Knowledge-driven Migration to Services

Maryam Razavian

2013
The research reported in this thesis has been carried out under the auspices of SIKS, the Dutch Research School for Information and Knowledge Systems.

The research in this thesis has been sponsored by the Dutch Joint Academic and Commercial Quality Research and Development (Jacquard) program on Software Engineering Research via contract 638.001.206 SAPIENSA: Service-enAbling PreexIsting ENterprISe Assets.

This research was partially funded by FP7 NoE S-Cube project via contract 215483.

Promotiecommissie:
dr. E. di Nitto (Politecnico di Milano)
prof.dr. G. Engels (University of Paderborn)
prof.dr. M. Aiello (University of Groningen)
prof.dr. W.J.A.M. van den Heuvel (Tilburg University)
dr. E. Poort (CGI)
prof.dr. H. Akkermans (VU University Amsterdam)


Copyright © 2013, Maryam Razavian
All rights reserved unless otherwise stated
Cover design by Afsaneh Rostami and Uitgeverij BOXPress
Published by Uitgeverij BOXPress, ‘s-Hertogenbosch
Typeset in L\TeX\ by the author
Knowledge-driven Migration to Services
promotor: prof.dr. J.C. van Vliet
copromotor: dr. P. Lago
# Contents

1 Introduction ................................. 1
   1.1 Context and Groundwork .................. 2
      1.1.1 Migration to Services Approach .... 2
      1.1.2 Knowledge and Knowledge Management 3
   1.2 Research Questions ..................... 4
   1.3 Thesis in a Glance ..................... 6
   1.4 Research Methods ...................... 7
   1.5 Outline of Thesis and Publications .... 8

2 Understanding SOA Migration ................. 11
   2.1 Introduction .......................... 11
   2.2 Running Example ....................... 12
      2.2.1 SMART Approach to Migration of Flight Booking System 12
      2.2.2 Sneed’s Approach to Migration of Flight Booking System 13
   2.3 The SOA Migration Framework .......... 14
      2.3.1 Conceptual Elements of SOA-MF .... 15
      2.3.2 SOA-MF Description ............... 18
      2.3.3 Tiers ............................. 20
   2.4 On the Role of SOA-MF ................. 21
      2.4.1 Mapping of SMART Approach on SOA-MF ... 22
      2.4.2 Mapping of Sneed’s Approach on SOA-MF .... 22
      2.4.3 What the Mappings Imply .......... 23
   2.5 Conclusion ............................ 25

3 A Systematic Review of SOA Migration Approaches 27
   3.1 Introduction .......................... 27
   3.2 Two-View SOA Migration Approach .... 28
   3.3 Research Method ....................... 29
      3.3.1 Pilot Study ....................... 30
      3.3.2 Data Extraction ................... 34
      3.3.3 Data Analysis Method ............. 34
   3.4 Results ............................... 38
      3.4.1 Overview of the Included Studies ..... 38
      3.4.2 RQ-I What activities are carried out in SOA migration? 39
      3.4.3 RQ-II What is the sequencing of the activities? .... 44
      3.4.4 RQ-III What are the knowledge elements that are used and produced? 45
   3.5 SOA Migration Frame of Reference ...... 55
      3.5.1 Frame of Reference Application Example 57
## CONTENTS

3.6 Discussion ........................................... 62
   3.6.1 Threats to Validity .............................. 62
   3.6.2 Additional Observations ......................... 63
3.7 Conclusion .......................................... 66

4 On The Differences Between Academic and Industrial Migration Approaches 67
   4.1 Introduction .................................... 67
   4.2 Design of Interview Survey Study ................. 68
       4.2.1 Research Questions .......................... 68
       4.2.2 Study Design ................................ 68
       4.2.3 Data Analysis ................................ 69
   4.3 Results ........................................... 70
       4.3.1 Migration Activities ......................... 70
       4.3.2 Sequencing of Migration Activities .......... 73
       4.3.3 Legacy Understanding through Personalization 74
       4.3.4 Service Extraction by Defining the Ideal Services 75
   4.4 Discussion ........................................ 76
       4.4.1 Threats to Validity .......................... 77
   4.5 Conclusions ...................................... 78

5 A Lean & Mean Approach for SOA Migration 79
   5.1 Introduction .................................... 79
   5.2 Industrial Migration Approach .................... 80
       5.2.1 Core Activities .............................. 80
       5.2.2 Core Knowledge .............................. 84
   5.3 A Lean & Mean Migration Approach ................. 86
       5.3.1 How Core Activities Benefit Practice ....... 86
       5.3.2 How Core Knowledge Benefits Practice ....... 87
       5.3.3 Specification of the Lean & Mean Migration Approach 88
   5.4 Threats to Validity ................................ 90
   5.5 Conclusions ...................................... 92

6 An Industrial Study on Lean & Mean Migration 95
   6.1 Introduction .................................... 95
   6.2 Industrial Case Studies ............................ 97
       6.2.1 Case Descriptions ............................ 98
   6.3 Lean & Mean Migration Approach: A Treatment For Efficiency 98
   6.4 Study Design .................................... 99
   6.5 Lean & Mean Approach for Data Migration ......... 101
       6.5.1 Core of Lean & Mean-DM ..................... 102
       6.5.2 Extensions .................................... 107
## 6.6 Implications of Lean & Mean Migration Approach (RQ-I)

### 6.6.1 Increased Reuse

- (RQ-I)

### 6.6.2 Improved Problem-Solving

- (RQ-I)

### 6.6.3 Supporting Existing Way-of-working

- (RQ-I)

## 6.7 Generalizability of the Lean & Mean Migration Approach (RQ-II)

### 6.7.1 Core Activities

- (RQ-II)

### 6.7.2 Core Knowledge

- (RQ-II)

## 6.8 Threats to Validity

- (RQ-II)

## 6.9 Discussion and Conclusion

- (RQ-II)

## 7 Change Viewpoint: An Extension for Dealing with Change

### 7.1 Introduction

- (RQ-II)

### 7.2 Context and Groundwork

- (RQ-II)

### 7.3 Nation-wide Electronic Health Record

- (RQ-II)

### 7.4 Understanding EHRS using Knowledge View

- (RQ-II)

### 7.5 Using Change Viewpoint to Deal with Changes

#### 7.5.1 Architects’ Mental Model of How to Deal with Change

- (RQ-II)

#### 7.5.2 Change Viewpoint

- (RQ-II)

#### 7.5.3 Change Viewpoint: Metamodel and Conforming Notation

- (RQ-II)

### 7.6 Applying the Change Viewpoint in EHRS

#### 7.6.1 Scenario 1: Scoping Change in Problem Space.

- (RQ-II)

#### 7.6.2 Scenario 2: Design Decision Impact Evaluation.

- (RQ-II)

### 7.7 Discussion

- (RQ-II)

### 7.8 Related Work

- (RQ-II)

### 7.9 Conclusion

- (RQ-II)

## 8 Conclusions

### 8.1 Contributions

#### 8.1.1 RQ-1: How to understand and characterize SOA migration?

- (RQ-II)

#### 8.1.2 RQ-2: How is SOA migration perceived in academic research?

- (RQ-II)

#### 8.1.3 RQ-3: How is SOA migration perceived and applied in industrial practice?

- (RQ-II)

#### 8.1.4 RQ-4: How to guide SOA migration so that specific problems and needs of practice are addressed?

- (RQ-II)

### 8.2 Future Research

- (RQ-II)

#### 8.2.1 Enriching SOA Migration with Knowledge Management

- (RQ-II)

#### 8.2.2 Industry-relevant SOA Migration

- (RQ-II)

## Appendix A Overview of Primary Studies of the Systematic Literature Review

- (RQ-II)

## Appendix B Overview of Interview Survey Results

- (RQ-II)
Acknowledgements

First and foremost, I sincerely would like to thank my supervisors Patricia Lago and Hans van Vliet. Patricia, I am very grateful for your constant supervision, mentoring and encouragements. Looking at you, I learned what it means to be a scientist: carrying out the research that matters, making my work meaningful to others, and realizing that timing is everything. I cannot thank you enough for showing me how joyful research can be. I would also like to thank Hans, who always encouraged me to see the “big picture” and to find the “right message”. I greatly appreciate your efforts, guidance, and of course all “multi-culti” dinner parties. I am also very grateful to my thesis committee, for their encouragement, insightful comments, and questions.

In performing this research, I had the opportunity to collaborate with many talented colleagues in the Sapiens and S-CUBE projects. In Sapiens with Dinh Khoa Nguyen and Willem-Jan van den Heuvel, and in S-CUBE with Elisabetta di Nitto and Ita Richardson. I would also like to thank my fellow labmates: Christina Manteli, Qing Gu, Damian Andrew Tamburri, Ivan Razo Zapata, Pieter de Leenheer, Jaap Gordijn, Remco C. de Boer, Rik Farenhorst, Rahul Premraj and Adam Vanya. I particularly want to thank Antony Tang for his guidance on my research work, Victor Clerc for writing the Dutch summary of my thesis, and Maurizio Morisio for his support during our collaboration in Italy. I also thank Elly Lammers and Caroline Waij whom without them life in the university and integrating in the Netherlands would have been much more difficult. Finally, I am thankful to Parisa Zahedi whose Master’s thesis research contributed to parts of this thesis.

The research reported in this thesis could not have been done without the fruitful collaborations with our industrial partners. Special thanks go to Anne-Marie Brooijmans, Edwin van Dis, Ton Mulders, Ronald Evers, Karel de Smet and Wouter Paul Trienekens.

There are many friends who helped me survive my Ph.D. years. Nikoo and Ghazaleh, I am thankful for all the support and encouragements over these years. I would like to thank my parents, Afsaneh and Mehdi, for teaching me how to look for the work I love, and to believe in myself. I am grateful to my sister, Nassim, who by moving to the Netherlands brought me a lot of joy. There is one person who I cannot imagine being here without. Nima, you listened to all my research ideas, you reviewed all my papers, you inspired me in many ways, and I cannot wait to take the next steps with you.

February 2013

Maryam Razavian
Introduction

Service Orientation is having an impact on application development similar to that brought by Object Orientation in the nineties (Cockburn, 1993). Both paradigms went through the peak of inflated expectations coined in the Gartner Technology Hype Cycle. While Object Orientation reached mainstream adoption long ago, Service Orientation is still, in the slope of enlightenment (Gu and Lago, 2009b; Fenn, 2010) and the real benefits of service-oriented architecture (SOA) are now becoming more widely understood. One of those benefits is rehabilitating legacy assets by migrating them to services.

Service Orientation promises to rehabilitate pre-existing legacy assets by encapsulating them as added-value services. Such services embed business functions from legacy assets on diverse hardware and software platforms. Some of those legacy assets may be legacy systems while others could be technically-healthy and value-adding applications, business processes, or data of an enterprise. Migrating those legacy assets into services that can smoothly operate with modern technology has become one of the major challenges of service engineering methodologies (Papazoglou et al., 2006).

Migration to services has received much attention in both academia and practice. Enterprises perceive many values in service-enabling their legacy assets such as achieving the advantages offered by SOA while reusing the embedded functionalities in the legacy assets. Assumption behind this is that those legacy functionalities are valuable for the enterprise for instance due to their core position in the market. According to Gartner (Scholler, 2012), enterprises migrate their legacy elements for three reasons: (i) to retain legacy applications indefinitely due to their core position in the market, while coping with ever-changing requirements (ii) to improve business process efficiency and agility by integrating monolithic legacy systems (iii) to move to new delivery solutions such as software-as-a-service (SaaS). To achieve these goals, those enterprises have to deal with many challenging problems, such as data model inconsistency, functional overlap, and architectural and platform inconsistency. Moreover, in carrying out migra-
CHAPTER 1. INTRODUCTION

tion projects, various demands for high quality, fast delivery, and decreasing costs have to be addressed. Hence, migration projects remain challenging, despite the enterprises extensive experience. To cope with those challenges, well-defined approaches for migration to services are needed.

In academia it is often perceived that sufficient methodologies and practices have been developed such that SOA migration is close to main-stream adoption by industrial practice. In spite of what academics think, we observed that practitioners still face difficulties in consolidating to a successful yet cost-effective SOA migration approach. The many available methods often prove to be too abstract or lacking sufficient support to be applicable. Consequently, practitioners define their own migration approaches in an ad-hoc manner.

This problem is addressed in SAPIENSA (Service-enAbling Pre-exIsting ENTERpriSe Assets), a joint research project of the VU University Amsterdam and Tilburg University. SAPIENSA has the main purpose of providing SOA migration approaches to support elicitation of legacy assets and transformation of them to added value services. This thesis is one of its results.

1.1 Context and Groundwork

1.1.1 Migration to Services Approach

An approach for migration to services states a path from the As-Is state (i.e. As-Is legacy assets) to the To-Be state (i.e. To-Be services). Such a migration approach provides guidelines, models, best practices, standards and reference architectures necessary to move from the As-Is state to the To-Be. Although SOA (Service-Oriented Architecture) is an architectural style, it is however commonly used to represent the broader notion of a service design approach or even Service Orientation paradigm. In this thesis we refer to the approach for migration to services as “SOA migration” and by SOA we mean the broader notion. More precisely, with the term SOA migration we mean an approach that frames what you do to transform legacy assets into service-oriented software. With term service-oriented software we mean both software services, and service-based applications (SBAs) reusing (i.e. composing) them.

We consider migration to services as a specific form of software reengineering in which the To-Be software is of type services. As with the definitions on software reengineering, there is no universally definition of SOA migration. In this thesis we use the definition of Chikofsky and Cross II (1990) and customize it for migration to services as follows: the examination and alteration of legacy assets to reconstitute it in the new form of services and subsequent implementation of service-enabling those legacy assets.

This definition embraces the following two generic types of SOA migration:
(i) activities related to legacy understanding and porting existing functionality, such as reverse engineering legacy assets and wrapping them as services, and (ii) activities geared towards software evolution and modernization, such as forward engineering new services while reusing legacy assets. Considering that our definition addresses both types, in this thesis the terms redevelopment, renovation, wrapping, and modernization all fall under the umbrella of SOA migration.

1.1.2 Knowledge and Knowledge Management

Developing services from the existing legacy assets requires deep technical knowledge in many specific aspects of both legacy assets and target services. Other forms of knowledge related to processes or human expertise also play a key role in migration. Migration, similar to any type of software development, is a knowledge-intensive work. Such knowledge-intensive work can be improved by learning how to manage knowledge better, i.e. the basic idea behind knowledge management [Dingsøyr and van Vliet 2009]. This thesis incorporates knowledge management in SOA migration.

The Oxford Dictionary defines knowledge as specific information, facts or intelligence about something. Here, the term knowledge is used to describe the whole spectrum of content for the following concepts concerning the migration: data, models, procedures, techniques, principles, and context. Nonaka and Takeuchi (1995) refer to two main types of knowledge, tacit and explicit. Tacit knowledge is a knowledge that a human is not able to express explicitly, but is guiding the behavior of the human. Explicit knowledge is knowledge that we can represent, for example in reports, books, talks, or other formal or informal communication.

Knowledge management has received much attention in various fields including software engineering. One of the most prominent works on knowledge management is the one by Nonaka and Takeuchi (1995). In their work they present a model of how knowledge is transformed and converted in organizations. Hansen et al. (1999) introduce two main strategies for knowledge management: (i) codification, to systematize and store information that constitutes the knowledge of the company, and to make this available to the people in the company for reuse, (ii) personalization, to support the flow of information in a company by keeping information about knowledge sources, like a yellow pages of who knows what in a company.

In software engineering, there are many approaches to knowledge management, ranging from focusing on codifying knowledge in different forms, like describing architecture, to focusing on knowledge reuse and sharing (Aurum et al. 2003, Babar et al. 2009). A well-known example of knowledge management in software engineering is the “Experience Factory” that supports reusing life cycle
CHAPTER 1. INTRODUCTION

experience, processes and products for software development (Boehm, 1999). In this framework, experience is collected from software development projects, and is packaged and stored in an experience base. Much of knowledge management work in software engineering, however, mainly focuses on codifying knowledge in different forms (Babar et al., 2009).

To date, there are limited results in identifying the knowledge relevant for Service Orientation, let alone SOA migration. Those few works in the area of Service Orientation mainly focus on disciplined design decisions (Zimmermann et al., 2007, 2009) and compliance to SOA patterns (Erl, 2009). As such, the related knowledge mostly captures reusable technical solutions, while the related rationale or decision process that led to the solution is lost.

The combination of knowledge management and SOA migration appear to be fruitful; typical types of knowledge that shape and drive migration could be identified and the challenges and issues involved in migration could be addressed using knowledge management strategies. If suitable knowledge management strategies can be incorporated into migration approaches, they can be particularly helpful for service engineers to provide necessary foundation for guiding migration decision making.

1.2 Research Questions

As noted, although Service Orientation is often promoted as the solution for leveraging and modernizing legacy assets, migration to services remains a challenging and complex task. Such complexity amounts to the shift in the ways we conceive services, compared to the ways legacy assets are already developed: from a large system to a set of small pluggable services, where services are neither owned nor always part of a “monolithic” system. Services are well-specified, loosely coupled, and cohesive pluggable elements, whereas legacy assets are often tightly coupled, not cohesive, and support multiple business functionalities. Services need to be designed under open world assumption, and be distributed across organization boundaries, whereas legacy assets are often owned and controlled by a single owner. Those inherent distinctions make the service-enabling of the legacy assets to be a complex and demanding task. Under such a complex environment, the ways migration is guided can greatly influence the outcome of the migration. Hence, we formulate our central research (RQ) question as follows:

RQ: How can SOA migration be guided?

