and try obtaining the "Client Fidelity". Properties for the Product to be delivered value object are in Table 4.

Properties	Description
Name	Product to be delivered
Description	Product which should be delivered to Consumer who acquiring it. It including requested product and a gift
Kind of Value	Object (#)
Kind of Object	Mixed Bundling
Initial Value	$ 18.00
Aggregated Objects	1. Requested Product; 2. Gift

Table 4. Product to be delivered value object's properties.

G5 – Identifying Value Offerings. There are just two offerings in this value modelling: Company offers product that is "Company-Product to be delivered" offering and Consumer offers payment for product that is represented by "Consumer-Payment" offering. Properties for Consumer-Payment value offering are in Table 5.

Properties	Description
Name	Consumer-Payment
Description	It means payment offering Consumer makes to Company in order to acquire Product
Actors and their views	1. Consumer (Out-going Offering); 2. Company (In-going Offering)
Object	Payment
Reciprocity Offering	Company-Product to be delivered

Table 5. Consumer-Payment value offering's properties.

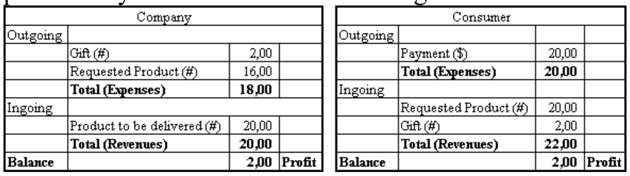
G6 – Identifying Value Exchanges. Two value exchanges are seen in this modelling. They are the exchange in that Consumer will give the bought product's payment to Company, called “Consumer-Payment-Company”, and other exchange in that Company will deliver bought product to Consumer, called “Company-Product to be delivered-Consumer”. The value exchange Consumer-Payment-Company properties are in Table 6.

Properties	Description
Name	Consumer-Payment-Company
Description	Value exchange of product payment, to be sold by Company to Consumer
Exchanged Object	Payment
Origination Actor	Consumer
Destination Actor	Company
Transaction Exchanges	This value exchange does not belong to any value transaction

Table 6. Consumer-Payment-Company value exchange’s properties.

G7 – Identifying Value Transactions. There are no transactions in this value modelling.

About the initial financial viability for our example, we could conclude that both actors have had profit: Company spent $18.00 ($16.00 for Product plus $2.00 for Gift). Consumer spent $20.00 (Payment) and received the Requested Product (now for $20.00, because the additional value), and the Gift ($2.00). We can see the profitability sheets for each actor in Figure 10.



Company			
Outgoing			
	Gift (#)	2,00	
	Requested Product (#)	16,00	
	Total (Expenses)	**18,00**	
Ingoing			
	Product to be delivered (#)	20,00	
	Total (Revenues)	**20,00**	
Balance		**2,00**	**Profit**

Consumer			
Outgoing			
	Payment ($)	20,00	
	Total (Expenses)	**20,00**	
Ingoing			
	Requested Product (#)	20,00	
	Gift (#)	2,00	
	Total (Revenues)	**22,00**	
Balance		**2,00**	**Profit**

Figure 10. Profitability sheets, which show Initial Information for Viability to the Example.

5 Conclusions and Future Work

We have presented, in this work, a proposal of applying value concepts to organisational modelling for early requirements elicitation.

Integrating Organisational Modelling to Value Theory, we can obtain a more complete elicitation of early requirements for stakeholders who will participate in an e-commerce system elaboration, because it includes several fundamental value concepts for such systems.

We believe that we could give a contribution when we have defining a value diagram’s model, including each component properties of referred diagram. Furthermore, we also define a group of guidelines to be used, aiding requirements engineers in the Model of Conceptual Value development for e-commerce system to be elicited.

When we compare this work with others, those related to organisational modelling as well as those related to value theory, we emphasized this work has advantage of approaching two stages for requirements elicitation in e-commerce solutions as they were one: one stage, for eliciting organisational aspects, and another stage, for exposing system’s financial viability aspects. At the same time we make organisational aspects modelling like goals and dependences among actors, related to organisational modelling, we also capture information about elicited e-commerce system’s financial viability, like value exchanges, value activities, and value offerings. Doing these both stages together, we believed the early requirements elicitation becomes more complete and specific for e-commerce systems.

As future work, we guess would be important implementing all of properties described in value modelling, as well as implementing a way to model value exchanges and value transactions in *i** technique. Furthermore, integration of early requirements, related to value concepts, with other phases in a software project can also be developed. Although, we believe, deepening studies about value theory, it is possible extending the components’ properties, and defining ontology in order to include more information related to value concepts and organisational modelling, becoming requirements elicitation better informative to all involved stakeholders.

References

[1] D. Marques. *Introduction to e-commerce*. Papal Catholic University of Minas Gerais, Poços de Caldas’ Campus, 2001.

[2] P. Jayaweera. *Methodology to Generate e-commerce Systems: Process Pattern Perspective (P3)*. Stockholm University and Royal Institute of Technology, 2002.

[3] M. E. Porter. *Competitive Advantages. Creating and Sustaining Superior Performance*. The Free Press, 1998.

[4] V. Santander. *Integrating Organisational Modelling with Functional Modelling*. Federal University of Pernambuco, 2002.

[5] J. Gordijn. *Value-based Requirements Engineering: Exploring innovative e-commerce idea*. Vrije University, 2002.

[6] F. Carvalho. *Organisational Modelling and Knowledge Administration: The State University of Bahia Southwest Case*. Federal University of Pernambuco, 2003.

[7] E. Yu. “*Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering”*. Proceedings of the 3rd IEEE Int. Symp. on Requirements Engineering (RE'97), 1997.

[8] E. Yu, L. Liu, and Y. Li. *Modelling Strategic Actor Relationship to Support Intellectual Property Management*. 20th International Conference on Conceptual Modelling (ER-2001), 2001.

[9] A. Antón, J. H. Dempster, and D. F. Siege. *Deriving Goals from Use-Case Based Requirements Specification for an Electronic Commerce System*. North Caroline State University, 2000.

[10] L. E. Martins. *A Software Requirements Elicitation Methodology Based on the Activity Theory*. State University of Campinas, 2000.

Measuring the Business / System Alignment

Anne Etien, Colette Rolland, Camille Salinesi
Centre de Recherche en Informatique, Université Paris 1 – Panthéon Sorbonne
90, rue de Tolbiac – 75013 Paris – France
aetien@univ-paris1.fr, rolland@unv-paris1.fr, camille@univ-paris1.fr

Abstract

It is widely acknowledged that organisations gain in performance when the system functionality captured in a system model matches organisational requirements available in the business model. However, few approaches help to determine if the system model and the business model are aligned and to which extent. The paper proposes criteria and associated generic metrics to quantify to which extent there is a fit between the business and the system which supports it. The definition and use of the metrics is part of a larger approach dealing with alignment in face of change, in a requirements engineering perspective. In order to illustrate the use of the proposed generic metrics we present them for two specific models, the MAP and UML class and transition diagrams. We then illustrate the use of the specific metrics in a case study.

1. Introduction

Aligning information systems (IS) to business processes (BP) that they support, is equally considered important by researchers [1], [2] and by professionals [3]. Recent field surveys also seem to demonstrate this importance. For example, a 2001 study conducted in 226 companies [4] clearly proofs that alignment of IS to BP significantly improve business performance.

If alignment clearly appears as desirable, some issues still remain unsolved as for example: (i) the achievement of BP-IS alignment, (ii) its management over time, (iii) the identification of non fit and (iv) the evaluation of this latter. Indeed, [5] considers that insufficient researches have been conducted to deal with the two first issues. P. Soffer [6] highlights that, because business processes changes are often not immediately followed by adjustments in the system, it is important to manage the identification and the evaluation of non fit. According to her, this would help in making decisions and undertaking adjustments.

In this paper, we try to bring solution to the three last issues.

P. Soffer [6] suggests that identifying the non fit requires the application of fit measurement methods. Measurement is indeed a way to avoid a subjective evaluation of the degree of alignment. Our concern is in line with this topic and we defined a set of alignment metrics. These metrics use the alignment definition of Nadler: “alignment is the degree to which the needs, demands, goals, objectives and/or structure of one component are consistent with the needs, demands, goals, objectives and/or structure of another component” [7]. The measurement of BP/IS fit is therefore, based on the degree of correspondence between business model and system model components. We propose the use of alignment criteria and associated metrics to measure these.

There exist different ways to manage alignment faced to change. One of this is called co-evolution, by analogy with the discipline of biology (the co-evolution) which studies interacting species, i.e. species that influence each other’s evolution. Researches have been achieved in computer sciences to analyse the reciprocal evolution of systems or software and other entities such as organisations [8], business processes [9], or environment [10]. In the context of information system engineering, co-evolution can occur differently according for example, the dependence relationship that can exist between the two evolving entities or the way to verify alignment [11].

We are developing a co-evolution approach using a requirement engineering (RE) perspective. RE can be seen as a way to establish a relationship between the “why” and the “what” of the system under development [12], [13]. The latter deals with the system functionality whereas the former provides its rationale. Our co-evolution approach is in line with the RE goal-driven approaches where the IS and the BP are represented by a unique language to express stakeholders and users intentions.

In this paper we introduce the co-evolution approach and the use of the alignment metrics. The rest of this paper is organised as follows. Section 2 provides an overview of our co-evolution approach. Section 3 presents the set of alignment metrics. Section 4 defines the metrics for two specific models: the MAP and the UML class and transition diagrams. Section 5 illustrates the use of metrics on a case study. Finally, discussion on the approach is provided in section 6. Conclusions are drawn in section 7.

2. Overview of the co-evolution approach

We identify two major classes of approaches of system modifications. On the one hand, the initial system modifications can generate other ones, which are discovered through impact analysis [14]. On the other hand, once the system has evolved, a conceptual mismatch [15] can appear between the system and another entity such as its architectural environment or business processes, hence reducing the performance of the organisation [5]. This conceptual mismatch requires the other entity adaptation in order to re-establish the alignment [9]. Such successive evolutions can obviously occur the other way round, which again causes system evolution.

Our approach follows the second paradigm. It adopts the change handling view in which change creates a movement from an existing situation captured in the *As-Is model* to a new one captured in the *To-Be model* [15]. This approach has furthermore two particularities: (i) it relies on the existence of an intentional pivot model to represent together the system and the process and (ii) it uses gaps to explicitly express change requirements.

Figure 1 presents our approach organised in three steps.

The first steps aims to realise the system / business alignment and to construct the pivot model. It uses alignment metrics to determine to which extent system and business are aligned and to identify eventual corrective actions. The second step corresponds to the evolution of the pivot model and relies on the identification of gaps expressing requirements changed [16]. The third step corresponds to the propagation on the system and business models of the gaps identified on the pivot model. This part of the process uses traceability of the relationship links between elements of the pivot model and those of the process model (respectively system model).

The rest of this paper presents the alignment metrics and illustrate through an example how they can help to evaluate the system / business fit.

3. The alignment metrics

We define ten metrics corresponding to different facets of the alignment relationship. In order to facilitate the metrics definition to various models, we proceed in two steps (1) the definition of the metrics at a generic level and (2) their adaptation to specific models. The specific metrics allow to calculate values determining in a particular project, at a particular instant in which extent BP and IS are aligned as shown in Figure 2.

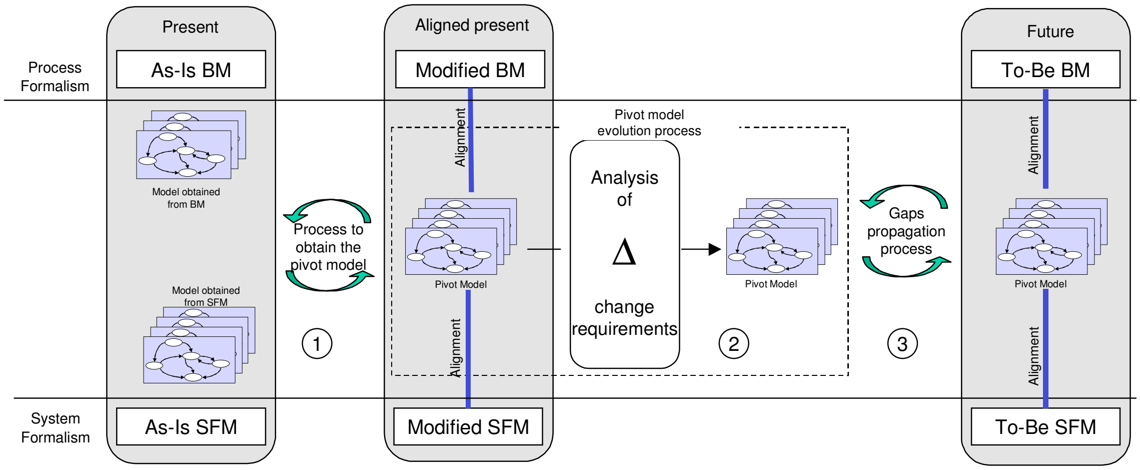


Figure 1. Schema of our co-evolution approach

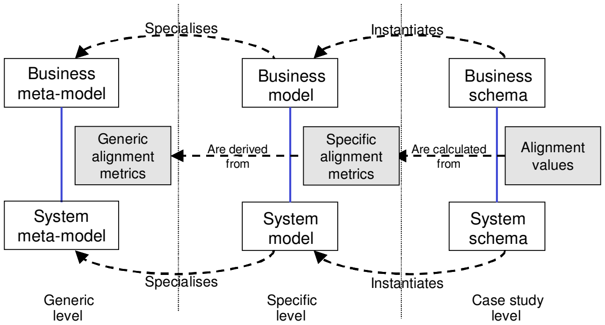


Figure 2. Generating specific alignment metrics

The generic metrics are defined between constructs of meta-models inspired from (a) the Bunge-Wand-Weber (BWW) ontology [17], [18] and (b) the Soffer-Wand (SW) ontology [19]. These two ontologies are adaptations of Bunge's ontology [20], [21]. They respectively represent the system and the business.