This thesis is about guiding SOA migration. In order to arrive at useable and successful approaches to SOA migration, we first need to understand what SOA migration approaches entail. Obtaining such an understanding in an emerging
and still fuzzy research field like SOA migration, however, is difficult. To provide such an understanding, it is important to identify the elements that support conceptual characterization of SOA migration approaches. Our first research sub-question is thus:

**RQ-1:** *How to understand and characterize SOA migration?*

After we have gained an understanding of what SOA migration entails, we must understand and characterize the existing migration approaches. For addressing the central research question, we aim at understanding the-state-of-the-art in SOA migration approaches and the categories of the approaches that exist in the field. This leads to our second research sub-question:

**RQ-2:** *How is SOA migration perceived in academic research?*

In this thesis, we are not merely interested in the migration approaches in academia; we aim to explore the migration approaches adopted in industry, too. We seek for comparing and contrasting the academic research against practice, and learn what industry-relevant research is still needed in order to bring SOA migration to maturity. In addition, we aim at understanding the typical problems and needs that industrial practice faces. Hence, the third research sub-question is:

**RQ-3:** *How is SOA migration perceived and applied in industrial practice?*

Like today’s software developments, SOA migration projects are faced with steadily increasing demands for efficiency: migration has to be carried out faster, better and cheaper. At the same time, migration complexity increases due to challenges in migration such as data model inconsistency, functional overlap, and architectural and platform inconsistency. On the other hand despite the companies’ extensive experience, past experiences are not adequately reused and SOA migration is still carried out in an ad-hoc manner. To cope with such demands and complexities, migration needs to be guided using a well-defined migration approach. For instance, practitioners need to be guided about the essential activities and techniques as well as the fundamental inputs and outputs.

Based on our research of what practitioners do when doing migration and what their problems and need are, we focus on how to guide SOA migration. To fully understand what kind of guidance practitioners need, we start with identifying what type of approach is favored in carrying out migration. This enables us to guide migration in ways that address the problems of industrial practice and is tailored to practice needs. Thus, our fourth research sub-question is:
**RQ-4:** How to guide SOA migration so that specific problems and needs of practice are addressed?

### 1.3 Thesis in a Glance

Our research context is schematically depicted in Figure 1.3. This research started by defining a *SOA Migration Framework* aimed for understanding and characterizing different migration approaches (RQ-1). Using the framework we carried out a systematic literature review (SLR) on SOA migration in which we categorized *Academic Approaches* (RQ-2). To gain an understanding on how industrial practice perceives SOA migration (RQ-3), we further conducted an *Industrial survey* in seven SOA solution provider companies. Results showed that the industrial migration approaches (*Industrial Approaches* in the figure) are considerably different from the academic ones. The differences between academic approaches and industrial ones motivated us to seek for a deeper understanding of industrial migration approaches. By further analyzing the results of the interview survey, we generalized the practice of industrial migration into a *Lean & Mean SOA migration approach*. The Lean & Mean approach was devised as a general tool to guide and steer migration projects (RQ-4). To address specific problems and needs of practitioners, a Lean & Mean approach could be extended. We applied the Lean & Mean approach in two major industrial studies: a Technical Action Research (*TAR* in the figure) and a *Case Study*. Using the two industrial studies we devised two extensions for the Lean & Mean approach: (i) *Extension for practice reuse*, and (ii) *Extension for dealing with change*. In this way, we addressed two main problems of industrial practice related to SOA migration: (i) how to reuse past experience, (ii) how to deal with changes during migration.

![Figure 1.1: Research context of this thesis: research question and research methods](image)

Figure 1.1: Research context of this thesis: research question and research methods
1.4 Research Methods

In this thesis, we have used a number of qualitative research methods that are common in the field of software engineering research. In the following, we briefly introduce these methods.

- **Systematic literature review.** Also called systematic review, this method is an evidence-based approach to thoroughly search studies relevant to some pre-defined research questions and critically select, appraise and synthesize findings to answer such research questions (Kitchenham, 2007). It is particularly powerful in collecting and analyzing existing work, which is a common task in establishing background knowledge in any research. This method has been used in exploring and categorizing SOA Migration approaches (Chapter 3).

- **Literature study.** Different from a systematic literature review that follows a pre-defined, structured review protocol, a literature study is much less formal in the sense that it allows more freedom in collecting relevant studies and analyzing their content. Although the results of a literature study might not be as complete and valid as those of a systematic literature review, thanks to its effectiveness and efficiency, this method is often used to gain certain knowledge or understanding on a specific topic. In this thesis, we have used this method to study some existing reengineering methods to define the SOA migration framework (Chapter 2).

- **Interview survey.** This method aims at collecting information on a one-shot basis that is relevant to answer pre-defined research questions (Yin, 2008) and it represents a wide target population. In this thesis in order to understand what type of migration approaches is adopted in industrial practice, we carried out an interview survey (Chapter 4, 5). We chose interviews as the method for gathering survey data because the presence of the interviewer can help clarify queries and can stimulate the respondent to explain more fully. Furthermore, being flexible, interviews are more suitable for explorative studies.

- **Technical action research.** This research method is a nested, cyclical approach where the researcher actively participates in the case studies that (s)he performs. With the goal of collaboratively improving a real situation, and learning from it, we followed a typical technical action research including two industrial case studies (Chapter 6). Using this approach we started by the Lean & Mean approach as our research artifact, and then tested it under conditions of practice by solving concrete practice problems.
CHAPTER 1. INTRODUCTION

• Case study. This method is used in many situations to contribute to our knowledge of individual or organizational phenomena (Yin 2008). Case studies strive to portray what it is like to be in a particular situation, by looking at a case or phenomenon in its real-life context, usually employing many types of data. In Chapter 6 we used two different case studies in the area of data migration in two of action research cycles. Furthermore, in Chapter 7 we carried out a case study on a migration project where an important change occurs during the migration.

Table 1.1 depicts an overview of the research methods we used in each chapter of the thesis.

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Research methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chapter 2</td>
<td>Literature study</td>
</tr>
<tr>
<td>Chapter 3</td>
<td>Systematic literature review</td>
</tr>
<tr>
<td>Chapter 4, 5</td>
<td>Industrial survey</td>
</tr>
<tr>
<td>Chapter 6</td>
<td>Technical action research</td>
</tr>
<tr>
<td>Chapter 6, 7</td>
<td>Case study</td>
</tr>
</tbody>
</table>

1.5 Outline of Thesis and Publications

The research presented in this thesis has either been published previously or is currently under submission. The chapters are based on the following publications.

• Chapter 1 (this chapter): provides a high-level overview of the context of this research and the research problems addressed in the reminder of this thesis. Parts of this chapter has been published as:


• Chapter 2 addresses research question RQ-1; it presents a framework for understanding and characterizing SOA migration approaches. It has been published previously as:
1.5. OUTLINE OF THESIS AND PUBLICATIONS


An extended version of this paper is published as:


- Chapter 3 addresses research question RQ-2; the objective of this chapter is to bring order in the existing SOA migration approaches and facilitate determining ones own migration approach. This chapter has been previously published as:


Parts of this chapter has been submitted as:

- Chapter 4 addresses research question RQ-3; it describes how industrial practice performs SOA migration and contrasts those industrial approaches with academic ones (described in Chapter 3). This chapter has been previously published as:


- Chapter 5 addresses research question RQ-4; it generalizes the practice of industrial migration into a Lean & Mean SOA migration approach. In addition, the uses of the approach pinpoint the needs and problems of industrial migration. This chapter has been previously published as:


- Chapter 6 addresses research question RQ-4; it applies our Lean & Mean migration approach, and evaluate it, in two large industrial case studies in
the field of data migration. The use of the approach in our case studies reveals the benefits of Lean & Mean approach in guiding practitioners for carrying out migration while meeting the efficiency demands. This chapter has been submitted as:


• Chapter [7] addresses research question RQ-4; it describes an extension for Lean & Mean approach intended to guide practitioners to deal with changes occurring during migration. This chapter has been previously published as:

Understanding SOA Migration

This chapter proposes a conceptual framework for the approach of migrating the legacy assets to services. We describe what such migration approach entails and what distinct conceptual elements systematically define the approach. Based on the comprising conceptual elements the framework which is considered as a basis for understanding of different approaches is proposed. Further, the role of the migration framework in positioning and assessing the existing approaches, is discussed. Finally, the procedure for positioning and mapping of migration approaches on the framework is explained using two example migration approaches.

2.1 Introduction

Since the early use of SOA, migration of legacy assets to services has caught a lot of attention. Various studies present an approach for such migration. These studies mainly differ in the way they provide solutions for two challenging problems of “what is migrated” (i.e. the legacy assets) and “how the migration is performed” (i.e. the migration approach). As an example, some studies transform the legacy code to web services using automatic code translation. In such studies the legacy code constitutes “what is migrated” and the code translation represents “how the migration is performed”. Other studies transform pre-existing business processes (i.e. what) using business domain recovery and refactoring techniques (i.e. how). However, there is still little conceptual characterization of what the legacy to service migration approach entails. As a result, a common understanding of the SOA migration is difficult to achieve. Furthermore, the lack of study on the state of the art makes the understanding, comparison and analysis of existing approaches especially difficult. To solve this problem, we need to establish a framework for SOA migration which facilitates achieving a general understanding on SOA migration.

The question we address in this chapter is what this framework entails. As an
CHAPTER 2. UNDERSTANDING SOA MIGRATION

answer, based on the definition of SOA migration, we propose a framework which embraces a holistic illustration of the SOA migration approach, along with the distinct conceptual elements involved in such an approach. This framework facilitates the representation of the SOA migration approaches in a unified manner and therefore provides the basis for their comparison and analysis. As such, different migration approaches may be mapped to a portion of (or all) the framework.

The reminder of the chapter is organized as follows. § 2.2 presents a running example, which aims for clarifying the role of the framework in understanding and positioning the migration approaches. In § 2.3 the proposed SOA migration framework is described. § 2.4 discusses the use of the proposed framework. Finally, § 2.5 concludes the chapter.

2.2 Running Example

To get a feel for how different approaches provide solutions for SOA migration, we present the following example. Let’s consider the hypothetical problem of migrating a flight booking system to SOA. This system supports a number of business processes including “search flight catalogue”, “set traveling preference”, “book flight” and “bill customer”. For this problem, we propose two different solutions based on two different SOA migration approaches including SMART (Lewis et al., 2005) and the approach proposed by Sneed (Sneed, 2006). § 2.2.1 and § 2.2.2 respectively discuss SMART and Sneed’s approach to flight booking migration. Having the same problem domain facilitates better understanding of the commonalities and differences among the selected migration approaches. We will return to this example and use it to clarify different aspects of our proposed framework throughout the chapter. For the sake of simplicity, some details of these two approaches are ignored.

2.2.1 SMART Approach to Migration of Flight Booking System

SMART is an approach to make initial decisions about the feasibility of reusing the legacy assets within an SOA environment, along with an understanding of costs and risks involved. By using interviews and questionnaires, this approach gathers information about legacy components, the target SOA, and potential services. The SMART migration approach entails a number of activities, which are applied in this example and are discussed below. As the first step, SMART defines the primary candidate services from existing business processes. Here, the services which have clear inputs and outputs are considered as candidate services. The next activity of SMART, “describe existing capability”, gathers descriptive
characteristics such as name, function, and size of existing components. In addition, information regarding the target SOA environment is extracted within “describe target environment” activity. In this example, services should provide contracts embracing different security policies of customers billing information. The “gap analysis” activity, extracts the required changes to components code for exposing their functionality as services given the service requirements, the service inputs and outputs, as well as the characteristics and components of the target SOA environment. Based on this, an estimate of the effort required to make these changes are provided. Finally, the strategy, embracing the options for proceeding with the migration effort, along with their associated risk is devised. Table 2.1 illustrates the migration strategy for flight booking system.

<table>
<thead>
<tr>
<th>Service</th>
<th>Migration Approach</th>
<th>Level of Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>get flight catalog</td>
<td>Wrap related functions in the Flight Catalog component</td>
<td>low</td>
</tr>
<tr>
<td>bill customer</td>
<td>Extract the billing functionality from Flight Reservation component; create code for implementing different policies of billing information security</td>
<td>high</td>
</tr>
<tr>
<td>get flight information</td>
<td>Create an interface to Flight Catalog component</td>
<td>low</td>
</tr>
<tr>
<td>get customer information</td>
<td>Create an interface to Customer component</td>
<td>low</td>
</tr>
</tbody>
</table>

### 2.2.2 Sneed’s Approach to Migration of Flight Booking System

The migration approach proposed by [Sneed (2006)](#) starts from identifying candidates for services by performing portfolio analysis and listing out essential business rules. Consider the “bill customer business process” embracing the secondary business rules including “aggregating the billing items”, “computing the sales tax”, “obtaining the customer address data”, “producing the bill”, and “dispatching the bill”. These business rules are candidates to be migrated to services. The second step is to assess the business value of these candidate services. The next step is to extract the existing legacy code which implements these business rules using the code stripping technique. The key idea in this technique is to identify the names of the essential data results and to trace how they are produced. This is achieved via an inverse data flow analysis. The data flow trace may pass through several approaches or procedures in different classes or modules. For instance, in order to extract the bill customer service, several results including “billing items”, “sales tax” and “bill” should be traced back in the existing code. All the modules and classes identified should be combined to formulate the “bill customer” service. Once a code fragment has been identified.
as being a potential "bill customer" service, the next step is to extract it from the system in which it is embedded and to reassemble it as a separate module with its own interface. This is done by copying the impacted code units into a common framework and by placing all of the data objects they refer to into a common data interface. This entails generating a module which reads the input parameters "billing items", "customer info" and "sale tax" from a WSDL input file and writes the result of "billing result" or error message into a WSDL output file. If we compare this approach with previous one, it is apparent that this approach mostly focuses on technological aspects of SOA migration while in SMART mostly migration strategy is formulated.

2.3 The SOA Migration Framework

The SOA migration framework addresses the question of "what does the migration of legacy assets to services entail". In (Bisbal et al., 1999) migration is defined as a modernization technique that moves the system to a new platform while retaining the original system data and functionality. Chikofsky and Cross II (1990) define reengineering as the examination and alteration of a subject system to reconstitute it in a new form and the subsequent implementation of the new form. The commonalities among these two definitions are considerable. In practice, the notions of "legacy migration", "integration" and "architectural recovery", which all deal with legacy applications, are considered as approaches to reengineering. Accordingly, we consider the notion of "migration" as a reengineering problem.

According to (Kazman et al., 1998) any type of reengineering consists of three basic reengineering processes: 1) analysis of an existing system, 2) logical transformation, and 3) development of a new system. In the context of architectural recovery, a conceptual "horseshoe" model has been developed by the Software Engineering Institute, which distinguishes different levels of reengineering analysis and provides a foundation for transformations at each level, especially for transformations to the architectural level (Kazman et al., 1998). Given that migration is considered as a reengineering process and that the horseshoe model is a generally accepted conceptual model for reengineering, we propose an extended form of the horseshoe model as a holistic model of the migration approach.

Figure 2.1 illustrates our proposed SOA migration framework (so called SOA-MF) for the migration approach. Here, a migration approach follows a horseshoe model by first recovering the lost abstractions and eliciting the legacy elements that are suitable for migration to SOA (left side), altering and reshaping the legacy abstractions to service based abstractions (transformations in the middle area), and finally, renovating the target system based on transformed abstractions as well as new requirements (right side). To adequately illustrate the notion of legacy migration, we should recognize the corresponding key characterizing
2.3. THE SOA MIGRATION FRAMEWORK

corcepts. Migration approaches generally consist of three processes, reverse engineering, transformation and forward engineering (thick arrows in Figure 2.1). We argue that, the migration approach is a transformation of the representations or artifacts (parallelograms in Figure 2.1), that are carried out by means of a certain activity (rounded rectangles in Figure 2.1). Activities can be supported by different types of knowledge as a resource or as the span of information that they should handle. Finally, the process of moving and mapping among the artifacts within the overall migration task, which graphically resembles a horseshoe, can be performed at different levels of abstraction ranging from “code-level” to “concept-level”.

2.3.1 Conceptual Elements of SOA-MF

In the following, we provide a systematic presentation of the main building blocks of the migration framework, so called conceptual elements. Figure 2.2 shows these conceptual elements and the relationships among them. Below, each conceptual element is presented along with its role in the migration approach.

I. Process  According to [Fayad 1997], software processes define what needs to be done in a software development effort and how it is done. Similarly, the migration approach could be defined as the set of tasks carried out during migration. As mentioned, the migration approach is divided into three processes including reverse engineering, transformation and forward engineering. These processes are carried out through the four levels of abstraction (discussed in § 2.3.2).

Reverse engineering is the process of analyzing the existing system to identify the system’s structure, functionality and behavior and to create representation of the system in another form or at a higher level of abstraction (Chikofsky and Cross II 1990). Reverse engineering is the matter of examination and not a process of change. In this process “meaningful higher level abstractions” of the existing system are identified based on bodies of knowledge addressing, for instance, the domain, technology and architecture. From the migration point of view, the main goal of the reverse engineering process is to reach an understanding of the legacy asset to the extent to identify the best candidates among the existing legacy elements for migration to SOA. Here, legacy elements are inherently the “meaningful higher level abstraction”, recovered by means of reverse engineering techniques. In other words, the output of reverse engineering process is a number of legacy assets (in different levels of abstraction) extracted by means of the reverse engineering techniques and are suitable for the migration purpose. From the life-cycle coverage perspective, this process could start from existing implementation and continue with extracting the design entities, recovering the architecture and recapturing abstractions in requirements or business models.
CHAPTER 2. UNDERSTANDING SOA MIGRATION

Figure 2.1: Overview of SOA Migration Framework
2.3. THE SOA MIGRATION FRAMEWORK

Transformation is the process of restructuring one representation form to another at the same level of abstraction (Chikofsky and Cross II, 1990). This transformation could be in the form of reshaping design elements, restructuring the architecture and/or altering business models and business strategies. It should be noted that each of these transformations belongs to a specific level of abstraction. In other words, based on the level of abstraction of the initial and the target representations, different types of transformations are carried out. From the SOA migration perspective, transformation process embraces migration of legacy assets to service based assets and is performed through a set of activities associated to different levels of abstraction. Accordingly, transformation in a particular migration approach could be performed within just one level or across number of them. § 2.3.2 provides more description of different types of transformation activities.