Table 1 summarizes the constructs used as components of the fit relationship.

	Constructs get from the ontologies	Definition
Constructs common to the system and the business	Class	A Class is a set of things that can be defined via their possessing a single common property.
	Property	A Property characterises a class
	State	A state is corresponds to the set of values of all attribute functions of a class
	Unstable state	An unstable state is a state that must change
	Stable State	A stable state is a state that can only change as a result of an action of something outside the domain
	Law	A law is a function from the set of states to itself
Constructs peculiar to the system	Event	An event is an ordered pair of states, where one state precedes a transformation and the other succeeds the transformation
Constructs peculiar to the business	Actor	An actor is a class that takes actions in response to their state changes
	Resource	A resource is a class that takes no further action
	Process	A process is a sequence of unstable states leading to a stable state, the goal.
	Goal	A goal is a set of stable states
	Activities	An activity is a state transitions caused by transformation

Table 1. Main constructs used to represent the system and the business at the generic level

Two types of links, namely *maps* and *represents*, have been identified in order to establish a construct correspondence across the constructs of the SW and BWW meta-models. The former link expresses equality between SW and BWW identical constructs. The latter specifies that a BWW construct has an impact on a SW construct. Thus, two constructs of a different nature, for example a BWW class and a SW property or a BWW event and a SW activity can be linked through a *represents* link.

The metrics at a generic level are defined from the constructs of these meta-models and these links.

The generation of the specific alignment metrics from the generic metrics relies on (i) the establishment of the relationship between constructs of the chosen business model (respectively system model) with those of the SW (respectively BWW) meta-model and (ii) the adaptation of the generic metrics.

There are a number of advantages of proceeding in this way, (1) the generic metrics are based on a solid theoretical ground provided by the Bunge's ontology (2) the generic metrics serve as a guide to define the specific ones: the latter are just a specialisation of the former, (3) the process of producing the specific metrics is easier and less error prone and, (4) specific sets of alignment metrics are consistent with each other as they are generated from the same mould and this facilitates comparisons across methods.

The rest of the section successively presents the ten metrics.

3.1. Overview of the alignment measurement system

In order to structure the set of the metrics, we use the Cavano and McCall framework [22]. This framework of software quality measures comprises three levels, *factors, criteria* and *metrics*. Factors are characteristics that can be appreciated on an external point of view. They can be used as an aid in specifying software quality objectives. These high level factors are then broken down into criteria that are more software directed. Criteria correspond to product characteristics and represent internal and technical viewpoints. Metrics allow to measure a criterion. The Cavano and McCall framework is a reference in the domain of the software quality measure and has inspired a number of other frameworks as those of Boehm [23], the ISO quality standard [24]…

In our framework, we identify four factors along which the alignment can be measured namely, the *intentional factor*, the *informational factor, the functional factor,* and the *dynamic factor*. These factors reflect the four perspectives that have been reported in IS literature, namely, the holistic view brought by the goal-actor-resource-process perspective; the information perspective; the functional perspective; and the dynamic perspective. They can be used to aid in specifying alignment objectives. Each factor has associated *criteria* corresponding to alignment characteristics. They are in turn, related to *metrics* that allow the actual computation of the degree of alignment.

Table 2 sums up our alignment measurement system. As shown in the table it comprises ten criteria and ten metrics grouped along the four identified factors.

Factors	Criteria	Metrics
Intentional Fit	Support Ratio	Activity representation count
	Goal Satisfaction	Goal mapping count
	Actor Presence	Actor mapping count
	Resource Presence	Resource mapping count
Informational Fit	Information Completeness	Business / System class mapping count
	Information Accuracy	Business / System state mapping count
Functional Fit	Activity Completeness	Business / System class mapping count
	Activity Accuracy	Business / System state mapping count
Dynamic Fit	System Reliability	Law mapping count
	Dynamic Realism	Path mapping count

Table 2. Framework of alignment metrics

In the rest of this section, we describe in detail the factors, criteria and metrics.

3.2. The intentional factor

Along the intentional dimension, the objective is to measure to which extent the system is meeting the business requirements. This is achieved by providing four criteria associated to the *intentional factor* dealing, respectively, with the business activity and the goal support, and the actor and resource representation.

3.2.1. Support ratio. *The support ratio*, similar to the technological coverage criterion defined by Bodhuin [25] is the extent to which business activities are supported by the system. The higher this ratio is, the more automated the activities are. Conversely, a low support ratio expresses that a large number of business activities are manually carried out. In order to raise this ratio it is necessary to automate some activities and thus, to introduce in the system new classes, properties, laws, events to support these business activities.

3.2.2. Goal satisfaction. Clearly, evaluating to which degree the system meets the business goals is an essential part of the alignment measurement.

The *goal satisfaction criterion* is proposed to measure the extent to which goals are supported by the software system. The *goal satisfaction metric* is defined as the ratio of goals for which each state *maps* a system state. A low goal satisfaction ratio indicates that a number of goals cannot be achieved with the use of the system. On the contrary, a high ratio shows that the system fits the need for business goal satisfaction.

3.2.3. Actor presence. An actor is a class that triggers a state transition on another class. For example, a client that makes an order in a case of an e-commerce application on the Internet is an *actor*. It provokes the creation of an order (and thus a state change of the class order). However, some of its properties can be used in another process as for example, his/her address for delivery. It seems important to check that business actors are present in the system in order (1) to trigger state transitions and (2) to permit the use of their properties in the system. This is the role of the *actor presence criterion.* It calculates the proportion of business actors that *maps* a system class compared to the total number of business actors.

A low value of this criterion means that the causes of some state transitions are not the same in the business and in the system. In order to increase its value, it is possible (i) to remove the business actor who is not present in the system if the check demonstrates that he does play a significant role in the business process or, (ii) to introduce the actor in the system.

3.2.4 Resource presence. Resources are classes that are used during an activity but do not take action. For example, in an ATM project, a credit card is a business resource. In order to memorize some properties of the card such as the credit card number and the validity date it is necessary to have a class in the system, which corresponds to the credit card resource. The *resource presence criterion* compares the number of resources that map a system class to the total number of business resources.

A low value shows that numerous business resources are not supported by the system. This means that either some resources play a role in the business which does not need to be known by the system or that the correspondence of some of them in the system is missing. In contrast, a high value demonstrates that the majority of the business resources are known by the system. No indication is given relatively to the way they are used.

3.3 Informational factor

The *intentional factor* and its related criteria defined above are focusing on the evaluation of the degree to which business activities are supported by the system and the extent to which the system allows business requirements fulfilment. The *informational factor* which we consider now, complements the former by a deeper analysis of the way activities are supported in the system. In order to provide a good alignment between the system and the business processes, the system must (i) manipulate all the business process classes and (ii) support all the business process states. Two criteria have been defined in order to permit such an evaluation, the *informational completeness* and the *informational accuracy*, respectively.

3.3.1. Informational completeness. Obviously, to avoid a poor alignment between the business and the system there should be a strong correspondence between the business information and the system information. This can be measured by the fact that there exist classes in the system that *map* business ones. The *informational completeness criterion* measures the proportion of this correspondence.

A high value computed by the metric shows that the information system really meets its purpose as it manages the information needed to support business processes. This value might diminish if for example, new classes are introduced in the business. Corrective action should be taken to increase the *information completeness* ratio either by modifying the system to manipulate the new classes or by removing business classes if they demonstrate to be of low value.

3.3.2. Informational accuracy. Clearly it is important that the business process and the supporting system manage the same information, but it is not sufficient. Intuitively it seems that the information has to be managed in the "same way" at the business process and system levels. The "same way" means for us, that the states of the business correspond to states in the system. Thus for example, in the case of order management, it is necessary for the system to recognize the state "satisfied" of an order because events and thus activities depend on this state. If such a state does not exist in the system, the order can, for example, not be delivered, the payment not be asked for and the stock not be managed.

A low informational completeness ratio implies low informational accuracy because if a business class is not supported by the system then neither are its states. However, the contrary is not true. Business classes can correspond to system classes without a similar correspondence for their states. Such a situation detected due to the information accuracy measure implies elicitation of change requirements to raise this measure.

3.4. Functional factor

The *functional factor* aims to measure the degree to which activities in the system correspond to business activities. The correspondence is based on involvement of classes and their states in business and system activities, respectively. The functional factor has two criteria associated to it, *activity completeness* (Ac) and *activity accuracy* (Aa). Both apply to each individual activity of a business process, separately.

These criteria allow to enter into the details of each business activity. They express the degree to which a given business activity is supported by the system by measuring the supported classes and states involved in the activity, respectively. Misfit, according to the *activity completeness* criterion, requires checking actions to verify if the lack of completeness is justified whereas a low value computed by the activity accuracy metric signals some inconsistency in the representation of the activity in system terms that requires to be corrected.

3.5. Dynamic factor

The fourth factor, the *dynamic factor* aims to evaluate the extent to which the dynamicity of business processes is reflected in paths of system state transitions. It has two criteria namely the *system reliability criterion* and the *dynamic realism criterion*.

3.5.1. System Reliability. The two previous factors take into consideration classes and/or the states independently as well as separately. The *system reliability criterion* considers the states and transitions between these states to check their fit with the corresponding business states and transitions. The alignment is based on the *maps* relationship between these. Indeed an absence of *mapping* reveals errors. Assume for example, that in a supply chain process orders are paid only if their state is 'satisfied'. Even if using previous measures it is known that the system manages orders, knows the states 'satisfied' and 'paid' for an order, nothing guarantees that the system respects the state constraint for triggering the order payment.

The *system reliability* criterion is defined to measure the alignment between business laws and systems laws. The associated metric computes the ratio of business laws that are supported in the system.

A low value of the *system reliability criterion* shows that the system will not behave as a mirror of the business performance as business laws poorly correspond to system laws. Such a situation is critical and calls for corrective design actions.

3.5.2. Dynamic realism. Complementarily to measuring the alignment of business and system laws, it is important to evaluate if the succession of business activities is adequately supported by the system. This leads to look for *mapping* between a path of business activities and a path of system state transitions. The *dynamic realism* criterion is defined for this purpose.

A low level of the value computed by the metric shows that the system does not provide what is expected at the business level. To increase this value, either the business model has to be modified keeping the system constant or vice versa. Besides, it shall be noticed that the reason for a low value might be low values of other criteria. Corrective actions should examine the latter before considering the former.

4. Definition of the metrics for the MAP and UML class and transition diagrams

This section illustrates the use of metrics. These metrics are associated to constructs of two specific models, namely the MAP and the UML class and transition diagrams to respectively design the business and the system. Business processes and Information systems are not always described with UML diagrams. They are designed with their own formalism. The choice of these two specific models relies on their usability but also their diversity and allows to show that specific metrics can be generated even if the models are very different.

MAP model has been chosen to design the business processes. It is an intentional process model [26]. It provides a representation mechanism based on a flexible ordering of intentions and strategies. As shown in Figure 3, a map is a labelled directed graph with *intentions* as nodes and *strategies* as edges between intentions. The directed nature of the graph shows which intentions can follow which one. An edge enters a node if its strategy can be used to achieve the intention of the node. Since, there can be multiple edges entering a node, the map is capable of representing the many strategies that can be used for achieving an intention.

The key concepts of MAP are *intentions* (goals to achieve or maintain), *strategies* (means or manners to attain a goal) and *sections* which are triplets $<I_i,I_j,S_{ij}>$ where I_i is the source intention, I_j the target intention and S_{ij} the strategy to attain when I_i has been achieved. The map of Figure 3 contains for example, 5 intentions (including *Start* and *Stop* which exist in any map to start and stop the process, respectively), 6 strategies and 6 sections MS0 to MS5. MAP includes a refinement mechanism by which a section in a map can be modelled as a map in its own. This leads to the representation of a business as a hierarchy of maps.

Formally, an intention I is defined as a set of *desirable states* G_I and every section has an *initial condition* and a *final condition*, both expressed in terms of states. A section S from intention I to intention J starts in a subset of states $I_S \subseteq G_I$ and ends on a subset of states $F_S \subseteq G_J$.

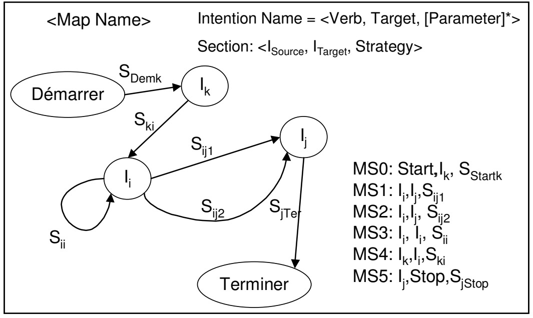

<Map Name>
Intention Name = <Verb, Target, [Parameter]*>
Section: <I_{Source}, I_{Target}, Strategy>
S_{Demk}
Démarrer
I_k
S_{ki}
S_{ij1}
I_j
I_i
S_{ij2}
S_{jTer}
S_{ii}
Terminer
MS0: Start,I_k, S_{Startk}
MS1: I_i,I_j,S_{ij1}
MS2: I_i,I_j, S_{ij2}
MS3: I_i, I_i, S_{ii}
MS4: I_k,I_i,S_{ki}
MS5: I_j,Stop,S_{jStop}

Figure 3. A map

To design the system, we chose the UML class diagram and UML state transition diagram. Associating these two diagrams allows us to describe a system from both a static and a dynamic perspectives. The static part provides description of durable links between instances of constructs (i.e. composition) whereas the dynamic part models behavioural interactions inside the system or with its environment (i.e. in response to events).

We select the support ratio to show the way metrics are formulated. At the specific level, the *support ratio* is established between the number of *map sections* ***represented*** by *events* and the total number of *map sections*. It is defined as follows:

Let:

- S_b be the set of business sections card(S_b) = the number of elements contained in S_b.
- E_s be the set of system events
- S_b^r be the set of business sections for which it exists event representing them;

$S_b^r = \{s, s \in S_b \mid \exists\, e \in E_s \wedge e \Re s\}$ and card(S_b^r) = number of elements contained in S_b^r

$$Sr = card(S_b^r) / card\ (S_b)$$

The nine other metrics can be found in [27].

Table 3 sums up the ten specific criteria and metrics.

Criteria	MAP constructs	UML constructs	Comments
Support Ratio	Section	Event	Number of *sections* **represented** by *events* / number of *sections*
Goal Satisfaction	Intention	State	Number of *intentions* for which each state **maps** a *state* in the system / Number of *intentions*
Actor Presence	Actor	Class	Number of business *actors* **mapping** a system *class* / Number of business *actors*
Resource Presence	Resource	Class	Number of business *resources* **mapping** a system *class* / Number of business *resources*
Information Completeness	Object	Class	Number of business *objects* **mapping** system *class* / Number of business *objects*
Information Accuracy	State	State	Number of business *states* **mapping** to system *states* / Number of business *states*
Activity Completeness	Object	Class	Same as Information Completeness but for one given section
Activity Accuracy	State	State	Same as Information Accuracy but for one given section
System Reliability	Law	State	Number of business laws for which each business *state* **maps** a system *state* and the transformation between business states are possible between system states / Number of business *laws*
Dynamic Realism	Path	State	the succession of these system states is possible / Number of possible paths

Table 3. Specific metrics for measuring business/ system fit modelled in MAP and UML terms

5. Illustrating the use of metrics for a hotel room booking system

In this section, we illustrate the usage of the specific fitness measurement system in a hotel room booking case study. This example is a largely used and recognised textbook case. However, our approach has been used on industrial project as described in [27]

5.1. Overview of the case-study

The example presented in this section is partially fictions. Let assume that several small hotels made the decision, several years ago, to become partners in order to try to resist to the competition with international hotel chains. This partnership led to the construction of business processes and an associated IS support system. Stakeholders want to analyse the actual situation in order to understand why their performance is not as good as they expect. We suggest to quantitatively evaluate the alignment between their business and IS.

The business process is divided into two parts corresponding to (i) the creation and the management of the product list and (ii) the management of the room booking contracts.

These objectives are reflected in the map presented in Figure 4. The map comprises two intentions: "*Offer Product*" and "*Manage booking contracts*". There are a number of strategies associated with each of these two intentions, particularly with the "*Manage booking contracts"*. The "*by offering booking facilities"* strategy for example, is a cluster showing that booking can be done on the spot or through an agency. The process terminates (i) with a product removal or (ii) the payment and the product (eventually partial) consumption or (iii) the cancellation of the booking.

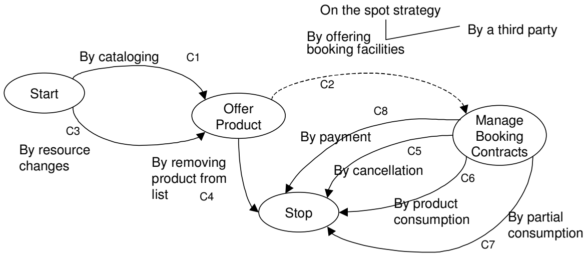


Figure 4. The room booking business as a map

In order to get a complete understanding of business intentions and strategies, it was necessary do refine a number of sections of the above map. In total, there are five maps organised at two levels of abstraction. The complete specification of the business includes 39 sections, 3 actors (the hotel keeper, the agent and the client), 8 objects (Hotel, Room Availability, Demand, Booking, Resort, Room, Temporal_Hotel_Closing, Temporal_Resort_Closing) and 23 states.

5.2. Measuring the alignment

From the models of the business and those of the system, we identified each instance of the concepts involved in the different metrics. Thus, for example, 'client' and 'booking' are Map business objects, "manage booking contracts by offering booking facilities" is an activity. We proceeded in this way for all elements. As a result, the process is composed of activities; goals are defined in terms of objects states, etc… The metrics can then be applied based on their definitions. For instance, for the information completeness metric, we count the total number of business objects and compare it to the business objects *mapping* a system class. In our case study, there are 8 business objects. However, only four *maps* a system class. Indeed, 'Resort' and 'Room' are managed in the business as business objects and are considered in the system as hotel properties and not as classes. Furthermore, the system does not handle temporal closing. As a consequence, the information completeness equals 4/8.

On the 39 sections of the maps only 32 are *represented* by system events. Indeed, the system does not make distinction between temporal and definitive closing. It does not manage room and resort creation or modification and so on. The support rate thus equals 32/39.

Furthermore, some business states do not *map* any system state as the states 'partially consumed' for the booking. This impacts several criteria as the Information Accuracy or the Dynamic Realism. A detailed study of this latter allows to realise that only 29 among the 51 business processes are implemented in the system. A similar approach is used to calculate the other values of Table 4.

Table 4 presents the ten measures (notice that for sake of space, the metrics of the functional factor have been calculated only for the booking creation activity).

Criteria	Measure
Support Ratio	0,82 (32/39)
Goal Satisfaction	0,5 (3/6)
Actor Presence	1 (3/3)
Resource Presence	0,6 (3/5)
Information Completeness	0,5 (4/8)
Information Accuracy	0,65 (15/23)
Activity Completeness	1 (3/3)
Activity Accuracy	1 (5/5)
System Reliability	0,70 (19/27)
Dynamic Realism	0,57 (29/51)

Table 4. Alignment measures

A global study of Table 4 shows that the IS and the business are not perfectly aligned as some measures are inferior to 1. The support ratio is relatively high with about 82% of the business activity being supported by the system. However, other measures allow to refine this view. The goal satisfaction measure is 0,5 as well as the information completeness.

Stakeholders have to take corrective actions in order to raise the proportion of satisfied goals. The goal satisfaction metric relies on states mapping. Thus, the high number of business objects (and thus of business states) not supported by the system can be an explanation to the low value of the goal satisfaction criterion. Corrective actions must thus first concern objects mapping.

An analysis of the alignment related to the product list shows that the business object 'Resort' does not exist in the system as a class. It is considered as a property of a hotel. This solution is sufficient to search hotels according to the client destination. However problems occurred in situations of partial closures of hotels.

There is an inconsistency between the concrete concept of a room and its representation in the information system. Rooms exist in the system only through their total number for a given hotel. In the business, rooms are considered as objects. This gap generates difficulties in the room creation or closure. Furthermore, the system includes the class 'hotel' but does not memorise temporal hotel closure that it considers as definitive closure followed by opening.

The goal corresponding to the products management can thus not be satisfied.

A similar analysis shows that booking modification initiated by an agent or a hotel keeper is not possible in the system. This has repercussion on the goals satisfaction.

In order to raise the value of the information completeness criterion, Room and Resort are added as class in the system. The states 'open', 'closed' and the properties 'number of beds', 'sanitary', 'comfort (smoking/no smoking)' are associated to the class 'Room'. The states 'open', 'temporarily closed', 'closed' and the properties 'sport activities', 'distance to the main city' characterise the 'Resort' class. Such corrective actions have an impact on other criteria as information accuracy, goal satisfaction, dynamic realism…

Calculations of the *functional factor* criteria have been performed for the booking creation activity. For this activity, each class (respectively state) *maps* a class (respectively state). This activity is thus properly supported by the system.

6. Discussion

The use of the BWW and the SW meta-models allows to *formally* define a fit measurement system. Indeed, the metrics are not only expressed in natural languages as for example in [25], but rely on formal concepts and links that are clearly and precisely defined.

The fit measurement system is defined at a *generic* level. Specific metrics are easily generated for two models representing the business and the system respectively. Such an approach allows to provide *model and project independent* metrics.

Measures can help relating problems which are felt in the daily business life to lack of system support and then, making decisions to overcome these problems.

The *measures rely on models* and thus depend on their pertinence and how we can trust them. They allow to identify misfit problems between models and not directly between the business and the system. It can be pertinent to, first of all, bring up to date the models in order they correspond as precisely as possible to the system or the business.

The metrics described in this article give an image of the alignment at a given point of time. They help in the choice of design option, or in determining when corrective actions are necessary. However, they should *not* be *the unique decision criterion* of the stakeholders. Indeed, fit has not only to be reached but also and above all to be maintained. Thus, a flexible but not very well aligned system could have the preference of the stakeholders on a system less flexible but better aligned with the business.

The metrics allow to put figures on an abstract concept, the alignment between business and system. They thus help to identify what is aligned and not and in which ratio. It is extremely rare that every measure equals one. This does not obviously imply that corrective actions are necessary. *Thresholds* introduction [25] allows a better appreciation of the fit relationship. Different values can be affected to the thresholds but correspond to the stakeholders' vision. If the metric determines a value lower than the associated threshold, the business process and supporting software system are misfiting, motivating corrective action. Thresholding allows to put alignment into perspective and not to restrict fitness to cases where each fitness measure equals 1.

This approach has not been evaluated on e-business or e-government project. We thus don't know if the measure of alignment can be performed in the same way when IT is used as business strategy and operational concerns. Future works on this point have to be undertaken. However, we believe that measuring alignment has to rely on formal and explicit links between system and business constructs and has to take into account the multi facets of alignment.

7. Conclusion

In this paper, we have presented business / system alignment metrics. They are organised into factors and criteria. Each criterion brings a different viewpoint on the nature of the business/system alignment. The metrics have been defined at a generic level bringing a theoretical basement, and serving as guide to define specific metrics. The generic metrics have been specialised for two specific models MAP model and UML class and transition diagrams that respectively represent the business and the system. We have illustrated the use of the metrics through a hotel room booking system and show how they can help to identify correctives actions. In the example, the corrective actions concerned the system since the business practice was considered as more adapted. Nevertheless, it would have been possible to modify the business and to conform it to the system. Furthermore, we have shown that (i) a high value of the support ratio does not imply a complete alignment between the business and the system, (ii) for each criterion, it is possible to define a threshold value helping to fix a limit value under which corrective actions must be performed.

Some extensions can be envisaged as the weighting of the criteria. Indeed, in order to better take into account the characteristics of each project in the calculation of the alignment measures, we intend to explore the weighting technique that we illustrated in the case study. This allows to attribute relative importance according to, the added value, the customers satisfaction, the frequency, etc… For example, the definition of the *Support ratio* considers the number of automated activities and ignores the relative value addition of activities in the business. Thus, it can happen that the support ratio is high but the most value adding activities are not automated. Appropriate weighting obviates this problem.

We also project to quantitatively and qualitatively evaluate the use of the metrics in order to determine their efficiency, their usability or their performance.

10. References

[1] BPMDS 04 special topic "Creating and maintaining the fit between business processes and support systems", Riga, Latvia, 2004.

[2] S. J. Bleinstein, K. Cox, J. Verner "Strategic Alignment in Requirements Analysis for Organizational IT: an Integrated Approach" *20th ACM Symposium on Applied Computing, track on Organisation Engineering*, Santa Fe, USA, 2005.

[3] R. A. Renner, D. Wong, D. Latimore. Business and IT operational models in financial services: Beyond strategic alignment. New York: IBM Institute for Business Value 2003.

[4] R. Sabherwal, and Y. E. Chan. "Alignment Between Business and IS Strategies: A Study of Prospectors, Analyzers, and Defenders". *Information Systems Research*, Vol. 12, N°1, March 2001, pp. 11-33.

[5] R. Hirschheim, R. Sabherwal. "Detours in the path toward strategic information systems alignment" *California Management Review*, Vol 44, N°1, 2001

[6] P. Soffer "Fit Measurement: How to Distinguish Between Fit and Misfit", note for BPMDS'04, Riga, Latvia, 2004.