During forward engineering, the subject system is renovated on the basis of the new requirements and goals offered by the target service based environment as well as the target artifacts produced during the transformation process. Specifically, by considering the new requirements and goals, and also the business model, service composition and/or services produced during the transformation process the service-based system is implemented in a top down manner.

II. Artifact In their work Conradi et al. (1993) define artifacts as the products of a process. Here, an artifact represents any product or “raw material” (i.e. models, architecture, piece of code) extracted, transformed or developed during each of reverse engineering, transformation or forward engineering processes.

III. Activity An activity is an atomic or composite production step of a process which aims to generate or modify a given set of artifacts (Conradi et al. 1993). Activities indicate regarding steps of what must be done during each of reverse engineering, transformation and forward engineering processes.

IV. Knowledge Here, we define knowledge as the whole spectrum of content for the following concepts concerning the migration approach: data, models, procedures, techniques, principles, and context. These concepts are the set of information about software systems and business domain which shape the migration approach (given that they provide inputs to the migration activities). Different approaches are distinguished based on types of knowledge they exploit during the migration approach. In (Kazman et al. 1998) the set of levels of information about software system from the source code level to the architectural level is proposed. We follow the same view and classify the bodies of knowledge exploited within migration based on their associated level of abstraction. More precisely, where knowledge originates from (the level of abstraction), indicates the type
2.3.2 SOA-MF Description

Migration approaches can be built in many different ways, i.e. it is not possible to identify one universal migration approach. However, a basic general migration approach constituting the skeleton of migration approach at its most complete form can be defined. SOA-MF is devised based on the same scheme (skeleton of a complete migration approach), that migration approaches cover a portion (or all) of.

So far, we have discussed our definition of the migration approach and the conceptual elements providing basis to describe the approach. Now we describe SOA-MF addressing the migration of legacy assets to SOA. The framework illustrates the migration approach together with details of the artifacts included, activities carried out and types of knowledge exploited within each of migration processes. The graphical representations of the conceptual elements are depicted in Figure 2.1. The processes, activities, artifacts and knowledge elements are depicted by thick arrows, rounded rectangles and parallelograms respectively. In this section, the role of each activity in the whole migration approach and the associated input and output artifacts and the knowledge exploited by the activity of knowledge. As a consequence, the corresponding categories of knowledge are as follows: code-related knowledge, design element-related knowledge, composition knowledge and business domain knowledge. For example, code grammars and models are categorized as code-related knowledge and are used within the reverse engineering process. Cohesion is considered as a design element-related knowledge since it addresses a principle about a single element (i.e. module, component or service). Architectural patterns and styles are of type composition knowledge and, finally, business rules, risks, benefits and plans are categorized as a business domain type of knowledge.
are described. The conceptual elements comprising the framework are in italic.

I. Reverse Engineering In its most basic form, reverse engineering starts from analyzing the legacy code within the code analysis activity. This activity aims to extract the legacy elements identified as candidates for transformation to services. Code analysis techniques such as graph-based analysis, lexical analysis, code querying, etc., are considered instances of the code analysis activity. With regard to this activity, the input artifact is the legacy code while the output consists of set of legacy elements (which could be in the form of components, modules, segments of code, etc.). The extraction of legacy elements from code is influenced by involvement of code related knowledge (such as code grammar and model) as well as bodies of knowledge addressing higher level concepts (such as business domain knowledge).

So far, within the reverse engineering process, the extracted legacy elements are inherently design entities recaptured by means of reverse engineering techniques. However, we could go one step further and recapture the meaningful compositions of these legacy elements within the architectural recovery activity. Here, the composition knowledge such as architectural patterns and architectural styles are involved in identification of the architectural elements and their associated relationships.

Finally, the legacy business model is extracted during the business model recovery activity. The inputs to this activity are the legacy architecture and the existing business domain knowledge such as business rules, business processes, etc.

II. Transformation As mentioned, transformation encompasses process of restructuring one representation form to another at the same level of abstraction. The activities of design element transformation, composition transformation and business model transformation, respectively, realize the tasks of reshaping design elements, restructuring the architecture and altering business models and business strategies. The bridge part of the SOA-MF represents these transformations.

Design element transformation activity is typically performed to move the encapsulation of the legacy elements (extracted during the reverse engineering process) to services. Most of the wrapping techniques fall in this category of transformations. The input artifact to this activity is the legacy design element (i.e. module, component or segment of a code) and the output artifact is basically a service. Types of knowledge which are inputs to design element transformation are: code related and design element knowledge.

Composition transformation activity embodies transformation of the legacy architecture (input artifact) to service compositions (output artifact) in terms of changing the allocation of functionality, their topology, etc. In other words,
components and connectors are transformed to a service composition embracing services and relationships among them. Pattern based architectural transformation techniques fall in this category of transformations. Commonly, this activity exploits composition knowledge and design element knowledge as inputs to perform the transformation. For instance, architectural patterns, service composition patterns and service inventory patterns (i.e. composition knowledge) are used within the composition transformation activity.

During business model transformation activity the existing business model is transformed to a to-be business model based on new requirements as well as opportunities offered by service-based systems. Here, existing business rules, business processes and strategies which are partially embedded in the legacy business model are transformed to new ones to form the basis for development of service-based system. The input artifact to this activity is legacy business model, whereas the to-be business model forms the output. The business model transformation activity is assisted by the business domain knowledge such as business rules, risks, benefits and plans.

III. Forward Engineering

In its most complete form, the forward engineering process starts from the to-be business model. During service analysis, based on the to-be business model a set of candidate service compositions which conceptualize the business processes are identified. Afterwards, the candidate service compositions are consolidated with service compositions identified during composition transformation activity. This activity is succeeded by service design during which the renovated services are designed based on the consolidated candidate service compositions. Similar to service compositions, candidate services are merged with the services identified during design element transformation activity (of transformation process). Finally, during service implementation the service design is transformed to code.

2.3.3 Tiers

Transformation process embraces reshaping of an artifact of the existing legacy assets to another artifact in the service-based environment. Considering four levels of abstraction including code, basic design elements, composite design element and concept, we argue that usually the as-is artifact (in reverse engineering process), the to-be artifact (in the forward engineering process) and their associated transformation activity all reside in the same level of abstraction. In that case, if we consider the as-is and to-be artifacts as well as the transformation among them as a tier, we could characterize and classify a migration approach based on the tiers supported. Figure [2.1] depicts the tiers distinguished from each other by solid lines. As shown in this figure, SOA-MF distinguishes four different
2.4. ON THE ROLE OF SOA-MF

tiers of code, basic design elements, composite design element and concept tiers. It should be noted that, in a sample migration approach, different set of tiers could exist, which are not necessarily adjacent. Consider a migration approach which covers the concept and design element tiers. This implies that the business model transformation and design element transformation activities are included in transformation process whereas no composition transformation is carried out. From another point of view, a transformation in higher level of abstraction may not entail the transformation in lower levels.

2.4 On the Role of SOA-MF

As mentioned previously, the main goal of the SOA-MF is understanding through classification and comparison of existing SOA migration approaches. We argue that, SOA-MF is an intuitive graphical representation, which provides pieces of information to illustrate and characterize each existing migration approach. More precisely, each migration approach could be described based on the processes it supports, artifacts included, activities carried out, types of knowledge exploited and finally tiers they reside in. In other words, if we consider the SOA-MF model as a diagram, each migration approach constitutes a portion of this diagram including the covered conceptual elements. Existing migration approaches use different terms and expressions for inherently similar tasks and concepts regarding migration. A general understanding of the migration approaches could be reached by mapping and positioning the approaches and their associated tasks and artifacts into a common framework. In the same vein, the SOA-MF model facilitates understanding and classification of existing migration approaches and possibly provides the basis for analyzing their limitations and pitfalls. This is realized by identifying each approach’s associated portion of the diagram and comparing them based on their supported conceptual elements and their position related to the SOA-MF model (main diagram). To identify and map the framework elements in a possible migration process, Table 2.2 provides a number of questions that facilitates this mapping.

We have studied a number of SOA migration approaches, and mapped and positioned them on SOA-MF. Two of these approaches, introduced in § 2.2, are discussed here to clarify how a SOA migration approach can be mapped on SOA-MF. This implies that the processes, activities, artifacts and knowledge elements residing in each approach are mapped to an associated conceptual element in the SOA-MF.
Table 2.2: Questions for Mapping a Migration Approaches on SOA-MF

<table>
<thead>
<tr>
<th>Process</th>
<th>Questions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reverse Engineer</td>
<td>Does the migration approach include activities related to extraction of abstractions of legacy assets?</td>
</tr>
<tr>
<td></td>
<td>Are the legacy elements extracted from the legacy code?</td>
</tr>
<tr>
<td></td>
<td>Is any kind of architectural recovery techniques included in the approach?</td>
</tr>
<tr>
<td></td>
<td>Are business models (e.g. business processes and business rules) extracted from the legacy systems?</td>
</tr>
<tr>
<td>Transformation</td>
<td>Does the method include techniques for transformation of legacy elements to services?</td>
</tr>
<tr>
<td></td>
<td>What legacy elements are transformed to service?</td>
</tr>
<tr>
<td></td>
<td>Are the composition of legacy elements transformed to service compositions?</td>
</tr>
<tr>
<td></td>
<td>Are the existing business models (e.g. business processes, business rules or business architecture) altered in order to meet the new requirements and goals?</td>
</tr>
<tr>
<td>Forward Engineer</td>
<td>Does the method include activities related to service development?</td>
</tr>
<tr>
<td></td>
<td>At what stage does the forward engineering process start?</td>
</tr>
</tbody>
</table>

2.4.1 Mapping of SMART Approach on SOA-MF

The bodies of information targeting the existing legacy assets are extracted within the “describe candidate services” and “describe existing capabilities” activities. We classify this set of information (i.e. the stakeholders, business processes, goals, type of platform, data on the existing components, code, etc.) as existing business model. The transformation process embraces development of the migration strategy for transforming the existing capabilities to target SOA environment. The migration strategy is developed by “gap analysis” and “developed migration strategy” activities and are both categorized as business domain transformation activity. The output of the transformation activity, categorized as to-be business model, encompasses standards and guidelines for the service implementation, information on target environment, interaction of services on SOA environment, QoS expectations, etc. In addition, the business goals including costs, efforts and risks of migration that are categorized as business domain knowledge derive the development of migration strategy. It should be noted that, although this method also considers the information regarding to existing legacy code, components and architecture, we assign it to the concept tier since these bodies of information are elicited using high level analysis techniques such as interviews with stakeholders. Besides, the output of this method is an enterprise level plan for migration which is also dedicated to the concept tier.

2.4.2 Mapping of Sneed’s Approach on SOA-MF

The first activity in Sneed’s approach is identification of business rules that are considered as candidates for migration to web services. This is considered as codi-
Figure 2.3: Mapping of SMART on SOA-MF

fying the knowledge regarding existing business processes, categorized as business domain knowledge. Reverse engineering process starts from extraction of segments of code realizing business rules with good reuse potential. This is realized by the code stripping techniques that we categorize as a code analysis activity. The inputs of this activity are business rules (i.e. business domain knowledge) and the existing legacy code. Besides, the code grammars and code models that we categorize as code level knowledge facilitate the extraction of business rules from code. The transformation process encompasses transformation of extracted modules to web services by means of wrapping techniques described in §2.2.2. Since the transformation encompasses wrapping a basic design element such as modules and altering them to web services, we categorize it as basic design element transformation activity. The forward engineering process enables the implementation of the business processes, which is categorized as service implementation activity. Accordingly, this method is dedicated to basic design element tier.

2.4.3 What the Mappings Imply

The mappings of the two example approaches already reflect the commonalities and differences among them. In this section, we provide the answer to the question of what do these two mappings imply and how they facilitate the un-
Figure 2.3 reflects the following features of SMART: from lifecycle-coverage perspective, the reverse engineering and the forward engineering processes are not covered. As a result, recapturing the abstractions of existing legacy assets are not addressed. In addition, development of the service based system is not covered. Transformation is carried out at concept level and embraces altering legacy business model of the existing system to the to-be business model using business domain level knowledge.

The following features of Sneed’s migration approach can be extracted from the associated mapping on SOA-MF (see Figure 2.4). The horseshoe like representation of this approach on SOA-MF illustrates that all three processes of reverse engineering, transformation and forward engineering are carried out. The transformation occurs at basic design element level and migration is limited to altering modules to services, however, business domain knowledge facilitates the migration.

To sum up, SMART provides high level solutions for migration problem, while ignoring the technical details of legacy element extraction and service development. Whereas, Sneed’s approach presents migration at lower level through focusing on technical details.

Figure 2.4: Mapping of Sneed’s approach on SOA-MF
2.5 Conclusion

It is hard to understand and classify existing approaches in an emerging and still fuzzy research field like SOA migration. In the past few years, several SOA migration approaches have been introduced, each focusing on a specific perspective on SOA migration, and using own concepts (activities, artifacts, etc.) to represent migration.

This chapter has presented a SOA migration framework (SOA-MF). SOA-MF facilitates to characterize and isolate the properties of migration approaches in terms of processes it supports, artifacts included, activities carried out, and types of knowledge exploited. A unified representation of different approaches is achieved by mapping them on SOA-MF, which also provides the basis for their comparison and analysis.

With regards to general applicability of SOA-MF, and assuring factor is that this framework stems from existing theory on reengineering and architectural recovery while it is is constantly refined through the thesis. Moreover, this framework is used to characterize 61 academic approaches (see Chapter 3) as well as 9 industrial approaches (see Chapter 4). This further consolidates its general applicability. In addition, the mapping of industrial approaches on SOA-MF were iterated and checked by interviewees (see Chapter 4 for details), which further assures its expressiveness.

During experimentation with earlier versions of SOA-MF, we observed that the notion of tier plays an important role in positioning and classifying the various migration approaches. The tiers of SOA-MF covered by a specific SOA migration approach can explain the following aspects: the associated level of abstraction in which the transformation occurs and the transformations that entail lower level ones. A relevant classification of existing approaches can be achieved by considering tiers they cover in SOA-MF. For instance, SMART is dedicated to the concept tier category (in which just the concept tier is covered); while Sneed’s method is regarded as a basic design element tier approach; and the approach presented in Winter and Ziemann (2007) is dedicated to the category of migration approaches covering all tiers. In the next chapters, we will see how SOA-MF helped us to understand both industrial and academic approaches.
A Systematic Review of SOA Migration Approaches

Given the diversity in the SOA migration approaches, it is hard for an enterprise to select an approach that fits best its specific needs. The objective of this chapter is to bring order in the existing SOA migration approaches and facilitate determining one’s own migration approach. A systematic literature review was conducted to extract the SOA migration categories. By categorizing the migration approaches, we provided a frame of reference for SOA migration. This frame of reference typifies the migration approaches with respect to two views including activity view and knowledge view. Each of the two views reflects a different aspect of SOA migration approach, namely, the migration activities and the available knowledge assets.

3.1 Introduction

Given many different migration approaches, a common question an enterprise faces is how to utilize the existing approaches in defining the enterprise’s migration approach. Let us consider the example of two enterprises one aiming at modernizing its legacy code through automatic code transformation, and the other one needing to reengineer the pre-existing business processes by interacting with stakeholders. Considering the significant difference in the two enterprise’s migration goals, it is apparent that the migration approaches, determining the path to be taken to achieve those goals, has to be different as well. To define its approach, the first enterprise needs to exploit a migration approach that supports code understanding and code transformation. Migration approaches supporting business process reengineering using interviews or workshops, however, fit better the needs of the second enterprise. Currently, enterprises are spending a significant amount of time and effort on selecting and devising approaches for their
migration purpose in an ad-hoc manner. Consequently, this mainly results in increase in costs. To resolve this shortcoming, a systematic definition of migration approaches is required.

To devise a migration approach, one needs to consider various aspects such as what activities are needed for SOA migration, and what are the available knowledge assets. Accordingly, to select a migration approach, it is essential to know how that specific migration approach is characterized considering those aspects. Without such a characterization devising the migration approach is more complicated. Summing up, a reference that frames different approaches facilitates systematically determining the migration path to take. This chapter presents such frame of reference.

To define the frame of reference, we conducted a systematic review that extracts migration categories existing in the field. We chose systematic review as our research method in aggregating existing SOA migration approaches for two main reasons: (i) it encourages methodologically rigorous results in identifying and selecting the relevant studies on SOA migration topic, (ii) the strength of systematic reviews in minimizing the bias in the review process enhances the extraction of sound and meaningful categorization of migration approaches. By devising a coding procedure, we analyzed the studies and categorized the approaches with respect to two views including activity, and knowledge. Each of the two views reflects a different aspect of SOA migration approach, namely, the migration activities and the available knowledge assets. Thus, the categorizations of migration approaches based on the two views act as a frame of reference for SOA migration which brings order and enhances understanding in how such migration can be carried out and what should drive the migration. Accordingly, this frame of reference increases awareness of the ways in which legacy assets can be migrated to SOA.

3.2 Two-View SOA Migration Approach

This work analyzes the academic migration approaches by representing them in a unified manner. Using SOA-MF, introduced in Chapter 2, one can represent the migration approaches based on the activities covered and knowledge elements used or produced. In this work we take a step further and separate the two views of knowledge and activity. In this section we explain the vision behind the view-based representation of the migration approaches. In the remainder of this chapter we use such a representation to categorize the migration approaches and create the SOA migration frame of reference.