[7] D. Nadler, ML. Tushman "A congruence Model for Diagnosing Organizational Behavior". *Resource Book in Macro Organizational Bahavior*, in Miles R (ed.), Goodyear, Santa Clara, CA, 1980, pp 30-49

[8] M.M Lehman, J.F Ramil and G Kahen, "Evolution as a Noun and Evolution as a Verb", *Proceedings of Workshop on Software and Organisation Co-evolution*, London, 2000

[9] C. Salinesi and C. Rolland, "Fitting Business Models to Systems Functionality Exploring the Fitness Relationship". *Proceedings of CAiSE'03*, Velden, Austria, 2003.

[10] E. Mitleton-Kelly and M-C Papaefthimiou "Co-evolution & an enabling infrastructure: a solution to legacy?", Systems engineering for business process change', Peter Henderson, Springer-Verlag, 2000, ch. 14

[12] A. van Lamsweerde. "Goal-Oriented Requirements Engineering: A Guided Tour". *IEEE International Symposium on Requirements Engineering*, Toronto, August, 2001, pp. 249-263

[11] A. Etien, C. Salinesi, "Managing Requirements in a Co-evolution Context". *To be published in the proceedings of the IEEE International Conference on Requirements Engineering*, Paris, France, 2005.

[13] E. Yu. "Agent Orientation as a Modelling Paradigm". *Wirtschaftsinformatik*. 43(2) April 2001. pp. 123-132.

[14] J. Han, "Supporting Impact Analysis and Change Propagation in Software Engineering Environments", *Proceedings of Workshop on Software Technology and Engineering Practice* (STEP'97 / CASE'97), IEEE Computer Society Press, London, UK, July 1997, pages 172-182.

[15] A. Arsanjani, J. Alpigini, "Using Grammar-oriented Object Design to Seamlessly Map Business Models to Component-based Software Architectures*", Proceedings of the International Symposium of Modelling and Simulation*, Pittsburgh, PA, USA, 2001, pp 186-191.

[16] C. Rolland, C. Salinesi, and A. Etien, "Eliciting Gaps in Requirements Change", *Requirement Engineering Journal*, Vol. 9, 2004, pp1-15

[17] Y. Wand, R. Weber, "An Ontological Model of an Information System", *IEEE Transactions on Software Engineering*, November, 1992, pp. 1282-92

[18] Y. Wand, R. Weber "An ontological evaluation of systems analysis and design methods", *Information System Concepts: An In-depth Analysis*, in E.D. Falkenberg, P. Lindgreen (Eds), North-Holland, Amsterdam, 1989, pp79-107

[19] P. Soffer, Y. Wand "Goal-Driven Analysis of Process Model Validity". *Proceedings of CAiSE'04*, Riga, Latvia, 2004.

[20] Bunge, M Treatise on Basic Philosophy: Ontology I. The Furniture of the World, Reidel, 1977.

[21] Bunge, M Treatise on Basic Philosophy: Ontology II. A World of Systems, Reidel, 1979.

[22] J.P. Cavano,. J.A McCall "A Framework for the Measurement of Software Quality". *Proceedings of the Software Quality and Assurance Workshop*, ACM SIGMETRICS and SIG-SOFT, San Diego, CA, 1978, pp. 133–139.

[23] Boehm B., J.R. Brown, H. Kaspar, M. Lipow, G. McLeod, M. Merritt Characteristics of Software Quality, North Holland, 1978.

[24] ISO/IEC 9126: Software Engineering - Product quality - Part 1: Quality model (2001)

[25] T. Bodhuin, R. Esposito, C. Pacelli, M. Tortorella "Impact Analysis for Supporting the Co-Evolution of Business Processes and Supporting Software Systems". *Proceedings of BPMDS'04*, Riga, Latvia, 2004.

[26] C. Rolland and N. Prakash. "Matching ERP System Functionality to Customer Requirements", *International Symposium on Requirements Engineering (RE)*, Toronto, Canada, August 2001

[27] A. Etien and C. Rolland "A Process for Generating Fitness Measures", *To be published in the Proceedings of CAISE'05*, Porto, Portugal, 2005

Exploring the Acquisition and Specification of Business Goals in Requirements Engineering

Sase Narine Singh *and* Carson Woo
Sauder School of Business
University of British Columbia
Vancouver, Canada, V6T 1Z2
sase.singh@sauder.ubc.ca, carson.woo@ubc.ca

Abstract

The paradigm of goals has recently emerged in the domain of requirements engineering (RE) in response to the appeal to realign system design with organizational context and rationale. While a number of frameworks have proposed in literature to comprehend and conceptualize goals within organizations, most of them are grounded on one of the four primary aspects, *elicitation, negotiation, specification,* and *validation*, as defined in RE, and use different representations. This paper suggests an alternate approach that incorporates the *elicitation* and *specification* aspects of goal based models, the *'what'* (i.e., business activities) and *'why'* (i.e., goals), and a unified goal schema as a small step to bring RE closer from inception to the final requirements product. The potential contributions of our proposed model are illustrated through a case study.

1. Introduction

By the early 1990's, research in the field of Requirements Engineering (RE) has been dominated by two popular schools of thought, namely the goal and process perspectives. In the former perspective, goals in RE are emphasized as logical mechanisms for identifying, organizing, and justifying software requirements [1]. Green [2] defined goals as requirements that describe states to be achieved, maintained, or avoided by a system. Prior to the introduction of goals, scholarship in RE, being entrenched in the process perspective has focused primarily on the '*what features*', i.e., what are the activities and events. With the introduction of goals, the purview of RE expands to include the '*why features*', i.e., why are systems constructed the way they are and what are the motivation and rationale behind the requirements used to construct these system [1].

The RE analytical process can be commonly classified into four phases [4], namely *elicitation*, *negotiation*, *specification,* and *validation*. Requirements *elicitation* focuses on understanding the current organizational situation and the need for change. Requirements *negotiation* establishes an agreement on the requirements of the system among various stakeholders involved in the process. Requirements *specification* maps real-world needs onto a requirements model. Requirements *validation* ensures the congruency of the system specification with the goals of the stakeholders [4].

There have been many frameworks proposed for the analysis of each of the aforementioned RE activities. Kavakali et al. [4], for one, have offered an exhaustive summary of the different frameworks used for eliciting, negotiating, specifying, and verifying requirements. Despite the multiple theoretical frameworks available for the representation of these requirements activities, none of them offers any comprehensive process capable of producing working models, which captures the detailed sequence and rationale behind the range of activities from the point of requirement elicitation to the subsequent steps of specification and validation.

Generally, each of these frameworks tends to lean heavily towards either the goal or the process perspective and in doing so, fails to realize the promise of an RE strategy founded on the fusion of the two dominant perspectives. Another potential deficiency detected in these analytical frameworks resides in their seeming disregard for the active involvement of stakeholders, a dimension that has been noted by scholars as an important aspect of the requirement modeling process [4].

In summary, Kavakali et al. [4] provided an analysis and critique on the current methods of RE analysis, and concluded that further research is necessary in order to arrive at a more holistic appreciation of the analytical techniques across the RE spectrum. This view was further echoed by other scholars who argued for the need to develop an overarching view of RE concepts and approaches [5, 6].

We thus proposed an alternate model that contributes to two of the four existing RE activities in this paper: requirements elicitation and specification. In the System Development Life Cycle (SDLC), requirements extraction and specification are the pillars to the development of any software application. While there will be significant benefits in incorporating the four components in the model, we have modeled only these two to allow for a more manageable research scope in this particular study. The elicitation element aids in acquiring requirements in the context of goals. With these given requirements, operational specifications are derived to lead into the design and implementation for the said system.

This paper is organized as follows: Section 2 discusses existing work of the popular approaches to requirements elicitation and specification. Section 3 provides the proposal of a RE model that leads to the development of *goal schemas* (A goal schema is a template that is used by many scholars [1, 17, 27] to consolidate goal information). Section 4 outlines a methodology in developing the goal schemas. Section 5 examines a case study of using the methodology. Section 6 analyzes the proposed model. Finally, Section 7 concludes with some directions for future research.

2. Related Work

2.1. Goal Elicitation Techniques

The identification of a suitable process, the selection of methods and techniques are considered as *elicitation* (or ground work) in RE—what Kavakali et al. [4] describe as an understanding of the current organizational situation and its need for change. Nuseibeh et al. [7] further operationalized this elicitation procedure as an instance of the process model which offers an abstraction of how to conduct a collection of activities, describing the behavior of one or more agents and their management resources. In this section, we will be examining how the current frameworks of goal elicitation as categorized by Kavakali et al. [4] are consistent or inconsistent with the above definition.

According to Kavakali et al. [4], Goal-based Workflow [8], GOMS [9], F^3 [10], and i* [11] are frameworks considered as elicitation approaches in RE.

The Goal-based Workflow views the organization as a tuple [*G, A, R*] where *G* represents a set of goals, *A* represents a set of actors, and *R* represents a set of resources. Actors collaborate using Resources in order to attain Goals. The primary interest of this framework is fixated on goals and the allocation of resources rather than on activities and procedures [4]. This framework does not describe how goals relate to organizational activities or how actors' goals impact actor collaboration.

The GOMS framework is considered as a technique in cognitive analysis which focuses on human tasks [4]. The framework consists of the following elements: goals (external tasks), activities (internal tasks), device, and actions. To attain a goal or a desired state, a set of activities are required. For each activity, actions are taken. These actions are modeled through a device such as a method, an agent, a tool, or a technique that induces the evolution of the system. The disadvantage of the majority of these cognitive task analytical techniques currently available is that they focus on routine human computer interactions. Thus, they capture very low level operational goals and do not scale up well for strategic enterprise-wide applications [Preece 1994].

The Objectives Model (OM) of the F^3 framework provides rich formalisms for expressing goals and goal relationships. The OM is appropriate for describing the intentional and motivational perspective of the enterprise, i.e., the enterprise goals along with the hurdles obstructing goals achievement [4]. It is often termed as a good 'conversational' tool among stakeholders for understanding current problems and explicitly identifying future goals and opportunities. However, ambiguities in goal interpretation can potentially occur as the relationships among the goals are depicted with alias names for flexibility. This can culminate into undesirable outcomes in implementation.

The i* approach provides an intuitive depiction of organizational work in terms of dependency relationships among actors. In this analytical technique, the organization is construed as a network of interdependencies among actors whereby each of the actors depends on one another for goals to be accomplished, tasks to be completed, resources to be supplied and soft goals to be satisfied. The assumption of this model is that actors within the organization have freedom of actions bounded only by the social constraints. While the i* focuses on relationships between actors, goals, and tasks, little or no emphasis is placed on activities and their relationships.

From the above analysis, it is perceivable that the current frameworks are mainly focusing on eliciting goals (the why feature), the 'what feature', which lays the map of suitable business processes, is either still missing or has only been partially addressed in comparison to the elicitation process prescribed by Nuseibeh et al. [7]. To compensate for this inadequacy, we put forward a proposed framework that addresses and encompasses both the 'what' and 'why' features in RE. These features are captured by constructing a framework that formalizes the operationalization of the business processes (which thus illustrates the 'what' feature) through the use of a rigorous methodology to elicit goals (which simultaneously illustrates the 'why' feature).

2.2. Goal Specification Techniques

The requirements specification focuses on operationalizing goals into functional and non-functional system components. Kavakali et al. [4] purports that requirement specification should go beyond traditional functional modeling approaches to encompass modeling procedures, which are sensitive to the enterprise context to accentuate the purpose of the intended system.

Researchers have advocated "increased user involvement in the systems development process, stating that the heightened level of participation contributes to the development of better systems" [15], thereby shaping a more concise purpose for the intended system. According to Hayes, user participation enables the creation of more relevant systemic models of business processes than those created solely from the perspective of the analyst, thereby enhancing the fit between the implemented system and the corporate objectives [39]. It is, therefore, imperative to initially validate the goals with the stakeholders (to ensure congruency) before defining the system components.

According to Kavakali et al. [4], there are four prevailing frameworks utilized for requirements specification: the KAOS framework [12], the NFR Framework [13], the Goal scenario coupling framework [14], and the GBRAM framework [1]. This section investigates whether the frameworks have adequately made provisions to verify the goals with the stakeholders.

The KAOS was derived from machine learning and adopts a formal methodology in representing goals. The framework describes a sequence of steps and associated techniques that can be applied when performing goal modeling. It takes three inputs (clients' requirements, a KAOS meta-model, and a meta-model traversal strategy) to output *Z* data and operation schemas. The KAOS methodology stresses the need to explicitly specify and structure goals, whilst devoting considerably less attention to the issue of initial goal identification and formulation [16]. Even though KAOS is rigorous in supporting the process of requirements elaboration (from high level goals that should be achieved by the composite system to the operations, objects and constraints to be implemented by the software), there is little evidence to suggest support or interaction from the stakeholders. By specifying goals in terms of *Z* notations, the task of validating the goals becomes increasingly difficult as stakeholders may not be familiar with the schematic representations.

The NFR framework is a comprehensive framework that provides for the representation of non-functional requirements in terms of interrelated goals. The model consists of mainly: goals that represent non-functional requirements (NFR goals); design decisions (*satisficing* goals); arguments in support or against other goals (argumentation goals); and relationships for relating goals to other interdependent goals. The NFR framework is subjective and relative because the non-functional requirements can be viewed, interpreted, and evaluated differently by different people [17]. Stakeholders usually have a better understanding of the general goals they want to achieve than they do the functionality that should be exhibited by the desired system [15]. The NFR framework concentrates on Quality Attribute/Non-functional attributes (e.g., *accuracy and security*). These types of goals will likely pose as sources of communication and comprehension problems with the stakeholders. As the name implies, functional requirements is not part of the framework.

The Goal-scenario coupling approach uses scenarios to elicit future organizational goals and to operationalize them in terms of system components. In accordance with the goal-scenario coupling strategy, the identification of alternate solutions is addressed through analysis of possible future scenarios by business experts. This approach is pivoted on the prerequisite that the goal and scenario must have already been explicated, which in itself is a presumptuous argument [26].

The GBRAM framework is a representation for specifying goals [1]. This model is well suited for identifying functional requirements which represent specific behaviors the proposed system should exhibit. The framework offers prescriptive guidelines on how to extract goals from different sources into one ordered goal set termed the goal schema. Goal information is captured initially and consolidated into a set of goal schemas which are ultimately translated into a set of requirement specifications. The schema consists of operationalized goals, responsible agents, stakeholders, scenarios, obstacles, and subgoals etc. [15]. The limitations of the GBRAM includes: an informal method, as opposed to formal semantics, for goals and thus, it does not support formal reasoning. While the GBRAM supports a high degree of stakeholders' involvement in its framework, we maintained that the process is resource intensive (since goals are identified in a semi-formal way using a number of different sources – an initial study [1] showed that 36 lines of text results in about 19 goals) and may result in complicated cognitive processing required of the stakeholders.

It should be clear from the above analysis that there are rooms for improvements. In particular, there is a need for a more effective and efficient way of verifying the goals with the stakeholders. To address this issue, provisions are made within the proposed framework to traverse on both ends of the stakeholders and analysts spectrum. The mid point of these two ends is represented by schemas. These schemas encapsulate knowledge that facilitates the high involvement of stakeholders and the outputs of specifications to be used in system designs.

3. Proposed Model

According to Hoffer et al. [36], systems analysis focuses on understanding the organization's strategies, objectives, structure, and processes. This understanding in turn is an important aspect of strategic alignment within an organization by defining a coherent architecture of Business-IS strategies and Infrastructure [39, 40]. To this end, our study puts forward a model whose constructs are consistent with those as proposed by Henderson & Venkatraman, Pigneur et al. [39, 40].

This study is formulated upon two premises. Firstly, the strategy of an organization constitutes the single most important input (Hackman & Lawler, 1979) by encompassing the core mission (mission statements, specific tactics, and output objectives (goals)) that the organization needs to accomplish. In realizing the core mission, strategies are derived from the environment, resources, and history. The strategies determine the work the organization should be performing and the nature of the desirable organizational output [20]. Secondly, to be able to achieve the desired output; the firm operationalizes these strategies through some transformational processes [20]. Today, the survival of many of these organizations is based on transforming these complex processes to computer based information systems. The first step in this transformational process is to create a representation of the organizational domain for which the information system is being developed. This representation is often structurally developed to exemplify, classify, and describe the operational data and its relationships with the existing business activities.

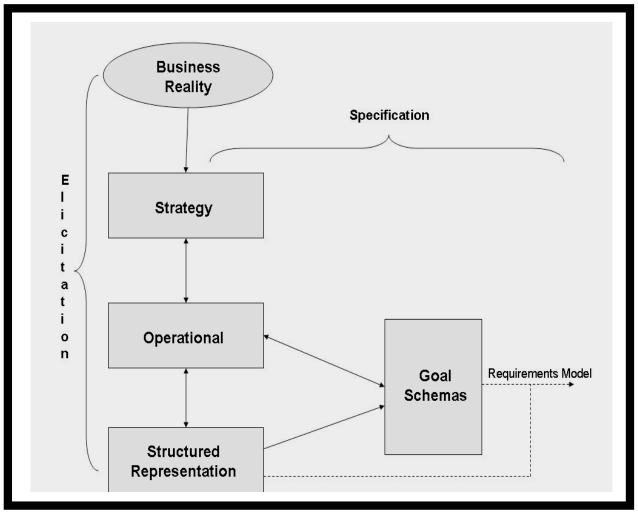


Figure 1 Model that combines goal elicitation and specification

3.1. Components Representation

The aforementioned two premises lead to the model in Figure 1, which adopts a top down approach. In the model, the '*business reality'* reflects the environment, resources and history; the *'Strategy'* reflects the organizational strategies, the *'Operational'* reflects the operationalization of the strategies, which includes organizational structure, job descriptions, processes etc.; and the *'Structured Representation'* reflects a specific representation of the organizational domain for which the information system is being developed. The arrow between the boxes reflects the transformational process.

After systematically and structurally representing the organizational domain, the subsequent steps in developing a system lies in the specification of these domains as formal representations (the why and the what) that can be interpreted in the design phase. In accomplishing this, '*Goal Schemas*' are used to represent the consolidation of the 'why' features. The combined output of the *'Structured Representation'* and *'Goal Schemas'* (dotted lines in Figure 1) form the amalgamation of the 'what' and 'why' for formal design purposes.

Goal elicitation and partial specification of the attributes in the goal schema are achieved through the goal elicitation process from *'Structured Representation'* to *'Goal Schema'* in Figure 1. Attributes not specified at this stage will be identified by revisiting the existing operations and stakeholders, since those are considered as important sources of information. This informal process is shown in Figure 1 as an arrow connecting the '*Goal Schema*' and the *'Operational'*.

3.1.1. Goal Schema

The goal schema (Table 1) in this model is derived such that it can drive architectural decisions while simultaneously, be used to verify stakeholder needs. Each schema consists of a set of attributes derived from the following criteria: (i) most commonly used attributes in existing schemas [1, 17, 27-29]; (ii) the attributes should be able to relate back to the stakeholders (since the system will be representing their needs and they will be the ones who will be signing the final contract); (iii) the attributes should be adequate for use in later design stage such as developing an Entity-Relationship Diagram (less informative, ambiguous, and irrelevant attributes will limit the ability to design well); and (iv) attributes should be adequate and relevant to elicit *early aspects* for design and implementation. Early aspects are crosscutting concerns that are identified in the early phases of the software development life cycle, including requirements analysis, domain analysis and architectural design [37].

Primary Goal	Name of Goal that was identified in the OOEM
Description	Description of the goal
Action	Action to achieve Goal
Agent	Object that is responsible to achieve goal

Stakeholders	Objects that claim direct stakes in the goal
Constraints[1]	Ways in which the goal can be blocked
Sub Goals	Other goals that leads to the achievement of the goal
Priority	Express the importance of the goal to the stakeholders

Table 1 A template of the attributes in a schema

3.1.2 A Specific Structured Representation: OOEM

Ideally, the *'Structured Representation'* in Figure 1 should capture the 'what' aspect of business operations and provide the necessary information to easily fill out or derive the attributes in the goal schema in Table 1. Since we are unaware of the existence of a methodology that can be used for systematically creating the structured representation, we may have to look beyond the scope of RE elicitation literature for an alternative to fill out as many of the goal schema attributes as possible and leave the remaining ones for the revisit step with stakeholders mentioned above.

The *Object Oriented Enterprise Model* (OOEM) is one such alternative. The end result of such modeling technique matches closely [21,23] the manifestation of our idea of the "structured representation" because of the methodology's ability to capture roles and responsibilities of organizational actors and their interactions, where the sequence of interactions can be used to derive business process and activities. This satisfies the 'what' aspect of RE and provides values for five of the attributes in the goal schema (see the first five attributes in Table 2). Furthermore, OOEM was derived from a theory of reality, Bunge's ontological principle [25], it should better reflect the business reality.

3.1.3. Goals from OOEM

Another advantage of using OOEM is that a method has been developed to deduce goals from it and the method was shown to possess the capability to capture the majority of the goals [26]. The missing goals are those that require additional information and knowledge beyond what is apparent in the business operations (e.g., common sense objectives in a particular business context, policy and regulation in a specific industry, and etc.). This is another reason why verifying goals with stakeholders is important (the arrow between *'Operational'* and *'Goal Schemas'* in Figure 1).

4. Method

4.1. Developing the OOEM – representing the 'what'

The OOEM diagram is developed based on seven rules proposed by Wand and Woo [21-24]. The fundamental constructs of OOEM are objects, services, attributes, and requests. The algorithm for developing the OOEM is, briefly, as follows. First, identify all the external objects relevant to the system (i.e., the enterprise) to be modeled. For each external object, identify all the requests it sends to the system. For each request, identify the receiving object, its associated interface attributes and services, and request it will send out to other objects. The above steps are repeated until all requests are modeled.

The result of this model is a set of services that needs to be performed in order to satisfy the external requests, who are the roles to respond to the requests, and how are these roles responding (through services) them.

4.2. Eliciting goals from OOEM – representing the 'why'

The methodology proposed by Wang [26] to deduce goals from OOEM is based on the semantics of OOEM constructs, concepts such as stability and emergent property in Bunge's Ontology [25], and Linguistic Negation Interpretation theory [26]. Briefly, the methodology starts by analyzing the activities inside a service and use them to form the goal of the service. The purpose of the request that triggers the service also plays a major role in determining the goal of the service. Since an object is composed of services, the goal of the object is the emergent result of all its services (i.e., a new goal attended that was not possible by any of the individual services). The Linguistic Negation Interpretation theory plays a major role here in forming the emergent goal. The goals of the subsystem and system are similarly deduced as the emergent goal of the objects and subsystems, respectively. Ideally, the goal of the system should be the same as the goal of the organization.

4.3. Mapping of OOEM goals to schema attributes

Following the design of the OOEM and the elicitation of the goals, the subsequent step is to map them to the goal schema. The partial mapping of a service goal is shown in Table 2, where the service goal is construed as a primary goal in the schema. While the combination of the service goals for each of the object forms an emerging goal, consideration for that goal will not be accounted for at this point. For future study, we will be aligning the emerging goals with the goals of the organization.

[1] Constraints, as defined in Table 1, are synonymous with the concept of obstacles posited by Anton and Potts [41].

OOEM and it Goals Elicited	Goal Schema Attributes
Object requesting service	Stakeholders
Object providing service	Agent
Service	Action
Service goal encompassed in object	Primary Goal
Description of Service Goal	Description
	Sub goals
	Constraints
	Priority

Table 2 Mapping attributes from OOEM to Schema

From the above table, it is conceivable that the OOEM and its elicited goals only account for five of the eight attributes in the goal schema - i.e., neither the OOEM nor the goal elicitation technique provides any elucidation for the sub goals, the constraints, and the priority of the goals. These auxiliary activities are identified after a verification process with the stakeholders, which will be discussed below.

4.4. Filling the missing attributes by mapping schema onto operational information

To determine the remaining attributes and to ensure high degree of accuracy, it is imperative to revisit the operationalization component and re-consult the stakeholders. Through this clarification process, conflicts can be resolved and *hidden goals* (goals that were not exemplified in the processes) are allowed to evolve. Figure 2 shows an approach to facilitate the identification of the remaining attributes and the verification of the existing attributes in the goal schema: 1) for each schema, identify agent; 2) Resolve any conflicts relating to any identified attributes with agent – so as to resolve disagreements and ensuring a higher level of consistency; 3) Identify sub goals by: searching for key words or phrases that suggest a continual state within the system (Anton, Brito, Elisa [1, 31, 32] have all adopted this searching process in their templates); and direct observation (for it is especially important, for often actors do not recognize their own subtle dependencies on other teams nor is it explicit in documents [33]); 4) Elaborate goals, uncover hidden goals and requirements and refine them; 5) Identify constraints; and 6) Identify priorities.

In the following section we will demonstrate the operation of our proposed methodology using a case study.

5. Applying the approach to a case study

The case study was adopted from a pilot study conducted by Ali and Zhu [34], in modeling the application process of a potential candidate of the Sauder School of Business at University of British Columbia. The sequences of events are as follows:

- The PhD application process starts with applicant submitting an application form with payment to the Faculty of Graduate Studies (FOGS).
- On the successful completion of the above process the application is sent to the PhD administrator of the Sauder School of Business along with any supplemental material from the applicant such as: undergraduate and graduate universities, Educational Testing Service (ETS), and referee letters.

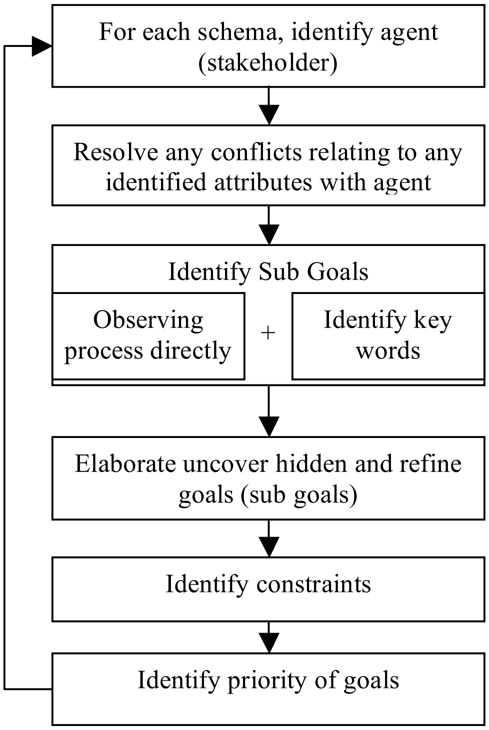

For each schema, identify agent (stakeholder)
Resolve any conflicts relating to any identified attributes with agent
Identify Sub Goals
Observing process directly
+
Identify key words
Elaborate uncover hidden and refine goals (sub goals)
Identify constraints
Identify priority of goals

Figure 2 Rules to populate remaining attributes in schema

- The PhD Administrator summarizes data collected, and forwards it to the School's PhD Director who in turn filters out/in the applications.
- The application material is then examined in detail by the Division PhD advisor who replies to the PhD director with the final recommendation.
- Letters from the PhD director are sent to the applicant through PhD administrator after all materials are authenticated.