An approach for SOA migration states a path from the As-Is state (i.e. As-Is legacy assets) to the To-Be state (i.e. To-Be services). To define such an approach, two questions have to be answered: (i) what knowledge is available
about the As-Is and To-Be states, and (ii) what activities need to be done to carry out the migration. We argue that each of these questions reflects a distinct view on the approach.

The notion of views has been introduced in the software architecture field since early 1990s (Kruchten 2009). An architectural view, there, is a partial representation of a system, from the perspective of a stakeholder’s concern. Following the same perspective, we define a view of the SOA migration approach as a partial representation of the migration approach, from the perspective of a particular concern. We propose the following views for SOA migration approach: (i) Knowledge view highlights the types of knowledge that shape and drive the migration, and (ii) Activity view reflects what needs to be done in SOA migration. We here discuss our rationale behind choosing this Two-View representation.

I. Understanding As-Is and To-Be states. As mentioned in Chapter 2 we consider the problem of migration of legacy assets to SOA as a reengineering problem. According to (Tilley and Smith 1995) any reengineering effort embraces an approach indicating how to move from the As-Is state to the To-Be state. As such, understanding the As-Is and To-Be states is essential for carrying out the migration. To determine the migration approach, the concern of what knowledge elements define the As-Is and To-Be states have to be addressed. This concern is reflected in the knowledge view.

II. Migration activities. To move from the As-Is to the To-Be state one needs to identify the best-fitting set of activities to perform the reengineering (Tilley and Smith 1995). To this end, the decisions regarding the best-fitting activities have to be addressed in the migration approach, that is the focus of activity view.

Figure 3.1 gives an example of the Two-View migration approach. The knowledge view (Figure 3.1.I) indicates the input and output knowledge elements. The available knowledge is shown in white, and the required knowledge that is not available with grey. The knowledge conversions, shown using arrows, indicate what input knowledge is required to create a certain output. The activity view (Figure 3.1.II) represents the activities covered in migration approach. To do so this view maps the migration activities on the SOA Migration Framework (SOA-MF).

3.3 Research Method

We followed a systematic literature review process based on the guidelines proposed in (Kitchenham 2007; Dyba et al. 2007). The protocol specified the method to be followed in terms of the research questions, the search process, and the data to be extracted.
3.3.1 Pilot Study

We developed the protocol by running a pilot study. The goal of the pilot study was to execute the protocol on one of the libraries (IEEE Xplore) and check if the protocol was generally rigorous enough and to improve it where necessary. We initially developed the search string inspired by relevant literature and prior systematic reviews in the field of SOA (Gu and Lago, 2009a). In order to validate the search string, we ran the string on IEEE Xplore library and checked if the search string is able to detect the 25 known relevant studies. The pilot study uncovered the problems with the search string as it overlooked some of the known studies. For example, the term ‘re-use’ was not included in the initial search string and the pilot study served to include this term to include known studies. In addition, the pilot study served to identify and add the alternative spellings of a specific term (re-engineering and reengineering). To make sure that the data analysis method best fits this study we consulted an expert in qualitative data analysis methods. Finally, the resulting protocol was checked by senior researchers experienced in software engineering and systematic reviews. The feedback helped us to improve the protocol. Examples are tuning the inclusion/exclusion criteria and better formalizing the data extraction.
3.3. RESEARCH METHOD

Research Questions

The systematic review sought for extracting the types of SOA migration approaches in academia. To achieve this goal, we defined the following research questions:

What approaches regarding legacy to SOA migration, have been proposed in the research community so far? In particular, the following aspects facilitate characterizing the SOA migration approaches:

- (RQ-I) what are the activities carried out?
- (RQ-II) what is the sequencing of the activities?
- (RQ-III) what are the knowledge elements that are used and produced?

Search Strategy

As the first step of systematic search, three main keywords were built from our research question, namely: **migration**, **legacy assets** and **SOA**. Furthermore, for each of keywords a set of related terms addressing synonyms and alternative spellings were identified. Based on the keywords and their related terms we defined the following search string:

(SOA or ‘service-oriented’ or ‘service-computing’ or ‘service-based’ or ‘service-centric’ or ‘service’ or ‘service-engineering’ or SOSE) AND (‘legacy code’ or ‘legacy system’ or ‘existing system’ or ‘legacy component’ or ‘existing code’ or ‘existing asset’ or ‘existing component’ or ‘pre-existing code’ or ‘pre-existing system’ or ‘pre-existing component’ or ‘legacy software’ or ‘existing software’ or ‘pre-existing software’) AND (migration or modernization or transformation or reengineering or re-engineer or evolving or reuse or ‘service mining’ or ‘service identification’ or ‘service extraction’)

Data sources. We used the following libraries as main resources: IEEE Xplore, ACM Digital Library, ISI Web of Knowledge, SpringerLink, ScienceDirect, and Wiley Inter Science Journal Finder.

Search process. As major venues on service-oriented systems started in 2003 onwards (ICSOC\(^1\) in 2003, SCC\(^2\) in 2004, SOSE in 2005\(^3\)), we decided to set 2000 as the start date to minimize the chance of overlooking relevant studies. This implies that a study is selected as a candidate study if it contains the search terms in the abstract and is published between Jan 1st 2000 and Jan 1st 2011 (i.e. when this review was conducted). We applied the search terms to titles and abstracts considering that they provide a concise summary of the work.

\(^{1}\)International Conference on Service Oriented Computing
\(^{2}\)International Conference on Services Computing
\(^{3}\)International Workshop on Service-Oriented System Engineering
CHAPTER 3. A SYSTEMATIC REVIEW OF SOA MIGRATION APPROACHES

This decision was validated by running the search string on the data sources and checking if the pilot studies are retrieved. Due to lack of standardization between the digital libraries, we had to adapt the search string for each data source. To ensure that search strings are adapted correctly two reviewers independently ran the search process and this way the overlooked article were identified.

Selection of Primary Studies

Articles that satisfied the following inclusion criteria were selected as a primary study.

- I1) A study that proposes approaches addressing migration of legacy assets to services. Rationale: we are interested in studies that are about SOA migration.

- I2) A study that is developed by either of academics and practitioners. Rationale: both academic and industrial migration approaches are relevant to this study.

- I3) A study that is published in software engineering field. Rationale: We seek for approaches specifically addressing migration of pre-existing software to services. For instance, methods for migration from an existing network to a new one is out of scope of this study.

- I4) A study that is peer-reviewed. Rationale: a peer-reviewed paper guarantees a certain level of quality and contains reasonable amount of content.

- E1) A study that is not about migration to services. Rationale: studies which support migration to other types of target systems (not to service-based) should be excluded.

- E2) A study that does not address migration from existing legacy assets. Rationale: studies that do not address migration, i.e. reuse of pre-existing assets needs to be excluded.

- E3) A study which does not specifically proposes a solution for migration. Rationale: studies that do not specifically provide a solution for the migration problem must be excluded. For instance, studies presenting challenges on SOA migration are out of scope of this work.
3.3. RESEARCH METHOD

Included and excluded studies

In total we found 448 publications, whose abstracts contain the keywords defined in the search string. Out of 448 papers only 165 studies that appeared to be completely irrelevant were excluded based on their title. At the second stage, by applying inclusion/exclusion criteria on the abstracts 136 studies were included. We found that abstracts could provide little information and consequently it was not always obvious that the study provides a specific approach for SOA migration. Therefore in this stage we included all studies that focus on SOA migration. Finally, the full-text of studies were reviewed against the inclusion/exclusion criteria and 69 papers were included in our review. Although we identified 69 articles by this search process, some articles were earlier or short versions of other articles. If the same migration approach were included in multiple publications it could bias the results as a single approach is considered as multiple primary studies. To resolve this issue we examined the publications written by same set of authors and affiliations, to see if they were reporting the same migration approach. In total, seven papers were excluded on this basis and the publication that was more complete was included. We further looked for the publications that represent the same approach that none of them is more complete but they are complementary. None of the studies fell under this category. Thus, we ended up with 61 primary studies.

Search result management

The reference details of each study was recorded using JabRef. For each subsequent stage of the selection process, described in the following, a separate JabRef database was established.

Quality Assessment of Primary Studies

We ranked the primary studies considering their level of applicability in practice. This can bring insight into the extent to which the SOA migration approaches proposed in the primary studies are likely to be applicable in practice. We used the three following scales to assess the applicability of the primary studies:
CHAPTER 3. A SYSTEMATIC REVIEW OF SOA MIGRATION APPROACHES

- **High:** the approach is applied in a real-world case study in industrial or organizational settings.

- **Medium:** the proposed approach is explained and discussed using descriptive examples.

- **Low:** the approach is not applied in practice, nor is its applicability in practice exemplified.

### 3.3.2 Data Extraction

We extracted the data related to migration approach followed in each of 61 primary studies to address the research questions. The extracted data was stored in an extraction form (see Table 3.1). The extraction forms helped us to store details of the primary studies as well as their migration approach. Regarding the migration approach, the summary of the approach the activities and their input/output knowledge elements were recorded in the data extraction forms.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Study identifier</td>
<td>[Provide a unique ID]</td>
</tr>
<tr>
<td>2. Title</td>
<td>[Title of the primary study]</td>
</tr>
<tr>
<td>3. Reviewer name</td>
<td>[Name of the reviewer conducting the data extraction]</td>
</tr>
<tr>
<td>4. Data source</td>
<td>[The name of the database where the primary study was found (i.e. IEEE Xplore, ACM Digital Library, ISI Web of Knowledge, SpringerLink, ScienceDirect, and Wiley Inter Science Journal Finder.)]</td>
</tr>
<tr>
<td>5. Type of article</td>
<td>[The type of article(workshop, conference, book chapter or journal)]</td>
</tr>
<tr>
<td>6. Summary of migration approach</td>
<td>[Enter the summary of the migration approach]</td>
</tr>
<tr>
<td>7. Goal of migration</td>
<td>[Enter the goal of performing the migration]</td>
</tr>
<tr>
<td>8. Activities with inputs/outputs</td>
<td>[List the set of activities carried out in migration together with their inputs and outputs]</td>
</tr>
<tr>
<td>9. Sequencing of activities</td>
<td>[Provide the sequence in which each of the activities above are carried out]</td>
</tr>
</tbody>
</table>
| 10. Usage of approach              | [Is the approach applied in a case-study? (Yes/No) If yes: describe whether the case-study is a real-world case study or an example one.]

### 3.3.3 Data Analysis Method

This study seeks for *categorization* of migration approaches based on the two views of activity and knowledge. To this end, we mapped each of the migration approaches on the two views. The question that we faced was how to systematically analyze and map the primary studies, in such a way that the meaningful categorizations are determined. We chose coding as our qualitative analysis
3.3. RESEARCH METHOD

Figure 3.3: Coding Procedure

Table 3.2: Codes Start-List

<table>
<thead>
<tr>
<th>Category</th>
<th>Code</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activity</td>
<td>ACT-CoAn</td>
<td>Code Analysis</td>
<td>The activity that analyzes the legacy code and extracts its comprising elements, called legacy elements (e.g. components, modules, or segments of code).</td>
</tr>
<tr>
<td></td>
<td>ACT-ArchRec</td>
<td>Architectural recovery</td>
<td>The activity that recovers the interdependencies and compositions of legacy elements.</td>
</tr>
<tr>
<td></td>
<td>ACT-BMRec</td>
<td>Business Model Recovery</td>
<td>The activity that extracts the business domain models (e.g. business process models, use case models, business rules) from legacy elements.</td>
</tr>
<tr>
<td></td>
<td>ACT-DETr</td>
<td>Design Element Transformation</td>
<td>The activity that transforms the legacy elements to services.</td>
</tr>
<tr>
<td></td>
<td>ACT-ComTr</td>
<td>Composition Transformation</td>
<td>The activity that transforms the legacy architecture to service compositions.</td>
</tr>
<tr>
<td></td>
<td>ACT-MPTr</td>
<td>Business Model Transformation</td>
<td>The activity that transforms the legacy business models to new ones to be used in the service-based system.</td>
</tr>
<tr>
<td></td>
<td>ACT-SrvAn</td>
<td>Service Analysis</td>
<td>The activity that identifies candidate services from business domain models.</td>
</tr>
<tr>
<td></td>
<td>ACT-SrvDsg</td>
<td>Service Design</td>
<td>The activity that focuses on design of software services and service compositions.</td>
</tr>
<tr>
<td></td>
<td>ACT-SrvImpl</td>
<td>Service Implementation</td>
<td>The activity that realizes the implementation of software services.</td>
</tr>
<tr>
<td>Knowledge</td>
<td>KN-Sol</td>
<td>Solution-related knowledge</td>
<td>The knowledge that addresses the solution to a specific problem domain.</td>
</tr>
<tr>
<td></td>
<td>KN-Prob</td>
<td>Problem-related knowledge</td>
<td>The knowledge presenting the analysis of the problem, the problem decomposition, and the world in which it is located (problem space).</td>
</tr>
<tr>
<td></td>
<td>KN-Tacit</td>
<td>Tacit Knowledge</td>
<td>The knowledge that remain implicit in stakeholder mind.</td>
</tr>
<tr>
<td></td>
<td>KN-Explicit</td>
<td>Explicit Knowledge</td>
<td>The knowledge that is codified in documents and artifacts.</td>
</tr>
</tbody>
</table>
CHAPTER 3. A SYSTEMATIC REVIEW OF SOA MIGRATION APPROACHES

method. To carry out the mappings systematically, we devised the coding procedure represented in Figure 3.3.

**Codes.** For creating the codes we followed the suggestion of Miles and Huberman (1994), namely, to have an initial set of codes, called “start-list”, that is refined during the analysis. Our start-list stems from the SOA migration framework (SOA-MF), proposed in 2. For each of the comprising activities of SOA-MF, shown in rounded rectangles in Figure 3.1 a code was created (e.g. for the activity Code Analysis the code ACT-CoAn was created). Based on the categorization of knowledge presented in Section 3.4.4 we created a code for each of the categories namely, problem-related, solution-related, tacit, and explicit. Table 3.2 represents the start-list of codes along with their descriptions. The start-list, of course, was refined during the analysis as the new insights occurred during the analysis.

**Coding procedure.** Inspired by the procedure proposed by Lincoln and Guba (1985) we devised a coding procedure to systematically codify the primary studies and refine the codes when needed. By codifying the primary studies using the start-list in a step-by-step manner, the coding procedure enables identifying the three-view representation of each study. Furthermore, each step of the procedure guides the relevant refinements to the code as described in the following.

- **Step 1: filling-in activities (RQ-I).** This step codifies the activities involved in the migration. Filling in refers to creating new codes as insights occur during the analysis. Accordingly, if an activity did not fit any of the existing codes, a relevant code was added. For instance, initially a code representing the “code translation” activity did not exist, since it was originated from the existing theory on architectural recovery (Kazman et al., 1998) that did not consider transformation at code level. During the early stages of this analysis, we observed that there are some migration approaches that translate the legacy code to web services. In order to characterize such approaches a code called “code translation activity” was added. Eventually, the ‘code translation activity’ was added to the transformation process of SOA-MF as well (see Figure 3.5.1).

- **Step 2: filling-in/surfacing knowledge elements (RQ-III).** The second step, codifies the input/output knowledge elements of activities. This way, the knowledge elements that are used and/or produced throughout the migration approach are extracted. Surfacing, as the name implies, accounted for the emergence of new categories of knowledge elements.

- **Step 3: surfacing knowledge conversions (RQ-III).** Once the knowledge elements used in each study are identified, this step identifies the interactions

---

4Whilst Table 2.2 provides guidelines for mapping migration approaches on SOA-MF, it does not ensure systematic mapping of the approaches. To support such systematic mappings in this work we devised the codes and the coding procedure.
3.3. RESEARCH METHOD

among those knowledge elements. By further surfacing the knowledge conversions the new categories are identified.

- **Step 4: surfacing sequencing of activities (RQ-II).** The forth step identifies the sequence of activities as well as the categories related to types of sequencing.

- **Step 5: bridging patterns.** The last step discovers the patterns among the categories in each of the activity view, and the knowledge view.

**Conducting coding procedure.** Once the codes and coding procedure were decided, the analysis could be undertaken. We went through the primary studies and applied the steps. By following the first four step, we were able to identify the Two-View migration approach of each of primary studies. Once all the primary studies were codified, by undertaking Step 5 general finding were obtained. Below we describe in detail how each step was undertaken.

By following Step 1 of our coding procedure, we labeled each activity using the relevant codes. As it is evident from the description of the activities in Table 3.2, each activity is considered as a transformation of a specific input to a specific output. For instance, an activity (or series of activities) that transforms a code to a *legacy element* should be codified as *code analysis*. By codifying the activities of all primary studies we identified the coverage of activities of each migration approach on SOA-MF. By doing this we were able to identify the activity view of each primary study.

Following Step 2 we coded each identified knowledge element and then mapped to two levels of problem and solution. By comparing and studying the coded element and consequently identifying new categories we further classified the knowledge elements (i.e. Surfacing). For instance, by comparing and categorizing different types of knowledge describing the system implementation, we identified four main types as illustrated in Figure 3.9. These new categories are described in §3.4.4. Furthermore, based on where a knowledge resides (documents or people’s mind) we coded the knowledge elements into one of the categories of *explicit* or *tacit* respectively.

In Step 3 we identified how different knowledge elements interact. More precisely, we studied what knowledge elements are used to produce new knowledge. This way we were able to identify the knowledge conversions. By comparing the knowledge conversions we identified different categories of conversions. These categories are described in §3.4.4.