The process is finalized when the applicant receives an offer from FOGS.

5.1. Developing the OOEM

In the given case study, the external request was the application for admission to the PhD program. The FOGS, PhD Administrator, PhD Director, Division Advisor, and Enrollment Office were the internal objects while the referee, previous university, ETS etc. were the external objects.

5.2. Eliciting goals from OOEM

After developing the OOEM, goals were elicited. Table 3 provides a summary of the goals (service goals) for each of the agents.

Goals that were extracted from the OOEM	
Agent	Services Goals
Faculty of Graduate Studies (FOGS)	• Captures all the necessary information to make a decision • Make sure transcripts are not fake • Make sure every body pays • Make sure students meet minimum requirement
Enrolment Office	• Make sure funds are collected
PhD Administr ator	• Summarize data and forward information to PhD director • Provides feedback and admission decision to corresponding parties. • Seeks final approval from FOGS.
PhD Director	• Filter out/in applications • Communicates decision to FOGS
Division PhD Advisor	• Admits best students.

Table 3 Goals elicited from OOEM

5.3. Mapping of OOEM and its goals to schema attributes

The roles, services, and attributes of the OOEM with its elicited goals were mapped to the partial attributes of schema. Each of the service goals was considered as one of the Primary Goals. The attribute *'Action(s)'* was obtained by examining the *'Service(s)'* in the OOEM that was responsible for the fulfillment of the goal. The attribute *'Stakeholder(s)'* was determined by identifying all of the other agents that were indirectly related to the achievement of the goal. Table 4 provides a summary of the identified attributes for one of the service goals.

AGENT: PhD Administrator	
PRIMARY GOAL:	Summarize data and forward information to PhD Director
DESCRIPTION:	with a given criteria, applicant information is summarized so as to be processed by the PhD Director
ACTION:	process supplementary application information, process official transcript, process reference letter, and process TOFFEL/GRE/GMAT scores
STAKEHOLDERS:	applicant, referee, ETS, previous university, FOGS, PhD Director, division
SUB GOALS:	
CONSTRAINTS:	
PRIORITY:	

Table 4 Attributes identified from OOEM and Goals

5.4. Filling the missing attributes

The remaining attributes of the goal schema were determined by revisiting the agent that was directly responsible for the achievement of the goal. We also attempted to ascertain consistency with the description, actions, and stakeholders that were indirectly involved. Conflicts were resolved by one or more of the following methods: having a discussion with the agent, direct observations, and examining existing documents that were directly related to the goal. We then established through the use of natural language and intuition a set of sub goals that were considered as *'personal goals'* by the agent. These sub goals were then listed in order of priority relative to the primary goal.

With the agent's experience and scenarios, we were able to identify current and potential new constraints of the primary goal. The remaining attributes for the schema are presented in Table 5.

AGENT: PhD Administrator	
PRIMARY GOAL:	Summarize data and forward information to PhD Director
DESCRIPTI ON:	with a given criteria, applicant information is summarized so as to be processed by the PhD Director
ACTION:	process supplementary application information, process official transcript, process reference letter, and process TOFFEL/GRE/GMAT scores
STAKEHOL DERS:	Applicant, Referee, ETS, Previous University, FOGS, PhD Director, Division

SUB GOALS:	Having the summarized information prepared in a defined format Having at least an unofficial summary submitted **before** deadline Having an official summary submitted **on/ before** a given deadline
CONSTRAINTS:	-collecting scores from different sources ETS (MBA's Office). -data from the different sources that were used to prepare the defined format. -summary are in different format (hard copy, email, attachments).
PRIORITY:	- having at least an unofficial summary submitted **before** deadline. - having the summarized information prepared in a defined format. - having an official summary submitted **on/ before** a given deadline.

Table 5 Attributes of Schema

Schematic representations were derived for every primary goal of every agent.

6. Analysis

While the model and the case study do not provide the level of rigor typified for model validation, it does offer some insights. In this section we discuss the potential value of our framework.

6.1. The use of 'what' and 'why'

One common problem in RE is the stakeholders' lack of understanding of their own requirements. One possible solution in dealing with this complexity is to provide technical requirements that are easily comprehensible to the intended stakeholders [15]. In the framework (Figure 1), the challenge of this complexity was dealt with by the OOEM. The OOEM provides an extensive understanding through a formal methodology of how human beings perceive an organizational process. It has attained a level of abstraction (requests, services and roles) that could be readily interpreted by stakeholders.

Following the formal illustration of the key roles within the system that address the external request, the subsequent stage in the framework (Figure 1) is to identify the goals (the 'why') for each role. This provides the motivation and rationale to justify software requirements.

The advantage of having both the 'why' and 'what' combined systematically is to fill the gaps and resolve any informational ambiguities that stakeholders may have with regard to the system. Also, the combination reduces any sort of ambiguities that the analyst may have in interpreting the needs of organization. From a practitioner's point of view, this contribution is cost effective in many dimensions (having a shorter time frame in capturing requirements since both parties are clear on the needs, and developing a system that accurately reflects the functional needs of the organization).

6.2. Ease of relating back to the stakeholders

The requirement specification based on formal terminology often serves as a contract with the stakeholders. These specifications are sometimes in the form of mathematical notations such as *z*. If stakeholders are unfamiliar with these notations, or have not been trained in formal specification procedures then the requirements documents may be deemed as cumbersome and intimidating. Since requirement specification documents often serve as a form of contractual obligation, it is essential to supply stakeholders with information in a comprehensible vocabulary in which they may actively participate [15].

We have confronted the challenge of providing a language that stakeholders can understand through the design of schematic templates. The attributes of the templates are expressed in the form of language that can be easily understood and communicated to the stakeholders. The initial attributes of templates were realized through a heuristic associated with the methodology of the OOEM. But not all of the attributes were captured in this process. The remainder of the attributes has to be determined through a semi-formal process i.e., revisiting the agent and stakeholders and trying to resolve conflicts, constraints and priorities.

In the case study, when relating back to the stakeholders, we found that they all understood the requirements as specified in the templates and demonstrated very little difficulty in the cognitive process of negotiation. The exercise was thus less resource intensive and conflicts were easily resolved.

We are confident that the ease of relating back to the stakeholders was attributed to the development of the OOEM as it facilitated a coherent understanding with the stakeholders, thus increasing the comprehensibility. As identified by Anton [1], stakeholders tend to express their requirements more in terms of operations and processes rather than goals and objectives [15, 35].

6.3. Targeting sources for missing information

When re-consulting with the stakeholders (represented by the line between the *Operational* and *Goal Schema* in Figure 1), we found that not all of the attributes were identified. As such we decided to reference the existing documentation in the organization. Since we were able to identify over 80% of the attributes in the template, finding sources for the remainder were relatively straightforward. We narrowed the information sources to the documentation relating to the agent directly and indirectly. In this process we gained the advantages in the form of a reduced search cost.

This process resulted in additional contribution to the analysts for they were able to uncover hidden sub goals and supporting information to some concepts which initially were not intuitively sound.

7. Conclusion and Future Research

In the Systems Development Life Cycle (SDLC), the requirements analysis stage is considered as the main feature that predominantly defines the relative success of software application projects [7]. The engineering discipline within this stage of RE comprises primarily of identifying the 'what' and 'why' aspects of the system [5]. In this paper we provided a framework and a methodology of combining the 'what' and 'why' aspects in RE. In combining the two, we first identified the 'what' by deriving an OOEM framework that represents a high level abstraction of the business processes and activities. From this framework, we then identified the 'why' through the elicitation of goals from the OOEM. These goals were then consolidated into a schematic representation. Through these steps, we found that the iterative process of resolving conflicts and converging to a common understanding with the stakeholders were neither time consuming nor resource intensive. We believed that by assimilating the 'what' with the 'why', stakeholders are in a better position of understanding the 'why' potentially leading to subsequent higher level of system acceptance.

In addition, we attempted to combine two aspects of goal analysis: elicitation and specification. Requirements elicitation, as defined by [4] is the process of understanding the organizational situation that the system under consideration aims to improve and describing its needs and constraints. Requirements specification involves mapping real world needs onto a requirements model. We achieved this by using the OOEM framework, its goal deduction method, and by developing a goal schema consisting of several attributes.

While this paper does not claim the proposed framework to be a substitute for other modeling approaches in goal analysis, its assimilation of requirements elicitation and specification offers an alternative and novel technique of aligning stakeholders' expectations with system design interpretations as conceived by analysts and developers.

In developing the goal schema, one important consideration to be noted when identifying the attributes is that the schema should possess the capability to be easily utilized by system designers. In a preliminary study that is not within the scope of this paper, we were able to derive use case diagrams and found that with the combination of these use cases and the OOEM, an ER model can emerge.

From this proposed framework some of the potential directions for future research lie in the following areas:

(i) The strategy and policy of a company are also very important to the goal of a department, but OOEM-base goal modeling is unable to identify any of these [26]. To combat this challenge, we propose a link between *'Strategy'* and *'Goal Schema'* (Figure 1). An alignment can then be determined between the strategic organizational goals and the operational goals (goals identified through the OOEM). One possible approach of creating the link between the *'Strategy'* and the *'Goal Schema'* (Figure 1) is through the i* framework [11].

(ii) We intend to match and align the personal goals of the stakeholders with the sub goals of the organization (i.e. the goals derived from the strategy) and propose a formal methodology for prioritizing these sub goals.

(iii) We intend to develop more formal representations apart from the goal schemas for the requirements specification. These representations will focus towards the design of system applications.

(iv) Finally, we intend to conduct more case studies. In this preliminary study, we have developed a methodology to elicit early aspects of the RE process in terms of goals. This work, however, needs to be further validated.

References

1. Anton, A., "Goal-Based Requirements Analysis," *Second International Conference on Requirements Engineering,* Los Alamitos, California: IEEE Computer Society Press, 1996, pp. 136-144
2. Green S., *Goal-Driven "Approaches to Requirements Engineering"*, Technical Report 1993/43, Department of Computing, Imperial College, London, 1993
3. Dardenne A., Lamsweerde A. van and Fickas S., "*Goal-Directed Requirements Acquisition"*, Science of Computer Programming', 20, pp. 3-50, 1993.
4. Kavakli E., and Loucopoulos P. "*Goal Driven Requirements Engineering: Evaluation of current methods"*. In EMMSAD 2003, 2003 Velden Austria.
5. Yu E. and Mylopoulos J.. "*Why Goal-Oriented Requirements Engineering"* Proceedings of the 4th International Workshop on Requirements Engineering: Foundations of Software Quality (8-9 June 1998, Pisa, Italy).