In Step 4 we captured the sequencing of activities by extracting the ordering of knowledge conversions identified in the previous step. As an example, the sequencing of activities shown in Figure 3.5 (F4.b) was recognized by identifying
the following knowledge conversions: Code → Design Model → Service Model → Service Implementation. In short, we identified two categories of sequencing, which we describe in §3.4.3.

In Step 5 we went through the activity and knowledge views of all primary studies and searched for meaningful patterns in types of migration approaches. As a result we identified a number of interesting observations and lessons learned. This is the topic of §3.6.2.

3.4 Results

We identified 61 primary studies on SOA migration; see Appendix A. In what follows, we discuss the primary studies. By analyzing the primary studies we answered our three research questions. We first present an overview of the primary studies. Then, we answer each of the research questions.

3.4.1 Overview of the Included Studies

The overview of primary studies is given in Appendix A. One of the first interesting observations from the primary studies is the upward trend in the number of publications that discuss SOA migration approaches. Figure 3.4 shows the number of publications per year as well as the trend-line. The positive trend-line slope indicates an increasing interest in the topic. The first paper on SOA Migration was published in 2000. However, until the early 2003 the topic received fairly little attention. Between 2003 and 2008, there has been an increase in the number of publications. The number of studies drops slightly in 2009 and 2010. Table 3.3 gives a breakdown of where the primary studies are published.

As noted, we assessed the quality of primary studies based on the extent to which the SOA migration approaches proposed in the primary studies are likely
to be applicable in practice. In this regard, most of the primary studies, have a certain form of evaluation of their applicability. In total, 13 out of 61 primary studies did not provide any explanation on applicability of the approaches. 14 of the primary studies described the description of the approach with example case studies. 34 of the primary studies were applied in real-world case studies and experiments.

Table 3.3: Distribution of primary studies by resources

<table>
<thead>
<tr>
<th>Digital library</th>
<th>Number</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEEE Xplore</td>
<td>36</td>
<td>60%</td>
</tr>
<tr>
<td>ACM Digital Library</td>
<td>4</td>
<td>6%</td>
</tr>
<tr>
<td>SpringerLink</td>
<td>15</td>
<td>24%</td>
</tr>
<tr>
<td>ScienceDirect</td>
<td>4</td>
<td>6%</td>
</tr>
<tr>
<td>Wiley Inter Science Journal Finder</td>
<td>1</td>
<td>2%</td>
</tr>
<tr>
<td>ISI Web of Science</td>
<td>4</td>
<td>6%</td>
</tr>
</tbody>
</table>

3.4.2 RQ-I What activities are carried out in SOA migration?

As noted, by following Step 1 of our coding procedure, we identified the activity view of each primary study. Figure 3.5.III illustrates the schematic forms of the activity views. As an example, F4.b is a schematic form of the activity view shown in Figure 3.5.II. By thoroughly analyzing the activity views, we identified a set of meaningful relationships among the approaches with graphically similar views and their migration objectives and solutions. For instance, the migration approaches that have an activity view similar to the one schematically shown in F1.a in Figure 3.5.III, provide a common solution for migration (i.e. translating or wrapping the legacy system as-a-whole to a service). More precisely, thanks to SOA-MF, the migration approaches that include conceptually similar activities, have graphically similar activity views as well. By considering similar SOA-MF coverage patterns, we clustered different migration approaches. This way, eight distinct families of SOA migration approaches were extracted. Figure 3.5.III illustrates the schematic form of distinguished activity views that are dedicated to each family. As an example, F4.b is a schematic form of the activity view shown in Figure 3.5.II.

In the reminder of this section we describe each family in the following way: 1) the family at a glance provides a general description of ‘what the activity view of each family implies’ 2) the observations include explanation of the observations related to ‘what migration entails’ in each family.
### Table 3.4: Overview of SOA migration families

<table>
<thead>
<tr>
<th>Family</th>
<th>Members</th>
<th>Number</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>Bodhuin and Tortorella (2003); Liu et al. (2004); Vargas et al. (2004);</td>
<td>10</td>
<td>17%</td>
</tr>
<tr>
<td></td>
<td>Lee et al. (2004); Huang (2009); Iocola (2007); Balis et al. (2008);</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cheng-hao et al. (2010); Koschel et al. (2009)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F2</td>
<td>O’Brien et al. (2005); Chen et al. (2009); Sindhgatta and Ponnaigudi</td>
<td>11</td>
<td>20%</td>
</tr>
<tr>
<td></td>
<td>(2008); Li and Tahvildari (2008); Ilk et al. (2008); Alahmar et al.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(2010); Bao et al. (2010); Chung et al. (2009); Nakamura et al. (2009);</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Zhang et al. (2010b); Zhou et al. (2010)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F3</td>
<td>Lavery et al. (2004); Lewis and Smith (2007); Hutchinsonson et al.</td>
<td>5</td>
<td>8%</td>
</tr>
<tr>
<td></td>
<td>(2007); Cinar and Zordan (2009); Chung et al. (2007)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F4</td>
<td>Ahn et al. (2008); del Castillo et al. (2009); El-Ramly et al.</td>
<td>15</td>
<td>25%</td>
</tr>
<tr>
<td></td>
<td>(2009); Gonzalez et al. (2009); Guo et al. (2005); Haidar and Abdallah</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(2008); Huang et al. (2008); Li et al. (2007); Sneed (2006); Zou and</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Kontogianis (2008); Lucas et al. (2008); Matos (2008); Bissyandie et</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>al. (2010); Djeloul et al. (2009)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F5</td>
<td>Cetin et al. (2007a); Chen et al. (2005); Koufi et al. (2010);</td>
<td>5</td>
<td>8%</td>
</tr>
<tr>
<td></td>
<td>Muthiah et al. (2010); Nasir et al. (2010)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F6</td>
<td>Canfora et al. (2008); Cuadrado et al. (2008); Heckel et al. (2008);</td>
<td>9</td>
<td>15%</td>
</tr>
<tr>
<td></td>
<td>Liu et al. (2008); Zhang et al. (2006); Marchetto and Ricca (2009);</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Millham (2010); Tibermac and Kerdoudi (2010); Wang et al. (2010)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F7</td>
<td>Arcelli et al. (2008); Kannan and Srivastava (2008); Pahl and Barrett</td>
<td>3</td>
<td>5%</td>
</tr>
<tr>
<td></td>
<td>(2006)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>F8</td>
<td>Nguyen et al. (2009)</td>
<td>1</td>
<td>2%</td>
</tr>
</tbody>
</table>
3.4. RESULTS

Figure 3.5: SOA Migration Families

**Code Translation Family (F1)**  
**Family at a glance:** The activity view representing this family (simplified in Figure 3.5 F1(a)), reflects the following feature: out of the three processes, the migration approach is limited to transformation at system level. Accordingly, the existing legacy code is transformed to service-based implementation.

**Observations:** In this family, migration entails moving the legacy system as-a-whole to a service-oriented platform or technology, without decomposing the existing system. We identified two main categories in this family: (1) translating the whole code to web services (e.g. Varga et al. (2004)), and (2) wrapping the whole application as a web service (Chenghao et al., 2010). The problem addressed by the first category is to translate a legacy code to a web service implementation. The second category embraces encapsulating the interfaces of the existing application to a (web) service interface.
CHAPTER 3. A SYSTEMATIC REVIEW OF SOA MIGRATION APPROACHES

Service Identification Family (F2)  
**Family at a glance:** In this family, the transformation process is not covered, meaning that reshaping of the legacy elements to service-based elements is not realized. The reverse engineering process is carried out in all family members, while forward engineering occurs only in some (Figure 3.5.F2(c,d)).  
**Observations:** In this family migration is limited to the identification of the candidate services in the existing legacy assets. This family uses reengineering activities to identify the elements that are candidates for migration to SOA.

Business Model Transformation Family (F3)  
**Family at a glance:** In this family, the reverse engineering and the forward engineering processes are not covered. Migration is realized by the transformation process, carried out at concept level.  
**Observations:** Based on the types of transformation at concept level, we found two main categories in this family: (1) approaches providing a meta-process for migration (e.g. [Lewis and Smith](2007); [Hutchinson et al.](2007); [Umar and Zordan](2009)) and (2) approaches with business process reengineering (e.g., [Lavery](2004)). The main goal of the meta-process category is to support the decision regarding ‘how to perform migration’. The constituent activities of these approaches support decision making on the migration approach itself. Due to the orthogonal view of this category on the migration approach, we recognize this category as a ‘meta-process’. Despite having the same coverage pattern on SOA-MF as the first category, we found that the business process reengineering category of this family reflects a different perspective on SOA migration: altering the business process of the existing legacy assets to serve as a basis for top-down service development.

Design Element Transformation Family (F4)  
In this family, the transformation process only occurs at ‘basic design element’ level (e.g. modules or components). Similarly, reverse and forward engineering processes, if covered, are also limited to this level.  
**Observations:** Migration in this family is limited to reshaping the existing legacy elements to the service-based elements. More precisely, a set of legacy elements, extracted by means of the ‘code analysis’ activity or simply known beforehand, are transferred to a set of services or service-based elements. For instance, a ‘component specification’ is altered to ‘service specification’ (e.g. [Li et al.](2007)), or a ‘module’ is reshaped to a ‘service’ (e.g. [Sneed](2006)) or ‘segment of code in the persistence layer’ is transformed to a ‘data service’ (e.g. [del Castillo et al.](2009)).
3.4. RESULTS

Forward Engineering Family (F5)  
**Family at a glance:** This family fully covers the forward engineering process, whereas transformation and reverse engineering processes occur at ‘basic design element’ level.  
**Observations:** The main focus of this family is on development of service-based systems starting from the desired business processes. This family uses the reverse engineering only to locate the functionality of services identified in forward engineering process. *As such the migration entails Top-Down service-based development while locating the realization of the required business functionalities and transforming them to services.*

Design and Composite Element Transformation Family (F6)  
**Family at a glance:** The three migration processes occur in the two levels of the ‘basic design element’ and ‘composite design element’, meaning that the members include both design element and composition transformations.  
**Observations:** What characterizes this family is having transformation at the both level of ‘basic design element’ and ‘composite design element’. This entails altering legacy elements to services (i.e. design element transformation) as well as reshaping the structure and the topology of legacy elements to realize new service compositions (i.e. composition transformation). *Migration, here, embraces recovering and refactoring of the legacy architecture to the service-oriented architecture as well as reshaping the legacy elements to service-based elements.*

Pattern-based Composition Transformation Family (F7)  
**Family at a glance:** Migration only includes the transformation process at ‘composite design element’ level. This implies that the architecture of the existing system is altered or configured into the service based architecture.  
**Observations:** A common feature in this family is using ‘patterns’ for transforming the existing architecture to service-based architecture. Patterns are inherently reusable solutions, that are here used to extract services or facilitate transformations of legacy elements to services. *Migration here entails pattern-based architectural transformation to SOA.*

Forward Engineering with Gap Analysis Family (F8)  
**Family at a glance:** The transformation process, in this family, occurs in the three levels of ‘concept’, ‘composite design element’ and ‘basic design element’. As shown by (Figure 3.5F8(a)), the forward engineering process covers the activities of ‘service analysis’ and ‘service design’ whereas the reverse engineering process is not covered.  
**Observations:** This family mainly focuses on top-down service development, starting from extraction of the business model of the target system and further designing service compositions and services. What distinguishes this family from
pure top-down service development approaches is that at each abstraction level (including concept, composition and design level) a comparison (a gap analysis) among the new and the pre-existing artifacts occurs. This comparison serves to assess how the desired business services can be realized by exploiting pre-existing capabilities. The migration here entails Top-Down service development while assessing the reuse opportunities at all abstraction levels.

3.4.3 RQ-II What is the sequencing of the activities?

The previous section described what activities are covered in the primary studies. Here we focus on the order in which the covered activities are carried out. Following the Step 4 of the coding procedure, we identified the sequencing of the activities of each primary study. In doing this we observed dissimilar sequencings in the migration approaches that are comprised of similar activities. Figure 3.6 and Figure 3.7 illustrate some examples of different sequencings on the schematics shapes. Considering the graphical representation of the sequencings we observed two main categories in the primary studies: arc-shaped (Figure 3.6) and bowl-shaped (Figure 3.7) approaches.

I) Arc-shaped approaches: the sequencing of activities in this category resembles an arc (see Figure 3.6), starting from reverse engineering process. As shown in Table 3.5, 68% of the primary studies are arc-shaped and they share the following similarity: the As-Is state (characterized by legacy assets) is what drives the migration. Moreover, we found that the arc-shaped approaches aim at renovating the existing legacy system to reconstitute it in the new form of SOA. Consequently, they mainly focus on how to adapt the legacy assets to the SOA environment. To this end, the reverse engineering process realizes understanding of the existing system; transformation process specifies how to restructure the legacy assets, while the forward engineering process realizes the restructuring.

II) Bowl-shaped approaches: The illustration of the knowledge conversions on the mappings has a shape similar to a bowl (see Figure 3.7). Unlike the
3.4. RESULTS

arc-shaped approaches, the bowl-shaped ones start from the forward engineering process. The To-Be state (characterized by requirements or properties of service-based system) is the main driver of the migration. Here, the main goal of migration is to facilitate reuse in building new service-based systems. This goal changes the order in which the three processes are carried out. Accordingly, the forward engineering process realizes the service-based development; to do so, the reverse engineering process facilitates identifying reusable legacy assets and the transformation process reshapes the legacy elements to service-based elements. When making compromises, there are some approaches that give priority to To-Be state and as such are driven by characteristics of ideal state.

<table>
<thead>
<tr>
<th>Category</th>
<th>Members</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arch-shaped approaches</td>
<td>Bodhuin and Tortorella (2003); Huang (2009); Iocola (2007); Lee et al. (2004); Liu et al. (2004); Varga et al. (2004); Zhang et al. (2008); Li and Abraham (2008); O'Brien et al. (2006); Smed (2006); Ahn et al. (2008); de Castillo et al. (2009); El-Ramly et al. (2009); Gonzalez et al. (2009); Guo et al. (2009); Haidar and Abdallah (2008); Huang et al. (2008); Li et al. (2007, 2009); Zou and Kontogianni (2000); Quadrado et al. (2008); Liu et al. (2008); Arcelli et al. (2008); Bals et al. (2008); Aiahmari et al. (2010); Bissyande et al. (2010); Chenghao et al. (2010); Chung et al. (2009); Djelloul et al. (2009); Koschel et al. (2009); Marchetto and Ricca (2009); Millham (2010); Nakamura et al. (2009); Libermachine and Kerdoudi (2010); Zhang et al. (2010b)</td>
</tr>
<tr>
<td>Bow-shaped approaches</td>
<td>Chen et al. (2009); Ik et al. (2008); Sindhgatta and Ponnusamy (2008); Cetin et al. (2007a); Lavery et al. (2004); Chen et al. (2009); Cantora et al. (2008); Zhang et al. (2006); Heckel et al. (2008); Kannan and Srivastava (2008); Nguyen et al. (2009); Bao et al. (2010); Kouit et al. (2010); Minicay et al. (2010); Naser et al. (2010); Wang et al. (2010); Zhang et al. (2010b); Zhou et al. (2010)</td>
</tr>
</tbody>
</table>

3.4.4 RQ-III What are the knowledge elements that are used and produced?

By following Step 2 and Step 3 of our coding procedure, we identified the knowledge view of each of the primary studies. The knowledge view frames two aspects of each migration approach: (i) what knowledge elements are used in the migration, and (ii) how those elements are created. By comparing and categorizing the
two aspects in the primary studies, we obtained the knowledge elements that are typically used or produced in the regarding migration approaches introduced in the primary studies, as well as the types of knowledge conversions that are used to create new knowledge (i.e. RQ-III). This is the topic of this section in which the typical knowledge elements and knowledge conversions in SOA migration are discussed.

Knowledge Elements

Figure 3.8 schematically represents the generic knowledge elements used or produced in the primary studies. As shown in this figure, knowledge elements are classified into four categories. This categorization stems from the following generic distinctions in knowledge elements: (i) the distinction between tacit and explicit knowledge and (ii) distinction between problem-related and solution-related knowledge.

In Chapter 1, we explained the difference between tacit and explicit knowledge. The distinction between problem-related knowledge and solution-related knowledge has been pointed out in a number of related works [Jackson 2001, Nuseibeh 2001]. We define these two types of knowledge as follows: (i) problem-related knowledge regards the type of knowledge presenting the analysis of the problem, the problem decomposition, and the world in which it is located (problem space) (ii) solution-related knowledge addresses the solution to a specific problem domain. Since the two distinctions are orthogonal, a combination of the two results in the following four generic categories of knowledge illustrated as quadrants in Figure 3.8.

In the rest of this section we will describe each of these categories and their constituent knowledge elements. Figure 3.9, Figure 3.10, and Figure 3.11 together represent the conceptual models of the knowledge elements and the associations among them. It should be noted that, these conceptual models does not cover all possible knowledge elements that can shape the migration, but it only illustrates the ones that were observed in the primary studies.

Solution-related explicit knowledge. Solution-related explicit knowledge encompasses all externalized knowledge addressing design and implementation of both pre-existing and target systems. This category is itself categorized into Code-related and Design-related knowledge.

Code-related Knowledge. We found that more than 50% of the migration approaches use knowledge about the implementation of the actual running systems. This indicates that information about the implementation of both legacy systems and target service-based systems constitutes an important source of
3.4. RESULTS

Explicit

Tacit

Problem space

Problem Domain Attribute

Solution space

Structural Model

Behavioral Model

Problem Domain Attribute

Quality Requirements

Design solutions

Structural Model

Behavioral Model

Problem Domain Attribute

Quality Requirements

Figure 3.8: Categories of Knowledge Elements

Explicit

Tacit

Problem space

Problem Domain Attribute

Solution space

Structural Model

Behavioral Model

Problem Domain Attribute

Quality Requirements

Design solutions

Functional Decomposition Model

Structural Model

Persistent Data Model

Interaction Model

Patterns

Code Grammar

Code Documents

Coupling

Reusability

Cohesion

Statelessness

Modularization

Composability

Figure 3.9: Code-related Knowledge
knowledge for migration. We call this type of knowledge *code-related knowledge*. Table 3.6 represents the types of code-related knowledge and the number of primary studies that exploit them.

**Design-related Knowledge**  The knowledge explaining the design of both the pre-existing system (i.e. As-Is design) and the target service-based system (i.e. To-Be design) is one of the key inputs of migration approaches. Different migration approaches focus on different aspects of software design. For instance, to select the legacy elements to be migrated as a service some approaches take the functional decomposition of legacy components (components model) as knowledge input, whereas some others use the interactions between parts of the legacy system (interaction model). Table 3.7 represents the categories of design-related knowledge and the migration approaches (of primary studies) that use or produce those knowledge. Figure 3.10 provides a conceptual model of the design-related knowledge observed in the primary studies. This type of knowledge is design information that is recorded within the *software design descriptions*. The software design is described using different *design elements*.

*Design elements*, each addressing a different aspect of design, include:

- **functional decomposition model**: we found that decomposition of the software system into elements constitutes the key knowledge element of 36% of the primary studies. These studies use different models such as *component models* (e.g. *Chen et al.* (2009); *O’Brien et al.* (2005)) or *service models* (e.g. *Lewis and Smith* (2007); *Chen et al.* (2005)) to describe decomposition of the software system into elements. They mainly use this knowledge

<table>
<thead>
<tr>
<th>Knowledge</th>
<th>Primary Studies</th>
<th>Number</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code Document</td>
<td>O’Brien et al. (2005)</td>
<td>1</td>
<td>1%</td>
</tr>
<tr>
<td>Code Grammar</td>
<td>Bodhuin and Tortorella (2003); Li and Tahvildari (2008); Huang et al. (2008); Zhang et al. (2008); Cheng-hao et al. (2010); Koschel et al. (2009); Tibermacine and Kerdoudi (2010)</td>
<td>7</td>
<td>10%</td>
</tr>
<tr>
<td>Code Model</td>
<td>Lee et al. (2004); O’Brien et al. (2005); Sindhgatta and Ponnalagu (2008); Huang et al. (2008); del Castillo et al. (2009); Chen et al. (2005); Cuadrado et al. (2008); Liu et al. (2008)</td>
<td>11</td>
<td>18%</td>
</tr>
<tr>
<td>Code Itself</td>
<td>Bodhuin and Tortorella (2003); Liu et al. (2004); Varga et al. (2004); Lee et al. (2004); Zhang et al. (2008); Huang (2009); Jococa (2007); O’Brien et al. (2005); Chen et al. (2005); Sindhgatta and Ponnalagu (2008); Li and Tahvildari (2010); Ik et al. (2008); del Castillo et al. (2009); Chen et al. (2005); Cuadrado et al. (2008); Liu et al. (2008)</td>
<td>31</td>
<td>51%</td>
</tr>
</tbody>
</table>
3.4. RESULTS

to identify where a specific functionality is located.

- **structural model**: describes the internal structure and organization of design by its constituent elements such as classes, interfaces and their relationships. 26% of the migration approaches (e.g. del Castillo et al. [2009]; Canfora et al. [2008]) exploit this type of knowledge to reverse engineer or to localize a functionality in pre-existing systems.

- **persistent data model**: concerns data structures, data content and data types. Examples of such design elements are conceptual data model (e.g. Li et al. [2007] or data base schema del Castillo et al. [2009]). This type of knowledge is mainly used by the approaches that aim at identification and extraction of data services.

- **interaction model**: describes the nature of the interaction between different components and parties. Some migration approaches (e.g. El-Ramly et al. [2009]; Canfora et al. [2008]; Cuadrado et al. [2008]) exploit interaction models to identify candidate legacy elements for migration to SOA.

- **pattern**: This type of knowledge addresses reusable design ideas as patterns. Patterns address the design ideas that can be reused in all three migration processes. Examples of these patterns are SOA patterns (Kanan and Srivastava 2008; Pahl and Barrett 2006), design patterns (del Castillo et al. 2009; Arcelli et al. 2008), and architectural styles (Chen et al. 2009; O’Brien et al. 2005; Gonzalez et al. 2009).

**Design constraints**, which specify a rule or restriction on design elements or provide characteristics of design elements, also shaped SOA migration approaches. We found that using different design constraints such as coupling or cohesion the migration approaches specify the characteristic of legacy elements that are suitable to be migrated to services. For instance, the legacy elements that are loosely coupled (Chen et al. 2009; O’Brien et al. 2005; Sindhgatta and Ponnalagu 2008; Cuadrado et al. 2008; Zhang et al. 2006), or are well granular (Chen et al. 2009; Li et al. 2009; Chen et al. 2005; Nguyen et al. 2009). Table 3.7 illustrates the design constraints that are used by primary studies. In total, 7 primary studies only use design constraints as input knowledge for extracting suitable candidate services.

**Problem-related Explicit Knowledge** This category pertains to problem domain knowledge which has been made explicit and is externalized regarding the As-Is and To-Be states. Examples are business processes, business rules, requirements, cost-benefit calculations. The problem-related knowledge addresses
<table>
<thead>
<tr>
<th>Category</th>
<th>Sub-category</th>
<th>Members</th>
<th>Number</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design Element</td>
<td>functional</td>
<td>Chen et al. (2009); Sindhgatta and Ponnalagu (2008); Huang et al. (2008);</td>
<td>19</td>
<td>30%</td>
</tr>
<tr>
<td></td>
<td>decomposition</td>
<td>Zou and Kontogiannis (2008); Lewis and Smith (2007); Chen et al. (2005);</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>O'Brien et al. (2005); Cuadrado et al. (2008); Zhang et al. (2006); Canfora et al. (2008); Kanan and Srivastava (2008); Paih and Barrett (2006); Alahmari et al. (2010); Koufi et al. (2010); Marchetto and Ricca (2009); Millham (2010); Miucai et al. (2010); Nasr et al. (2010); Wang et al. (2010)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>structural</td>
<td>Li and Tahvildari (2008); Guo et al. (2005); del Castillo et al. (2009);</td>
<td>16</td>
<td>26%</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Canfora et al. (2008); Zhang et al. (2006); Alahmari et al. (2010); Bao et al. (2010); Bisyande et al. (2010); Chung et al. (2009); Koufi et al. (2010); Nakamura et al. (2009); Nasr et al. (2010); Tibermacine and Kerdoudi (2010); Wang et al. (2010); Zhang et al. (2010); Zhou et al. (2010)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>persistent data</td>
<td>Li et al. (2007); del Castillo et al. (2009); Djellou et al. (2009)</td>
<td>3</td>
<td>5%</td>
</tr>
<tr>
<td></td>
<td>interaction</td>
<td>El-Ramly et al. (2009); Canfora et al. (2008); Cuadrado et al. (2008); Alahmari et al. (2010); Bao et al. (2010); Chung et al. (2009); Miucai et al. (2010); Nakamura et al. (2009); Nasr et al. (2010)</td>
<td>9</td>
<td>15%</td>
</tr>
<tr>
<td></td>
<td>pattern</td>
<td>Kanan and Srivastava (2008); Paih and Barrett (2006); del Castillo et al. (2009); Arcelli et al. (2008); Chen et al. (2009); O'Brien et al. (2005); Gonzalez et al. (2009); Liang et al. (2009); Zhang et al. (2010)</td>
<td>9</td>
<td>15%</td>
</tr>
<tr>
<td>Design Constrains</td>
<td>coupling</td>
<td>Chen et al. (2009); O'Brien et al. (2005); Sindhgatta and Ponnalagu (2008); Cuadrado et al. (2008); Zhang et al. (2006); Marchetto and Ricca (2009); Nakamura et al. (2009)</td>
<td>7</td>
<td>11%</td>
</tr>
<tr>
<td></td>
<td>reusability</td>
<td>Chen et al. (2009); O'Brien et al. (2005); Sneed (2006); Kanan and Srivastava (2008); Alahmari et al. (2010)</td>
<td>5</td>
<td>8%</td>
</tr>
<tr>
<td></td>
<td>cohesion</td>
<td>Canfora et al. (2008); Nguyen et al. (2009); Nakamura et al. (2009)</td>
<td>3</td>
<td>5%</td>
</tr>
<tr>
<td></td>
<td>statelessness</td>
<td>Chen et al. (2008); Nguyen et al. (2009); Nakamura et al. (2009)</td>
<td>1</td>
<td>1%</td>
</tr>
<tr>
<td></td>
<td>modularization</td>
<td>Li and Tahvildari (2008); Zhang et al. (2006)</td>
<td>2</td>
<td>3%</td>
</tr>
<tr>
<td></td>
<td>composability</td>
<td>Kanan and Srivastava (2008)</td>
<td>1</td>
<td>1%</td>
</tr>
</tbody>
</table>
both the problem domain of the target service-based system as well as the problem domain realized in the pre-existing system. We observed three main types of knowledge representing the problem domain: requirements, quality requirements and problem domain attributes.

Requirements: address the functionality and behavior of the domain. Requirements are illustrated using different models, called problem elements. The problem elements are codified by two main types of models: structural and behavioral models. As an example, conceptual data models (Chung et al., 2007; Liu et al., 2008), business service blueprints (Chen et al., 2009) and use cases (Lavery et al., 2004) represent the structural problem decomposition of both pre-existing and target systems. Business processes, workflows, and scenarios represent the behavioral aspect of the problem (Nguyen et al., 2009; Alahmari et al., 2010).

Problem domain attributes: a characteristic or property of the problem domain or migration problem. Examples of the business domain properties include the following: cost of migration (Lewis and Smith, 2007; Umar and Zordan, 2009), feasibility of migration (Chen et al., 2009; Li et al., 2009), value of reuse (Sneed, 2006; Canfora et al., 2008).

Quality requirements: specify the envisioned quality requirements of the target service-based system as well as those of the pre-existing system. We found very few approaches that address quality requirements in migration. Examples of such quality requirements that shape the migration approach are: interoperability (Haidar and Abdallah, 2008), flexibility (Haidar and Abdallah, 2008), sustainability (Cuadrado et al., 2008), maintainability (Cuadrado et al., 2008), configurability (Zhang et al., 2006).
CHAPTER 3. A SYSTEMATIC REVIEW OF SOA MIGRATION APPROACHES

![Diagram showing Problem Domain Attribute, Problem Domain, Quality Requirement, Requirement, Problem Element, Structural Model, and Behavioral Model.]

Figure 3.11: Problem-related Knowledge

Table 3.8: Problem-related Knowledge

<table>
<thead>
<tr>
<th>Category</th>
<th>Sub-category</th>
<th>Members</th>
<th>Number</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements</td>
<td>Structural Model</td>
<td>Chung et al. (2007); Liu et al. (2008); Zhang et al. (2008); Chen et al. (2009); Lavery et al. (2004); Bao et al. (2010); Chung et al. (2009); Djelloul et al. (2009); Koufi et al. (2010); Marchetto and Ricci (2009); Mukany et al. (2010); Nasr et al. (2010); Wang et al. (2010); Zhou et al. (2010); Zhang et al. (2010b)</td>
<td>15</td>
<td>25%</td>
</tr>
<tr>
<td></td>
<td>Behavioral Model</td>
<td>Lavery et al. (2004); Kannan and Srivastava (2008); Nguyen et al. (2009); Alahmari et al. (2010); Koufi et al. (2010)</td>
<td>5</td>
<td>8%</td>
</tr>
<tr>
<td>Problem Domain Attribute</td>
<td></td>
<td>Lewis and Smith (2007); Umar and Zordan (2009); Chen et al. (2009); Li et al. (2010); Sniecut (2008); Canfora et al. (2008); Zhang et al. (2010b)</td>
<td>6</td>
<td>10%</td>
</tr>
<tr>
<td>Quality Requirements</td>
<td></td>
<td>Haidar and Abdallah (2008); Cuadrado et al. (2008); Zhang et al. (2009); Alahmari et al. (2010)</td>
<td>4</td>
<td>6%</td>
</tr>
</tbody>
</table>
### 3.4. RESULTS

**Solution-related tacit knowledge**  This category includes intangible knowledge of the practitioner that is used to provide solutions at different stages of migration. For instance, in (Zhang et al., 2010b) the architects (out of experience) knows the services that are most reusable. Likewise (Li et al., 2007; Alahmari et al., 2010) exploit tacit knowledge of the architect about the candidate legacy elements for migration. Overall, the solution-related tacit knowledge was rarely observed in the primary studies.

**Problem-related tacit knowledge**  This category includes problem domain knowledge that reside in practitioners minds and are not codified in documents. Few primary studies rely on tacit problem-related knowledge (Lavery et al., 2004; Lewis and Smith, 2007; Umar and Zordan, 2009).

### Knowledge Conversions

So far, we have addressed the type of knowledge that are used in migration. In this section we discuss the interactions and conversions between knowledge elements. The migration approach can be seen as a series of knowledge conversions between **tacit** and **explicit** as well as **solution-related** and **problem-related** knowledge. As an example, the migration approach of Nguyen et al. (2009) starts with capturing the tacit knowledge of **most beneficial business functions** and representing those business functions in a **business service model** (i.e. conversion of **tacit** to **explicit**).

Next, the business service model is transformed to software service model (i.e. conversion of **problem-related** to **solution-related** knowledge).

Knowledge conversions embedded in the migration can distinguish different approaches. For some approaches, the migration mainly addresses the transition from solution-related to solution-related knowledge. While for some others, transition from solution-related knowledge to problem-related is also supported. Based on categorization presented in Figure 3.8, in total, there are 16 possible types of knowledge conversion that are determined by pairing one of the four conversions between tacit and explicit with one of the four conversions between problem-related and solution-related knowledge. Out of these 16 conversions, 7 were found in the primary studies. Figure 3.12 presents those knowledge conversions using solid arrows, that are further described in the following. To illustrate the the knowledge conversions between tacit and explicit, we use the terms introduced by Nonaka and Takeuchi including: **socialization** (tacit to tacit), **externalization** (tacit to explicit), **internalization** (explicit to tacit), and **combination** (explicit to explicit).

- **(a) Combination of problem-related knowledge.** This type of conversion is used to explore the problem-space (related to both As-Is and to To-Be states) by combining the problem related knowledge elements and creating
new knowledge elements out of existing ones. For example, [Umar and Zordan (2009)] create cost-benefit calculations of target system out of existing ones. [Lewis and Smith (2007)] create service requirements out of goals and critical business process. While exploration of problem space is crucial for determining the solution for migration, only 18% of the primary studies cover this type of conversion.

• **(b) Externalization of problem-related knowledge.** This conversion relates to making tacit problem-relates knowledge explicit. 29% of the primary studies, as a part of their migration approach, address this conversion. Some externalize goals ([Umar and Zordan (2009)]), some business processes ([Chung et al. 2007]; [Liu et al. 2008]; [Nasr et al. 2010]), and some constraints ([Haidar and Abdallah 2008]; [Cuadrado et al. 2008]).

• **(c) Combination of problem-related and solution-related knowledge.** This conversion addresses transformation of explicit problem-related knowledge (e.g. business processes) to the solution-related knowledge (e.g. service compositions). This type of conversion was observed frequently in the primary studies (see Table 3.9).

• **(d) Combination of solution-related knowledge.** Generally, the transformation of artifacts that are containers of solution-related knowledge within each of the reverse engineering, transformation and forward engineering processes corresponds to this type of knowledge conversions. For instance, within the reverse engineering activity the existing legacy code or design (i.e. explicit solution-related knowledge) can be converted to legacy architecture (e.g. [Zhang et al. 2006]; [Liu et al. 2008]; [Canfora et al. 2008]). Another example of such conversion is when existing architecture is transformed to the target service composition (e.g. [Arcelli et al. 2008]; [Kannan and Srivastava 2008]; [Pahl and Barrett 2006]). Not surprisingly, almost all of the primary studies covered this conversion.

• **(e) Externalization of problem-related to solution-related knowledge.** This conversion reflects the transformation of tacit problem-related knowledge to explicit solution-related knowledge. As an example, within the reverse engineering process, tacit problem-related knowledge such as most beneficial business rules can be exploited to extract legacy elements with high reuse potential ([Sneed 2006]). Also, this type of conversion occurs in forward engineering process when assumptions such as business value of a service (categorized as problem-related tacit knowledge) are taken into account in
candidate service identification\cite{Canfora2008}. Only 4% of migration approaches covered this conversion.

- (f) \textit{Externalization of solution-related knowledge}. This type of knowledge conversion occurs considerably in the SOA migration approaches in practice. As an example, an architect knows the legacy segments that are good candidates for SOA migration and comes up directly with the regarding service design. This conversion was only observed in\cite{Li2007, Alahmari2010}.

- (g) \textit{Internalization of problem-related knowledge}. This conversion addresses learning the discrepancies and mismatches within existing legacy assets from the explicit, problem-related knowledge (e.g. existing business processes or business rules). Few of primary studies (i.e.\cite{Lavery2004, LewisSmith2007, UmarZordan2009}) that provide a type of gap analysis among business domain models explicitly covered such conversion.