6. van Lamsweerde, A., "*Goal-Oriented Requirements Engineering: A Guided Tour,*", *Fifth International Symposium on Requirements Engineering*, Los Alamitos, California: IEEE Computer Society Press, 2001, pp. 249-262.
7. Nuseibeh, B., and S. Easterbrook, *"Requirements Engineering: A Roadmap,"* IEEE International Conference on Software Engineering, IEEE Computer Society Press, June 2000C.
8. Ellis, A., Wainer, J. *"Goal-Based Models of Collaboration"*. Collaborative Computing, 1, 1994.
9. Card S., Moran T., "*The Psychology of Human-Computer Interaction"* Hillsdale, NJ, Erlbaum 1983
10. Bubenko J., "*Enterprise Modelling"* Ingenierie des Systems d' Information 2(6), 657-678 1994
11. Yu E., Liu L., "*Modeling Strategic Actor Relationships to Support Intellectual Property Management"* 20th International Conference on Conceptual Modeling, ER-2001, Japan 2001.
12. Dardenne A., van Lamsweerde, A *"Goal-directed Requirements Acquisition"* Science of Computer Programming 20: 3-50. 1993
13. Mylopoulos J. L. Chung B. Nixon "*Representing and Using Nonfunctional Requirements: A Process-Oriented Approach"* IEEE Transactions on Software Engineering SE-18(6): 483-497 1992
14. Rolland C. Souveyet C. "*Guiding Goal Modeling Using Scenarios"* IEEE Transactions on Software Engineering 24(12): 1055-1071 1998
15. Anton A., "*Goal Identification and Refinement in the Specification of Software-Based Information Systems"*, Ph.D. thesis, Department of Computer Science, Georgia Institute of Technology, June 1997.
16. Kavakali E. "*Goal-Driven Requirements Engineering: Modeling and Guidance"* Chapter 2 PhD thesis
17. Chung L., Nixon L.B., Yu E. *Non Functional Requirements in Software Engineering* Kluwer Academic Publishers, 2000.
18. L.A. Macaulay. "*Requirements Engineering"* London, UK: Springer, 1996
19. Gleick J., "*Chaos: Making a New Science"* Harmondsworth, 1998.
20. Daft R., Sharfman, M. "*Organizational Theory Cases and Application's* 3rd Ed. West Publishing Company 95.
21. Wand. Y. and Woo C. "*Ontology-Based Rules for Object-Oriented Enterprise Modeling"*, Working paper 99-MIS-001, University of British Columbia, 1999
22. Wand Y., Woo C., Jung D. "*Object-Oriented Modeling: From Enterprise Model to Logical Design"*. Proceedings to the 10th Workshop in Information Technologies and Systems (WITS'00, Brisbane, Australia, 2000).
23. Wand Y., Woo C., Hui S. "*Developing Business Model to support Information System Evolution"* Proceedings to the 9th Workshop on Information Technologies and Systems WITS 99137-142.
24. Zhao H., "*Object Oriented Enterprise Modeling"* MSc. thesis University of British Columbia, July 1995
25. Bunge M., "*Treatise on Basic Philosophy Vol. 3, Ontology I: The Furniture of the World"*. Boston: Reidel, Dordrecht, 1977.
26. Wang W., "*An OOEM-Based Goal Modeling*," in Management Information Systems. Vancouver, B.C.: The University of British Columbia, 2004.
27. Moreira A. "*Crosscutting Quality Attributes for Requirements Engineering"* International conference on Software engineering and knowledge engineering 2002.
28. Malan R., Bredemeyer D. "*Functional Requirements and Use Cases"* Bredemeyer Consulting White Paper 1999
29. Malan R., Bredemeyer D. "*Defining Non-Functional Requirements and Use Cases* 2" Bredemeyer Consulting White Paper 2001
30. Connolly T., Begg C., "*Database System, A practical Approach to Design and Implementation", Management*. Addison-Wesley 3rd Edition 2001, USA.
31. Brito I., and Moreira A. "*Towards a Composition Process for Aspect-Oriented Requirements"*. Early Aspects Workshop at AOSD Conference. Boston, USA. 2003.
32. Elisa L. A "*Theme: An Approach for Aspect-Oriented Analysis and Design"* ICSE 2004
33. Briand L *"Characterizing and Assessing a Large-Scale Software Maintenance Organization*" Proc. of the 17th International Conference on Software Engineering, Seattle, WA, 1995.
34. Ali A., Zhu L. "*An Investigation of Combined Top Down and Bottom Up Approach in Goal Modeling* – A Research Proposal", in Management Information Systems. University of British Columbia, 2004
35. Anton A. I. "*Goal Decomposition and Scenario Analysis in Business Process Reengineering"* Adv Info. System End. 6th Int. Con. Utrecht. The Netherlands 1994.
36. Hoffer J., George J. "*Modern System Analysis and Design"* 3rd Edition Prentice Hall
37. Tekinerdogan B. "*Early Aspects: Aspects Oriented requirements Engineering and Architectural Design"* Workshop Report, Lancaster, UK, March 2004
38. Hayes G. Group Matrix tool "*A Collaborative Modeling Tool"* PhD thesis. University of Arizona 1991
39. Henderson J. C., Venkatraman N. "*Strategic alignment: Leveraging information technology for transforming organizations"* IBM Systems Journal, Vol. 38 number 2&3 1999.
40. Pigneur Y., Osterwalder A. "*An ontology for e-business models"* chapter in Wendy Currie (ed) Value Creation from E-Business Models, Butterworth-Heinemann, 2003
41. Anton, A., and C. Potts, "*The Use of Goals to Surface Requirements for Evolving Systems*," *Twentieth International Conference on Software Engineering*, Los Alamitos, California: IEEE Computer Society, 1998, pp. 157-166.

Integrated Methodology for linking business and process models with risk mitigation

Maria Bergholtz, Bertrand Grégoire, Paul Johannesson,
Michael Schmitt, Petia Wohed and Jelena Zdravkovic

Abstract—**This position paper introduces means of designing business processes that fit a more strategic business model. Our proposal focuses on value webs and risk mitigation as important forces for process design and management. Value webs are defined as a constellation of value exchanges, and risks are defined on flows of different kinds, such as financial flow and goods flow. In this article we try to move from business models to process models in a systematic way based on dependencies between activities and risk mitigation choices.**

Index Terms—**business model, activity dependency, risk mitigation, value web, process model**

I. Introduction

Business process models explain how agents co-ordinate their respective activities to achieve a common goal. But how can we be sure that an operational business process does support these common goals? How can we be sure that the goals are well understood by the agents and that all agents have converging goals? If we focus on the process activities, we can note that some of them may be considered as truly value-creating activities in that they contribute to the achievement of the formulated goals; others may be seen as being more of a coordinative or supporting nature, in that they facilitate the communication among the agents.

In order to draw the border line between these two groups of activities, and in order to align each process activity with one or more common goals, we need to introduce another level of description of the business collaboration that explains that "what" and "why" of the collaboration of the agents while the process model details the "how". This means that the business model describes entities like values, value propositions, agents and resource exchanges, while the process model focuses on procedural aspects like control flow. A link between the operational process model and the upper-level business model seems important with regards to the fact that the business goals may evolve over time, and hence should be reflected in the business process model.

Aligning process models with business models raises two major questions. First, what are the basic concepts in a business model? Secondly, how should the gap between business models and process models be filled? In this paper, we outline answers to these questions. In particular, we argue that risks and risk management are important forces in process design, and we discuss how risk mitigation instruments influence business and process models.

In section II, we introduce a number of basic concepts in business models and contrast two different views of business models. We also summarise previous work on risk management and methods for moving from business models to process models. In section III, we outline an integrated ontology of the two views on business models and present an integrated methodology for including risk mitigation instruments in business and process models. In the final section, we summarise the work and suggest directions for further research.

II. Ground work – state of the art.

A. Business model definition

A complete business model, as described by Osterwalder [1], details a general view of the context of a business. This global view includes, as main concepts, the *customer* that expresses a *demand* the business tries to meet with a *value proposition*, composed of *offerings*. The value proposition of a bundled product, for instance, would include a personal computer, the pre-installed and configured Operating System, and additional services, as home delivery. In order to be able to provide this proposition, totally different from selling only parts of it, through a fit *distribution channels*, the business is required to set-up a fit *infrastructure*, and eventually call upon some *partners* within contractual *agreements* in order to obtain missing resources. Some of our previous work (see [3]) proposed to describe, in addition, the (economic) motivation for each partner involved in the model as a set of *compensations* provided against his participation. A compensation may be financial (money), material (resource, information) or less tangible (market knowledge, trust…). This concept has been introduced to bridge the gap with the value-oriented work of Gordijn (see [5]).

The concepts of the business model, as described above, are shown in the form of a UML class diagram in Fig. 1.

We depict, in [3], a business model from the perspective of a single enterprise, highlighting its environment and concerns for facing a particular demand and turning it into profit.

A new business model is, from our point of view described in [4], defined according to the value proposition it sustains. Any change in the model, as involving a new actor, another product feature, or the use of another resource, defines a new business model *if and only if it matters to the customer* (and hence changes the value proposition).

Let us take the case that a company justifies a high product price with the saving of jobs in his local country where it claims that it recruits only local labour force. If a customer that bought the product phones up the company help-desk and finds out that he speaks with a global customer support in Asia that may change his perception of the product value. In this case, the promise of a local production is seen by [4] as being part of the value proposition.

A modification in the model that is of no interest to the customer (i.e., that is not visible in the value proposition, as any of its offerings) does not define a new business model, but only an alternative way of doing the same business.

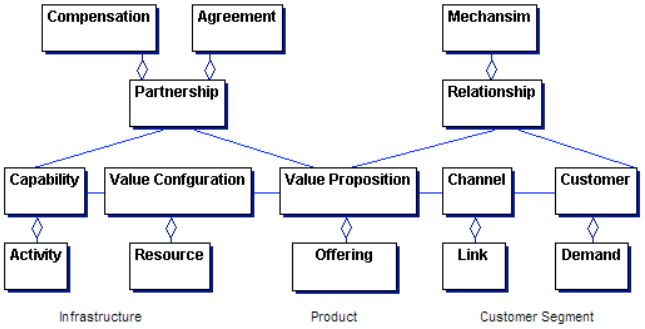


Fig. 1 Business Model Ontology: the core pillars

B. *Value web model definition*

Other approaches, following [5], focus on the value constellation*,* as the set of *value transfers* that holds between all the actors involved in a business model. This *value web* describes the value objects that agents exchange in different value transfers, and the retribution they earn therefore.

An *actor* is perceived by its environment as an independent economic entity. A *value object* is something of value to an actor, such as goods, services, money and even consumer experiences. An actor uses a *value port* to show to its environment that it wants to provide or request value objects. A value port has a direction, inbound or outbound. A *value interface* consists of inbound and outbound value ports of an actor. Groupings of inbound and outbound ports model economic reciprocity: a value object is delivered via one value port and another value object is expected in return via another value port. A *value exchange* connects two value ports of opposite directions of different actors with each other. It represents one or more potential trades of value objects between these value ports. The main concepts of a value web (based on *e*^{3}*-value* models) are described in Fig. 2.

Such an approach encompasses the whole network of partners that realize the business model, with respect to the business model (above) that focuses on a particular enterprise. Any modification in the network (adding an actor or a new value transfer) defines another value web. We may therefore define an equivalence relation amongst value webs that partitions them according to the business model they fulfil, that is: the value transfers that involve the final customer(s), namely the *value proposition* they provide.

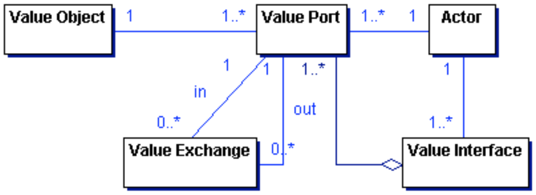


Fig. 2 Value web meta-model

C. *Business process model*

A business process, as outlined by [2] or [5] and used in the Efficient toolset to design and validate business transactions (see [7]), describes a sequence of activities contributing to the fulfilment of some goal of the business model, often by producing a result output. Therefore, it focuses on:

- The flow of value objects exchanged and its ordering.
- The flow of supportive information, facilitating the coordination and communication among the business partners of the process.

We focus in this article on the sequencing of the value transfers, and leave the harvesting of the flow content for a latter work, even if some paths are already suggested in [4].

D. *Binding value web and processes*

In order to design a sound process model that would match the value web implementing the business model, we complement, in [6], the value web described above by a so called activity dependency model, which identifies, classifies, and relates activities needed for executing and co-ordinating value transfers. By imposing dependencies on activities, we (weakly) constrain the succession order of value transfers. Several types of dependencies are proposed including trust, flow, and duality dependencies. Trust dependencies express that one value transfer has to be carried out before another as a consequence of low trust between the involved agents; an example could be a down payment. A flow dependency expresses that a resource obtained from one activity is needed as input to another activity. A duality dependency relates two reciprocal value transfer activities. These dependencies can be declaratively stated, have a clear business motivation, and can be used to (partially) derive a process model from a value web model.

E. *Relating business model and processes*

The complementary work of [4] defines risks and risk management as important forces for process design and management. Risks are defined on flows of different kinds, such as financial flow and goods flow, namely on any kind of *value object transfer*. Typical risk mitigation instruments are identified that are publicly available (depending on the type of flow) and may be used to tackle important risks. On Financial flow, we may as an example, cite the well-known Letter of

Credit, which completely excludes the risks of non-payment and non-delivery for both parties by the introduction of trusted intermediaries. Less expensive alternatives include payment of open account, payment in advance or cash payment; see [4] for more.

Some of the activity dependencies of [6] are related to the mentioned risk mitigation instruments, e.g. a state of lack of trust between two agents involved in a value transfer may result in a change/extension of the corresponding value web and/or process model in terms of a certain order of payment, a down payment etcetera.

We will discuss how risk mitigation instruments may be used in this manner in the design of business and process models in the following section.

III. Integrated Methodology

The methodology we are applying consists of the following three steps (see Fig. 3).

1. Construction of business model.
2. Partial derivation of a value web model from the business model
3. Detailing the value web model into process model

The approach we suggest facilitates requirements elicitation for business and process design by enabling designers to express requirements on the right level. Instead of expressing requirements as detailed and low-level constraints on activities in specific business processes, designers can formulate the requirements in business-oriented terms. These requirements will then, through the proposed methodology, provide a basis for the process design.

For the first step, construction of business models, we adopt the methodology proposed by Osterwalder [1]. For the second step, the construction of value web model, we argue that it can be partially derived form the business model. Finally, for the third step, i.e. the transition to process model we are relying on the methodology proposed in [6].

In the first part of this section the relationship between business and value web models is described, in order to clarify the partial derivation of a value web model from a business model. The second part of the section exemplifies how risks mitigation management influences the business models, value web models and process models. Based on this, at the end of the section, a refined methodology consolidating the risk mitigation instruments into the three-step methodology outlined above is proposed. This consolidation is shown in Fig. 3.