\section{3.5 SOA Migration Frame of Reference}

The frame of reference for SOA migration we developed depicts the categories identified in this SLR and introduced in Section\ref{sec:gap_analysis} and\ref{sec:knowledge_conversion}. As shown in Figure\ref{fig:knowledge_conversions}, this frame of reference structures the categories of (i) the knowledge
### Table 3.9: Knowledge Conversions

<table>
<thead>
<tr>
<th>Conversion Members</th>
<th>Number</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>11</td>
<td>18%</td>
</tr>
<tr>
<td>Lavery et al. (2004); Lewis and Smith (2007); Hutchison et al. (2007); Umar and Zordan (2009); Chung et al. (2009); Alahmari et al. (2010); Bao et al. (2010); Kouhi et al. (2010); Millham (2010); Nasr et al. (2010); Zhang et al. (2010b)</td>
<td>11</td>
<td>18%</td>
</tr>
<tr>
<td>b</td>
<td>17</td>
<td>29%</td>
</tr>
<tr>
<td>Lewis and Smith (2007); Umar and Zordan (2009); Li et al. (2009); Canfora et al. (2008); Chung et al. (2007); Liu et al. (2008); Zhang et al. (2006); Chen et al. (2009); Lavery et al. (2004); Kanan and Srivastava (2008); Nguyen et al. (2009); Haidar and Abdallah (2008); Canfora et al. (2008); Alahmari et al. (2010); Nasr et al. (2010); Zhang et al. (2010b)</td>
<td>17</td>
<td>29%</td>
</tr>
<tr>
<td>c</td>
<td>57</td>
<td>91%</td>
</tr>
<tr>
<td>Chen et al. (2009); Ilk et al. (2008); Sindhgatta and Ponnalagu (2008); Cebis et al. (2007); Chen et al. (2005); Zhang et al. (2006); Heckel et al. (2008); Kannan and Srivastava (2008); Nguyen et al. (2009); Chung et al. (2009); Kouhi et al. (2010); Millham (2010); Mulcahy et al. (2010); Wang et al. (2010); Zhou et al. (2010)</td>
<td>57</td>
<td>91%</td>
</tr>
<tr>
<td>d</td>
<td>3</td>
<td>4%</td>
</tr>
<tr>
<td>Zhang et al. (2006); Liu et al. (2008); Canfora et al. (2008); Arcelli et al. (2008); Kannan and Srivastava (2008); Pahi and Barrett (2006); Bao et al. (2010); Bisyande et al. (2010); Chenghao et al. (2010); Chung et al. (2009); Djeouli et al. (2009); Koschel et al. (2009); Kouhi et al. (2010); Milham (2010); Marchetto and Ricca (2009); Mulcahy et al. (2010); Nakamura et al. (2009); Nasr et al. (2010); Tiberachine and Kerdoudi (2010); Wang et al. (2010); Zhang et al. (2010b); Zhou et al. (2010)</td>
<td>3</td>
<td>4%</td>
</tr>
<tr>
<td>e</td>
<td>3</td>
<td>4%</td>
</tr>
<tr>
<td>Sneed (2006); Canfora et al. (2008); Zhang et al. (2010b)</td>
<td>3</td>
<td>4%</td>
</tr>
<tr>
<td>f</td>
<td>2</td>
<td>3%</td>
</tr>
<tr>
<td>Li et al. (2007); Alahmari et al. (2010)</td>
<td>2</td>
<td>3%</td>
</tr>
<tr>
<td>g</td>
<td>3</td>
<td>4%</td>
</tr>
<tr>
<td>Lavery et al. (2004); Lewis and Smith (2007); Umar and Zordan (2009)</td>
<td>3</td>
<td>4%</td>
</tr>
</tbody>
</table>

For selecting or defining a migration approach for a certain context, the frame of reference lends itself to a generic method in which first the knowledge view elements and the conversions among them, and (ii) activities covered. The SOA migration frame of reference serves to select existing migration approaches, or even to drive the development of new approaches. In this regard, using this frame of reference one can define the Two-View representation of the fitting approach for migration. In such a representation the knowledge view frames the knowledge elements that are feasible and favorable to be made available while the activity view frames the activities that has to be done. Such knowledge and activity views characterizes the suitable migration approaches (to be selected or newly defined).
3.5. SOA MIGRATION FRAME OF REFERENCE

is defined. Next, by providing the link between knowledge and activity views, it helps to arrive from the knowledge view to the activity view. What lies in the heart of this link is the desired knowledge conversions and the migration families able to realize those conversions. Figure 3.13 III shows the mapping of knowledge conversions and the families supporting those conversions. Here follows our reasoning behind this mapping.

The knowledge conversions between problem-related and solution-related knowledge are the key in identifying the required activities for migration. Considering the transformation process, for instance, in case the desired knowledge conversions reside in the solution space (e.g. modules are converted to web services), then automatic transformations can be a candidate. This make families supporting this type of transformation (e.g. F1, F4, and F5) suitable candidates for migration. On the other hand, if the transformation embraces the problem to problem conversion (e.g. conversion of legacy business processes to business services), a mapping mechanism (e.g. gap analysis) performed by experts is more suitable. This can be supported only by families covering transformation at top most level (i.e. F3 and F8).

Figure 3.13 IV pairs the knowledge conversions and migration families. The schematic forms in this figure represent the activities that realize the conversions. For instance, arrow X (i.e. the generalized form of conversions c, j, e,h) is paired with activities shown in bold in right hand of Figure 3.13 IV. These activities, as can be seen from Figure 3.13 III, are covered only in Families F2, F5, and F8.

To sum up, the SOA migration frame of reference would facilitate extracting the skeleton of the desired migration approach in form of the Two-View representation. Having the Two-View representation at hand one can select and/or define the suitable migration approach that best fits the specific characteristics and needs of a certain case. This is exemplified in the next section where the frame of reference is used to select an approach for a real-world migration case.

3.5.1 Frame of Reference Application Example

This section uses the frame of reference in an example migration case and determines the migration approach that fits the context and goals of that certain case. To provide a realistic example, we took a real-world migration case from our work on an industrial SOA migration presented in Chapter 4. In the following, we first introduce the example case and further we exemplify the uses of the frame of reference in defining the fitting migration approach for this case.

Example case

For the sake of simplicity, some details of this case are ignored. Our example case is an enterprise that wants to replace a pre-existing system with a new service-
CHAPTER 3. A SYSTEMATIC REVIEW OF SOA MIGRATION APPROACHES

I. Knowledge Elements

II. Knowledge Conversion

III. Activity Coverage of Migration Families

IV. Mapping
Knowledge Conversions to Migration Families

Figure 3.13: A Frame of Reference for SOA Migration

58
based system. Although the ultimate goal is replacing the pre-existing system, yet the enterprise aims at reusing the pre-existing functionality of the legacy system. Consequently, the enterprise targets migration of the legacy system to SOA. In short, there are two main goals for this migration: (g1) to incorporate a set of new goals and requirements, and (g2) to achieve flexibility. Moreover, the migration in this enterprise has the following characteristics:

- (c1) The knowledge about the pre-existing system mainly resides in the stakeholders’ minds (e.g. maintainer, developer, and architect). As such, the stakeholders know what functionalities (as well as non-functionalities) are available, and where those functionalities are located in the legacy system. Documentations related to architecture description and database schema of the legacy system are still available.

- (c2) The availability of the legacy system is restricted to working hours, as
the transactions are executed in batch after working hours.

- (c3) There is a local best practice for understanding the As-Is and To-Be states i.e. gaining understanding through extracting the information architecture (representing the business entities), and the functionality architecture (representing the business functionalities).

Defining migration approach of the example case using the frame of reference

1. Identifying the knowledge elements. For defining the migration approach suitable for the example case, first the relevant knowledge elements need to be identified. To this aim, the knowledge elements shown in Figure 3.13.I act as a checklist of relevant types of input knowledge. Using this checklist one could ask him/herself whether each type of knowledge is available or whether it is desirable and feasible to be further produced. This way the knowledge elements driving the migration are identified. Based on the goals and characteristics explained previously, and by using the knowledge view of the frame of reference, we elicited the knowledge elements that have to be externalized in this migration example. The knowledge elements marked in white in Figure 3.14.I exemplify the elements that would be already available in this certain example, whereas, the grey ones would indicate the knowledge elements that are currently unavailable.

As noted, to identify the relevant knowledge elements for a certain migration one needs to check the knowledge elements of each the knowledge categories in Figure 3.13.I (i.e. code-related, design-related and problem-related). For instance, in this example from the Design-related knowledge category in Figure 3.13.I, Functional Decomposition Model is a relevant type of knowledge which is already available in form of architecture descriptions (see characteristic c1 in the example case). Thus, the architecture descriptions is an available input knowledge that should be marked white in the knowledge view (see Figure 3.14.I). Likewise, Persistent Data Model would be relevant for this example since data base schema is available which can facilitate extraction of services of type data. In summary, by acting as a checklist the knowledge elements of the frame of reference (Figure 3.13.I) aid to identify the input/output knowledge that would drive this example migration.

2. Identifying the knowledge conversions. After identifying the available and unavailable knowledge elements one can decide on how to elicit the unavailable knowledge. If the unavailable knowledge is of type tacit then it might be desirable to either codify it (i.e. externalization) or to make it
available (e.g. by keeping track of who knows what) while it still remains tacit in stakeholders minds (socialization). In addition, if a knowledge is unavailable one might decided to create it out of other knowledge (i.e. combination conversion mode). For instance, creating business services from business processes.

Similar to knowledge elements, the description of knowledge conversions plays the checklist role for identifying a certain knowledge conversions. Consequently, the modes of knowledge conversion (i.e. externalization, combination, internalization and socialization), covered in the example case are determined.

As an example, for As-Is and To-Be business processes (see Figure 3.14.I) we need to go from tacit to explicit (i.e. Externalization mode). Explicit knowledge here takes the form of business process models. The conversion (b) is covered, since the business processes of both As-Is and To-Be states have to be modeled. In the same vein, information architecture and functionality architecture has to be externalized too.

To create the output knowledge elements we need to go from explicit to explicit knowledge, that is, to combine knowledge from different sources (i.e. Combination mode). For instance to create business services we need to combine information architecture and functionality architecture (conversion (a)). Likewise, using the data base schema, data flow and data model can be created (i.e. conversion (c)). All in all, knowledge conversions (a) and (c) need to be supported in this migration example (see Figure 3.14.II).

3. Identifying the activity coverage. As noted, the knowledge conversions pinpoint the generic types of activities that have to be carried out. Considering the knowledge conversions of this example, only family F8 can realize conversions (a), (b) and (c). This suggests migration of type forward engineering with gap analysis being a suitable choice for this example. Being a F8 migration approach implies that reverse-engineering process is not covered and understanding about the existing system can be achieved by consulting stakeholders, rather than generating the abstractions from the existing code. Figure 3.14.III illustrates the regarding activity coverage. The business model transformation makes gap analysis between As-Is and To-Be business processes. Within forward-engineering process services are analyzed, designed, and implemented. Finally, legacy element transformation activity transforms the legacy functionality to service model.

Taken together, an approach similar to the one presented in (Nguyen et al., 2009) appears to be the most fitting one for this certain migration.
CHAPTER 3. A SYSTEMATIC REVIEW OF SOA MIGRATION APPROACHES

3.6 Discussion

In the following we discuss the possible threats to validity of our analysis followed by our additional observations.

3.6.1 Threats to Validity

**Internal Validity** Internal validity aims at ensuring that the collected data enables researchers to draw valid conclusions [Creswell (2003)]. This validity relates to “how” the research is carried out, and whether the used methods are credible. One of the threats to internal validity of the study is that the review is mainly conducted by two researchers. However, subjective interpretations are mitigated by both following a systematic protocol, checked and validated by senior researchers experienced in software engineering, systematic reviews and SOA and validating it further using a pilot study. Additionally, we explicitly included only publications whose objective is to present a solution for migration. It is possible that a publication proposes also a solution for migration blended with other objectives, so that the contribution on migration is not clearly represented. To mitigate this threat, we added some more generalized keywords such as ‘reuse’ in the search terms. To further ensure the unbiased selection of articles, a multistage process was utilized that involved two researchers who documented the reasons for inclusion/exclusion of every step as described in §3.3.3.

Another threat is appraising the quality of published research based on their report on evaluation of the work as journal articles and, in particular, conference papers rarely provide enough detail of the evaluation of their work due to space limitations in journal volumes and conference proceedings. There is therefore a danger that what is being assessed is the quality of reporting rather than the quality of research. To mitigate this threat we contacted the authors of the primary studies and checked if there exists any publication stating or reporting the application of their approach.

Threat to validity of data extraction process is that the primary studies lack sufficient information for us to be able to document their migration approaches satisfactorily in the data extraction forms. There is therefore a possibility that the extraction process may have resulted in some inaccuracy in data. Nevertheless we mitigated this threat through consensus meetings. In the data extraction process each primary study was read by two reviewers. Once reviewer acted as the main data extractor, whilst the other reviewer acted as a checker. Any disagreements were discussed in the data extraction consensus meetings.

Threat to validity of the analysis is in the general applicability of the codes used for characterizing and classifying migration approaches. An assuring factor in this regard is that the start-list of codes is extracted from our SOA-MF
framework published in a service-oriented computing forum, after being peer reviewed by experts in the field (Razavian and Lago, 2010). This framework stems from existing theory on reengineering and architectural recovery while it is constantly refined through our coding procedure. This further consolidates its general applicability.

External Validity  External validity defines to what extent findings from the study can be generalized (Creswell, 2003). Publication bias is a threat to external validity of this study in that the scope of our review is restricted to the scientific domain. The threat here is that very relevant migration approaches originated in industry, if not described in scientific publications, are not covered. This gap is filled in Chapter 4 reposting the results of a survey focusing on how SOA migration is performed in industrial practice.

3.6.2 Additional Observations

In summary, besides the frame of reference presented so far our study yields the following complementary findings:

Reverse engineering approach to migration  Given the migration families, the number of primary studies in each of those families brings an insight about the current focus of the subject in the field. Families with the main focus on reverse engineering (i.e. F2, F3, and F6) initially attracted a lot of attention. The migration approaches in those families mostly take a reverse engineering approach to define methods, tools and techniques porting legacy assets to service-oriented ones. The families taking a forward engineering approach (i.e. F5, and F8), however, mostly elicit important knowledge about the legacy assets and re-engineer them into services in a top-down manner. Interestingly, the families centering around forward engineering are more recent. Accordingly, we conjecture that although academic approaches are still dominated by reverse engineering perspective, there is an increasing interest on forward engineering ones. In our opinion, this is in line with the shift in how SOA is perceived: from a technology enabler to a software engineering paradigm (Scholler, 2012). More precisely, in the SOA hype organizations perceived SOA as a technology enabler that facilitates integrating their existing applications, while now SOA is perceived as a paradigm that guides the development of enterprise services from existing assets.

As-Is driven migration  Given the two categories of bowl-shaped and arc-shaped approaches, we found that majority of academic approaches are arc-shaped. This implies that understanding As-Is state and modernizing As-Is legacy
assets constitutes the common view on migration in the field. This further confirms that in academia the main focus is on modernizing legacy assets, instead of identifying ideal services and reengineering legacy assets to realize those services.

**Focus on explicit solution-related knowledge** Considering the types of knowledge used and produced, we found that that academic approaches mainly focus on explicit solution-related knowledge. This is evident in Figure 3.12 where most of identified knowledge elements reside in the explicit solution-related square. This implies that, academic approaches mainly rely on how legacy assets are already implemented or how the services should be designed and implemented, instead of what problems or requirements that the legacy assets already address or To-Be services need to address. In other words important tacit knowledge is mainly neglected.

**Focus on externalization of knowledge** Considering the knowledge conversions involving tacit and explicit knowledge, the migration approaches cover mainly explicit to explicit and tacit to explicit knowledge conversions, but not conversions resulting in tacit knowledge. The former conversions capture the idea that knowledge has to be externalized, meaning that the tacit type of knowledge should be converted to the explicit knowledge. This, however, is not the only possible conversion. According to Nonaka and Takeuchi (1995) tacit and explicit knowledge should mutually interact. The explicit to tacit conversion, mainly centers around learning or gaining insight from the existing explicit knowledge. Only few approaches cover such conversions (approaches including conversion (g) in Figure). Interestingly, knowledge conversions that include learning or gaining insight from existing explicit knowledge are widely used in practice. This highlights the need for further research to support activities that embrace the explicit to tacit knowledge conversion. Interestingly, our industrial studies (Chapter 6,7) revealed that knowledge conversions which include taking externalized knowledge and making it into individual tacit knowledge in the form of mental models or know-how are widely used in practice (i.e. internalization). In addition, transfer tacit knowledge to another person through observation or “learning by doing” (i.e. socialization) are very common in industrial practice. Getting back to Figure 3.12 the knowledge conversions illustrated as dashed arrows, represent the conversions that occur frequently in practice and were not found back in the primary studies. These knowledge conversions are described in the following.

- (h) Explicit solution-related knowledge of the pre-existing legacy system may consequently lead to new ideas, insights and goals concerning the target SOA environment.

- (i) Based on experience, a software engineer could arrive directly from an
existing solution in the legacy system to a solution in SOA environment. This way, the migration process skips the intermediate conversions between problem-related to solution related knowledge (within forward-engineering), and vice-versa (within reverse-engineering). Consequently, the intermediate knowledge is never externalized. This kind of knowledge is thus tacit, since the software engineer “just knows” it fits the problem.

- (j) Explicit solution-related knowledge of the pre-existing system such as existing architecture or design models are converted to the problem-related knowledge such as existing business processes. This type of knowledge conversion is associated to business process extraction activity within the reverse engineering process.

This highlights the need for further research to support activities that embrace the explicit to tacit knowledge conversion.