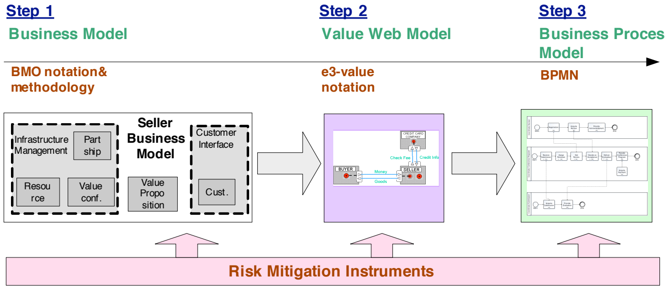


Fig. 3 applied methodology

A. *Comprehensive Ontology*

An attempt to integrate all the concepts evocated above is described in Fig. 4, where we link between the concepts at the heart of the business model ontology (see section II.A) on the one hand, and value webs with risk mitigation instruments on the other (see section II.D). This integrated view highlights the contribution of the *Partnership* in describing, in the business model, the notions seen in the value web as *value transfers* between *agents*. *Resources* and *compensations* are both considered as *value objects* exchanged during such transfers. Partnerships, moreover, might describe soft goals and risks, as explained in [4].

The BMO concepts relevant to the focus of the value web are mainly described in the i*nfrastructure* pillar of the model. This pillar details the *Value Configuration* required for being able to provide a given value proposition to the customer. This configuration consists of a set of required *resources*, and *partnerships* eventually established to provide these latter. Economic *Compensations* are given to partners, against their contribution that generally consist of money or resources, but may be less tangible (as strategic information, or trust).

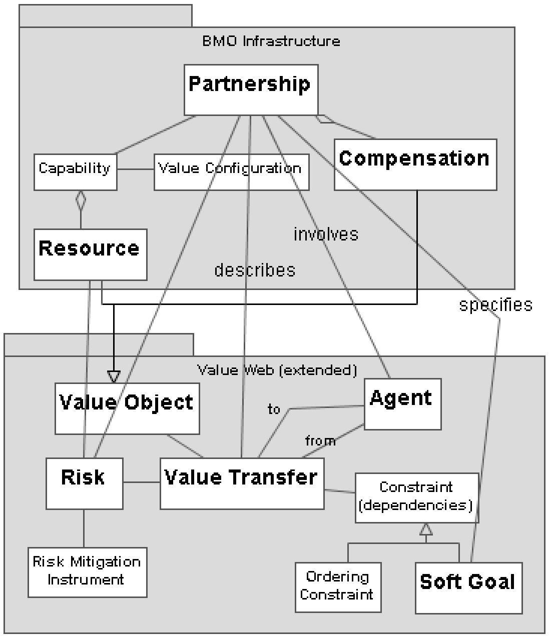


Fig. 4 Chaining Ontology

B. *Deriving value web from a business model*

The value transfers of a value web can be derived from the business model. The following elements of a business model allow the partial creation of a value web that corresponds to the business model, as already shown in Fig. 4:

- Partnerships that provide resources necessary for the manufacturing of the value proposition
- Compensations that flow to the partners that provide resources.
- The delivery of the final value proposition to the end-customer

C. *Risk Mitigation Instruments*

Risks and the handling of risks through risk mitigations instruments influence a business process. Three different categories of risk mitigation instruments are identified: risk mitigation instruments that influence the business model, i.e. the value proposition; risk mitigation instruments that do not influence the business model but the value web model only; and risk mitigation instruments that influence the process model only, i.e. neither the business model nor the value web model. These are illustrated through an example.

Suppose we have the very simple base line case of a Customer purchasing goods from a Seller. A very simple process model in BMO notation (in which for simplicity the financial aspect is left out) and a value web model in *e*^{3}*-value* notation are drawn in figure 5 and 6 respectively, they state the two ends of the refinement proposed in the second step of the methodology. For the sake of readability the BMO illustration (Fig. 5) recalls the main concepts of the model in shaded boxes and the instances of our current example as white boxes.

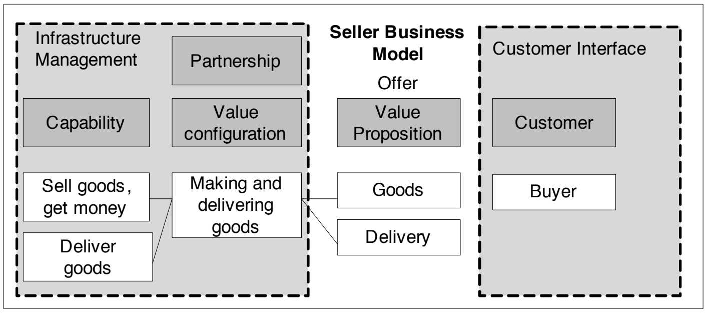


Fig. 5: business model of our simple case

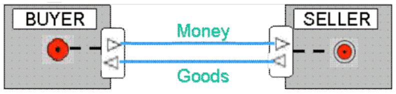


Fig. 6: corresponding value web model

a) *Risk Mitigation Instruments that influence the Business Model*

In a sub-ideal situation there is a risk that the goods may not be delivered properly. Depending on the situation, the buyer, the seller or both could choose to take or offer an insurance against this. Assuming that the Seller offers insurance for this, the business model will change as an insurance offer is added to the value proposition from Seller to Customer (see Fig. 7). This is further exemplified in the *e*^{3}*-value* model of Fig. 8, where the Value Interface between Seller and Buyer is now changed with respect to the base line business model (the new Actor Insurance Company may be part of the new business model or be omitted).

Other risks instruments exist that do change the exact terms of the Value Proposition, and hence define a new business model, for instance by adding features to the product finally delivered to the customer

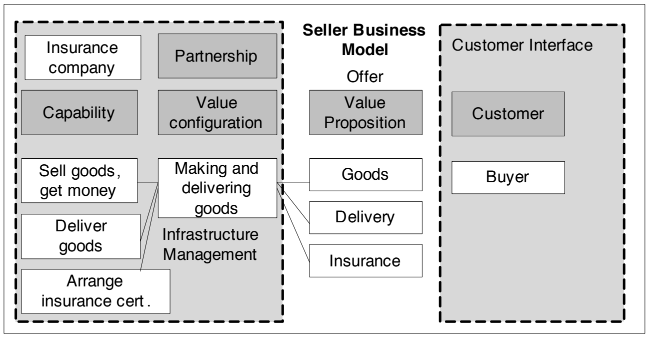


Fig. 7: BMO, extended business model for insurance

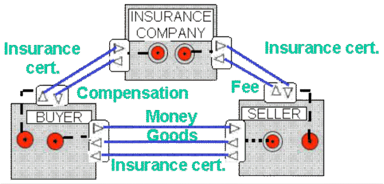


Fig. 8: *e*^{3}*-value* model, extended value web for insurance

b) *Risk Mitigation Instruments that influence the Value Web Model*

Identified risks may also be tackled in a way that is transparent to the final customer, that is: without modifying any term of the product he gets.

For instance, another risk is that the buyer may not pay. To mitigate this risk the seller may make a credit check of the buyer before accepting the order. This means there is an additional actor involved, namely the credit company. Technically, indeed, it does not influence the value proposition from seller to buyer and therefore the business model is considered to be unchanged. Such a risk mitigation instrument is introduced at the level of value-web.

See Fig. 9, (in which the only difference from the model in Fig. 5 is the introduction of a new responsibility/capability in the infrastructure management). However, the presence of a new actor certainly changes the value web model, where the actors involved in a business process provides a basic element of the approach. This change is depicted in Fig. 10.

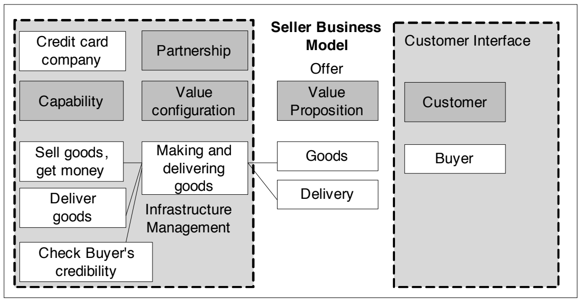


Fig. 9: BOM, business model for credit check

c) Risk Mitigation Instruments that influence the Process Model.

Some risk mitigation instruments only involve additional communication among actors already existing with regard to the base line business model. Some examples of such instruments are sending reminders or notifications. These risk mitigation instruments are also implementation alternatives for the base line business model but in this case *no* new actors or value transfers are added neither to the base line business model nor to the value web and hence are not represented in *e^3-value* models.

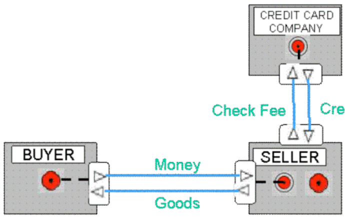


Fig. 10 *e^3-value* model, extended value web for credit check

D. Integrated methodology

According to the different risk groups described above, the following method for including risk mitigation instruments into the business to process modelling method, as illustrated in Fig. 3 and discussed in the beginning of this chapter, is suggested:

1. Start by constructing a business model that identifies the main customers and the value proposition offered to them (including compensations).
2. For each value proposition, identify all risks that may occur and determine whether they should be managed or not.
3. For each risk to be managed, determine what risk mitigation instrument to use.
4. For each risk mitigation instrument to use, determine whether it can be modelled in the business model and if so, extend the business model accordingly.
5. Derive a partial baseline value web from the business model.
6. For each value transfers, identify all risks that may occur (and that are not already handled in the business model) and determine whether to manage them.
7. For each risk to be managed, determine which risk mitigation instrument to use, some instruments might not be available from soft goals.
8. For each risk mitigation instrument to use, determine whether it can be modelled in the value web and if so, extend the value web accordingly.
9. Construct a process model that is compatible with the extended value web, extended to handle each risk mitigation instrument that was not handled in the business model (step 5) or the value web (step 10), applying, for instance, the procedure detailed in [6].

Alternative process models may be designed for the same value web by selecting other risk mitigation instruments in steps 4 or 9.

This methodology should enable requirements expressed in business-oriented terms to be used at the right moment in the process design, while staying aligned with the practical solutions available in the current business environment.

IV. CONCLUSION AND FUTURE RESEARCH

In this position paper we have introduced some systematic approach regarding the understanding of the "why" behind a process model in terms of business model achievement. We have therefore briefly outlined how risk analysis and risk mitigation instruments can be used for the purposes of business and process modeling. The approach is based on a methodology that suggests a systematic and stepwise development of a process model from a business model.

The natural continuation of this work is, hence, a thorough definition and test of this first draft of integrated methodology. One part of our future work will be related to the evaluation of the fitness of these techniques for reasoning about "business goals".

Other interesting areas for further work are:

- Reverse engineering application (from existing process models to a value web and a business model)
- Evaluate the pertinence of business models: the engineering process described in our methodology is an accurate means of highlighting goals conflicts between the actors involved in the business model. Such models, with heavy contradiction, should maybe not be driven further by the business.
- Define/reveal new business models by adding value-full features to existing ones. These features are revealed during the iterative methodology, since the introduction of alternative risk mitigation instruments inherently adds new features to the business model (as the adding of the insurance company, in Fig. 7 adds value onto the former value proposition). Such features might be an input for further business development.

Finally, it is worth mentioning the work of [8], where an approach on trust management is proposed. It is related to ours in the sense that lack of trust or low trust on some partners can be considered as a risk. However, we also consider some risks, such as currency fluctuation, damage to the goods, etc, that are not related to the level of trust that holds between the business partners. Furthermore, while the approach in [8] focuses on checking the consistence of a process model against the relevant business model, we are also dealing with the construction of a process model based on a business model.

REFERENCES

[1] Osterwalder A. (2004) *The Business Model Ontology*, Ph.D. thesis 2004, HEC Lausanne. Available: http://www.hec.unil.ch/aosterwa/

[2] Porter, *Competitive Advantage*. New York, Free Press, 1985.

[3] Schmitt M., Grégoire B., Ramel S., Incoul C., Brimont P., Dubois E.. (2004) *If business models could speak! Efficient: a framework for appraisal, design and simulation of electronic business transactions*. International Conference on Enterprise Integration and Modelling Technology (ICEIMT'04), Toronto. Available: http://efficient.citi.tudor.lu/

[4] Schmitt M., Grégoire B. (2005) *Risk Mitigation Instruments for Business Models and Process Models*, submitted for REBNITA'05. Available: http://efficient.citi.tudor.lu/

[5] Gordijn J., Akkermans H., van Vliet H. (2000) *Business Modelling Is Not Process Modelling*, ER Workshops 2000, 40-51. Available: http://www.cs.vu.nl/~gordijn/research.htm

[6] Andersson B., Bergholtz M., Edirisuriya A., Ilayperuma T., and Johannesson P., "A Declarative Foundation of Process Models", to be presented at 17th Int. Conf. on Advanced Information Systems Engineering (CAiSE05), Porto, Portugal, June, 2005

[7] Mammar A., Ramel S., Grégoire B., Schmitt M., Guelfi N.. *Efficient: A Toolset for Building Trusted B2B Transactions*. Accepted for 17th Conference on Advanced Information Systems Engineering (CAISE'05), Porto, June 2005.

[8] Wieringa R and Gordijn J., *Value Oriented Design of Service Coordination Processes: Correctness and Trust*, Accepted by the ACM Symposium on Applied Computing, Organizational Engineering Track, 2005.