**Lack of abstraction levels at concept level** Considering the abstraction levels, SOA-MF presents four levels of abstraction including: ‘code’, ‘basic design element’, ‘composite design element’ and ‘concept’. The ‘code’ level deals with the actual running system, the ‘basic design element’ and ‘composite design element’ levels represent the design solution, whereas the ‘concept’ level represents the problem. In a number of primary studies, design solution artifacts reflect two layers of atomic element (e.g. modules), and composite element (e.g. architecture). In the same vein, we expected to observe similar layering scheme for the concept level artifacts. For instance, having two layers of business services (i.e. atomic element) and business processes (i.e. composite element). This layering scheme, for example, has been leveraged in (Papazoglou, 2007) in which business services play a mediator role to articulate business processes, with the underlying solution services. Only one of the primary studies, however, distinguished the abstraction levels in the concept level (Nguyen et al., 2009). This motivates further research since the importance of having different abstraction levels at the ‘concept level’ has been considerably acknowledged in practice.

**Lack of focus on quality requirements** One of the main concerns of enterprises carrying out SOA migration should be whether or not the target service-based system can meet the pre-existing or envisioned quality requirements. Migration to SOA can have positive impact on some quality requirements and negative impact on some others. To support the quality requirements, the migration approach has to incorporate the identification, analysis and realization of them. In other words, the quality requirements should drive the migration as well. During the analysis, however, we only found 4 studies (see Table 3.8) that explicitly address some quality requirements in their migration approach. This shows an
important shortcoming since one of the key challenges of migration in practice is how to identify, transform and develop services that realize the required quality requirements.

3.7 Conclusion

With the availability of different migration approaches at hand, it is hard for a practitioner in-the field to devise an approach. To facilitate devising such an approach, this paper has presented our concept of SOA migration frame of reference. The frame of reference categorizes the migration approaches with respect to knowledge and activity views. Such categorization brings order in the existing SOA migration approaches and consequently provides a “bird’s-eye” view of “what SOA migration entails”.

To migrate the pre-existing legacy assets to services, the frame of reference is an essential tool for selecting or defining the most fitting migration approach in the following way: first, the knowledge view of the frame of reference helps to identify the knowledge elements and knowledge conversions that shape the migration; second, the activity view supports helps to decide what activities have to be covered. In this way, practitioners can represent their migration in two views of knowledge and activity. Representing the migration approach using the two views, instead of a single overly complex representation that mixes different aspects of the approach, further serves its understandability.
4

On The Differences Between Academic and Industrial Migration Approaches

Chapter 3 presented the SOA migration approaches defined in academia. There is, however, considerable difference between those academic approaches and those emerged in industry. This difference pinpoints a potential gap between theory and practice which is the topic of this chapter. To bridge this gap, we conducted an industrial interview survey in seven SOA solution provider companies. Results have been analyzed with respect to migration activities, the available knowledge assets. In addition, industrial approaches have been contrasted with academic ones, hence discussing differences and promising directions for industry-relevant research.

4.1 Introduction

So far, many SOA migration approaches have been proposed in both industry and academia with the ultimate goal of adoption in practice. There is, however, considerable difference between SOA migration approaches defined in academia and those emerged in industry. For example, while scientific approaches mainly take a reverse engineering perspective, industrial practitioners developed best practices in forward engineering from requirements to SOA technologies, where legacy code is not transformed but used as a reference. This difference pinpoints a potential gap between theory and practice. One of the key causes of such a gap is that the approaches proposed in academia do not fully fit the main goals and needs of practice. To bridge this gap, it is necessary to understand the properties of migration approaches that are both feasible and beneficial for practice.

This chapter provides an understanding of the types of migration approaches in industrial practice. To this end, we conducted an industrial interview survey in seven leading SOA solution provider companies. To the best of our knowledge,
CHAPTER 4. ON THE DIFFERENCES BETWEEN ACADEMIC AND INDUSTRIAL MIGRATION APPROACHES

this is the first survey of this kind. With the objective of understanding the industrial migration approaches, we designed and executed the interviews. Each interview was analyzed considering the activities carried out and the available knowledge assets. Furthermore, we looked for the best practices that companies have developed out of experience for successful legacy migration.

As a result we found that unlike the majority of academic approaches, SOA migration in industry mostly neglects reverse engineering. Rather, migration follows a forward engineering process initiated by identifying the ideal state (e.g. ideal business services), which is taken as a reference to extract and transform legacy elements to services. In addition, we contrasted the industrial approaches with academic ones, which we identified from the Systematic Literature Review (SLR) reported in Chapter 3. Here we use the results of the SLR to discuss the differences and draw promising directions for industry-relevant research.

4.2 Design of Interview Survey Study

This section describes the research methodology we followed in our industrial survey. The following introduces the research questions, the study design, and our data analysis method.

4.2.1 Research Questions

In order to investigate how migration is performed in industrial practice the following main research question was formulated: What approaches regarding legacy to SOA migration are used in practice? This main question is refined as:

- (RQ-I) what are the activities carried out?
- (RQ-II) what is the sequencing of the activities?
- (RQ-III) can we recognize any practices used in SOA migration?

4.2.2 Study Design

We chose interview survey as our research method for two main reasons: (i) Survey is the favorable method for the type of questions our research seeks to answer. In Yin’s categorization (Yin 2008) our research questions fall under “what” category (as we look for what activities and what knowledge). These questions are likely to favor surveys, according to Yin (2008). (ii) Our research is of explorative nature, hence requiring multiple data points. Surveys again are a suitable method for this purpose as they address a larger target population as compared to other research methods such as case studies.
To gather information about industrial SOA migration approaches, we conducted a series of interviews. Interviews are an appropriate strategy when the goal is to identify the experience of individuals and/or organizations in carrying out a task [Seidman (2006)]. The conducted interviews were semi-structured. The open-ended questions of this type of interviews allow interviewers to ask follow-on questions when necessary. The first version of the interview guide was piloted with one researcher and one practitioner with experience in SOA migration [1].

The theoretical population (Yin, 2008) target of this study included architects with considerable experience in carrying out SOA migration projects. To cover both high-level business view as well as technical view on the topic, we intentionally included both enterprise- and IT architects in our study. To recruit architects fitting our requirements we created a leaflet illustrating the goals of this study. We distributed it both among our industrial network and in the Dutch architecture conference 2009, a well-known practitioner conference with over 500 participants each year. To increase the generalizability of the results we intentionally chose both in-house (who migrate their own legacy systems to services) and consultancy companies (who support customer organizations to migrate to services). To ensure that the industrial survey is based on real-world experience, instead of participant’s opinion on how migration should be carried out, we asked them to select one recent project and answer the questions considering that specific project. Finally nine architects, affiliated with seven international companies residing in the Netherlands and Belgium committed to our study. Table 4.1 provides an overview of the information about the interviewees, their company and the domain of the migration project selected for each interview.

Before carrying out the interviews, the interviewees were sent a copy of the interview guide as well as some background information on the study. This allowed us to better synchronize terminology. The interviews were conducted on site and were all video-recorded. The initial findings along with any remaining unanswered questions were iterated with the interviewees to reassure their correctness and completeness.

### 4.2.3 Data Analysis

To typify the industrial migration approaches, we analyzed the interview transcripts. The analysis resulted in the categorization of the migration approaches based on the activities carried out and the knowledge used and produced. To categorize the industrial migration approaches, similar to SLR, we mapped them on SOA-MF. To carry out the mappings systematically, we used the coding procedure represented in §3.3.3.

---

1 The interview guide is provided in Appendix B
CHAPTER 4. ON THE DIFFERENCES BETWEEN ACADEMIC AND INDUSTRIAL MIGRATION APPROACHES

Table 4.1: Interviews Overview

<table>
<thead>
<tr>
<th>Interviewee code</th>
<th>Interviewee role</th>
<th>Company ID</th>
<th>Company Type</th>
<th>Company Size</th>
<th>Experience with SOA</th>
<th>Project Application Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>RVS Ent. Arch A</td>
<td>in-house 82,500</td>
<td>11 yrs.</td>
<td>Telecom</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CXB IT. Arch B</td>
<td>Consultancy 50</td>
<td>9 yrs.</td>
<td>Public</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVD Ent. Arch C</td>
<td>Consultancy 39,000</td>
<td>11 yrs.</td>
<td>Call Centers</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JXM IT. Arch D</td>
<td>in-house 10,000</td>
<td>10 yrs.</td>
<td>Banking and Insurance</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PXB Ent. Arch D</td>
<td>in-house 10,000</td>
<td>10 yrs.</td>
<td>Banking and Insurance</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DXL IT. Arch E</td>
<td>Consultancy 388,000</td>
<td>11 yrs.</td>
<td>Utility and Energy</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GWH Ent. Arch E</td>
<td>Consultancy 388,000</td>
<td>11 yrs.</td>
<td>Utility and Energy</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RXB Ent. Arch F</td>
<td>Consultancy 350</td>
<td>8 yrs.</td>
<td>Finance</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JWV Ent. Arch G</td>
<td>Consultancy 10</td>
<td>3 yrs.</td>
<td>Finance-Payroll</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4.3 Results

To gain an understanding on industrial migration approaches, we needed to typify the approaches in a unified manner. For this purpose, we used the SOA Migration Framework (SOA-MF) introduced in Chapter 2. The analysis of the approaches revealed patterns common among various companies. These are listed in four key findings presented in this section. Findings F1 and F2 address RQ-I and RQ-II respectively. Findings F3 and F4 pertain to RQ-III. Table B.1 provides a general overview of the findings. Each finding is summarized in a Reflection Box, which is followed by detailed discussion of the finding. Furthermore, each finding is compared with the results of our previous study on academic SOA migration approaches presented in Chapter 3. Major differences between industrial approaches and academic ones can reflect gaps between theory and practice.

4.3.1 Migration Activities

Reflection Box 1.

- **F1.1.** Different companies share the same set of activities for migration.
- **F1.2.** Industrial migration approaches converge to one, common, type of migration.
To answer what is done in industrial approaches, we identified the constituent activities of various approaches and mapped them on SOA-MF. Figure 4.1 II, represents the schematic forms of those mappings. Mappings revealed two main findings: (i) industrial approaches share the same set of activities for migration and (ii) industrial approaches are convergent to a subset of those activities. The two findings are further discussed in the following.

Finding F1.1. Various companies, independent from the company type (i.e., consultancy vs. in-house) and migration application domain, share the same set of activities for migration. This is evident from Figure 4.1 II, where the activities correspond to two graphically similar coverage patterns. It should be noted that the similarity among coverage patterns, thanks to expressiveness of SOA-MF, indicates the conceptual similarity of constituent activities and artifacts of the migration approaches. In Chapter 3, SOA migration approaches with similar set of activities constitute a migration family. Similarly, the two similar approaches identified in the interviewed companies belong to the same family.

Contrast with theory. While the industrial approaches are all members of one family, the SLR revealed that the academic approaches belong to eight very different families. By covering different sets of activities each of these eight fami-
ilies provide a very different view on what SOA migration entails. For instance, one family reverse engineers the legacy code and transforms the extracted code segments to services, another family only covers the forward engineering process. Considering the industrial approaches, all the approaches are categorized into (only) one of the eight families. Interestingly, the size of that family, called industrial family, is the smallest as compared to the others (i.e. 2% of academic approaches). Thus, one could conclude that 98% of the academic approaches do not fit in industrial family. This may indicate that academic research might be digging into aspects (like processes and techniques) that are less relevant for industry. On the contrary, by looking at the characteristics of the industrial family research could better focus on the open research questions pertaining such family and hence have a better chance to close the gap between academic research results and industry needs.

Finding F1.2. By further analyzing the activities of the industrial approaches, we found that those common among all approaches, called core activities, are the ones shown in Figure 4.1.I with bold boxes. The variable activity, i.e., the one not common to all approaches, pertain to the coverage of the transformation activity shown in Figure 4.1.I by dashed line box. Furthermore, we observed that the core activities are those performed more frequently and systematically, while the variable activity is carried out less frequently and in an ad-hoc manner. More precisely, the limitations posed by legacy systems makes the variable activity less frequent. The main reason brought by the interviewees is that legacy systems were monolithic and not decomposable, making it infeasible (or cost-inefficient), for instance, to restructure the architecture. Furthermore, we observed that core activities are mainly supported by the state-of-the-practice methodologies and techniques such as SOMA (Arsanjani et al., 2008). The variable activity, however, is mainly carried out using local best practices. Consequently, we argue that, due to higher feasibility of the core activities and support of well-established methodologies and techniques, the industrial migration approaches are characterized by core activities. Chapter 5 explains how migration in industrial practice is performed using the core activities.

Contrast with theory. None of the migration approaches in the SLR fully covers the core activities. I.e., none of the academic approaches comprehensively supports the type of migration that is both feasible and beneficial in industrial setting. This indicates an important gap between the migration activities emerged from practice and the ones researched in academia. Furthermore, even the support of academic approaches for each of the core activities, separately, was limited. More precisely, only 10% of the academic approaches cover business domain transformation activity (i.e. F2 and F8 family members); 20% of the approaches (i.e. F3 family members and 2 members of F6) cover code transformation activities and, finally, 10% cover forward engineering activities (i.e. F5
4.3. RESULTS

and F8 family members).

4.3.2 Sequencing of Migration Activities

Reflection Box.2.

- **F2.** In the industrial migration approaches the To-Be state initiates and drives the migration.

By providing the mappings on SOA-MF, previous section addressed what activities are covered in the industrial migration approaches. Here we focus on what is the sequencing of those activities. The SLR resulted in two types of sequencing of activities in the migration approaches, namely arc-shaped and bowl-shaped (see §3.4.3). In summary, in arc-shaped approaches migration is driven by As-Is state, while it is the To-Be state that drives the bowl-shaped ones. All the industrial approaches elicited by our study were bowl-shaped.

The sequencing of activities in an arc-shaped approach starts from the reverse engineering process. In this category, the As-Is state initiates and drives the migration. Unlike the arc-shaped category, the bowl-shaped one starts from forward engineering and the To-Be state is the main driver of migration.

**Finding F2.** The bowl-shaped sequencing of activities in industrial approaches implies the following: in all of the migration approaches the To-Be state, characterized by requirements or properties of the target service-based system, drives and shapes the migration. To shape the migration approach, first the To-Be state is defined within the forward engineering process; further, the To-Be state is compared with the As-Is and as such, the legacy elements are selected and re-shaped to services. For instance, in JWV migration starts with the To-Be business processes and business rules are defined. Those business processes are then compared with the ones already realized in the legacy systems. This way the legacy elements that fit the To-Be business processes are selected and migrated (see Figure 4.1.III(b)).

A question that arises is why industries perform migration in a bowl-shaped manner. Some of the participants, in one way or another, stated that in order to reach the migration goals they need to have the To-Be state as the primary shaping force behind migration. As such, we conclude that to ensure achieving the migration goals, companies shape their migration decisions primarily by the To-Be state.

**Contrast with theory.** Unlike the industrial migration approaches, the academic ones are mainly arc-shaped. In the SLR only 32% of the primary studies are categorized as bowl-shaped approaches and the rest are arc-shaped. As such, 68% of the approaches do not support To-Be driven migration, which is considered as the best practice among the practitioners. This highlights promising opportunities for research to focus on how to support To-Be driven migration.
CHAPTER 4. ON THE DIFFERENCES BETWEEN ACADEMIC AND INDUSTRIAL MIGRATION APPROACHES

4.3.3 Legacy Understanding through Personalization

Reflection Box.3.

- **F3.1.** The industrial migration approaches do not use reverse engineering techniques to understand the legacy systems.
- **F3.2.** The required knowledge is elicited from the stakeholders who own the knowledge.

Understanding the legacy systems plays an important role in SOA migration as it enhances extracting the best candidates among existing legacies for migration to SOA. In traditional software engineering, this understanding is gathered by extracting the representation of the legacy systems using reverse engineering techniques. As shown in Figure 4.1.11, we observed that in the industry-defined approaches none covers the reverse engineering process. This observation resulted in two key findings discussed in the following.

**Finding F3.1.** To gain the required understanding of the legacy system, the industrial approaches do not use reverse engineering techniques. This is due to the following two reasons:

a) the knowledge about the pre-existing system mainly resides in the stakeholders’ minds (e.g. maintainer, developer, and architect). As such, the stakeholders know what functionalities are supported, and where they are located in the legacy system. As a result, reverse engineering of the pre-existing system is not favorable considering the little Return On Investment (ROI) it brings.

b) the legacy assets are usually comprised of a set of heterogeneous systems that are implemented in different programming languages ranging from COBOL to Java. As a result, for reverse engineering of the code different tools are needed and this implies a considerable amount of costs.

**Contrast with theory.** To understand the legacy assets, more than 60% of the approaches in the SLR use reverse engineering techniques. Those approaches extract the representations of the legacy assets using techniques such as code analysis and architectural recovery. Only one of the academic approaches ([Lewis and Smith 2007]), supports the legacy understanding without reverse engineering techniques (i.e. using structured interviews). This indicates an important gap between theory and practice since reverse engineering is not adopted in industrial SOA migration. This highlights two promising research opportunities (i) identifying causes of such lack of adoption (e.g. research is not mature enough) as well as prerequisites and types of reverse engineering approaches that fit industrial needs, and (ii) defining approaches for legacy understanding without reverse engineering.

**Finding F3.2.** We further observed that the industrial migration approaches elicit the relevant knowledge by directly asking the stakeholders, who own, de-
4.3. RESULTS

dveloped, or maintained that system. More precisely, knowledge about the legacy assets mainly remains tacit in stakeholders minds. As such, understanding is achieved by person-to-person knowledge elicitation. We argue that, this type of knowledge elicitation is in-line with personalization knowledge management strategy [Hansen et al., 1999]. As noted in Chapter 1, personalization deals with exchanging tacit type of knowledge. Using personalization, the legacy understanding is gained by knowing ‘who knows what’ and consequently sharing the tacit knowledge about the legacy assets in that regard. For instance, the knowledge about where a functionality is located in the legacy assets (e.g. account management functionality) is elicited from the corresponding domain architect (e.g. accounts domain architect), who generally knows where in the legacy assets the domain-related functionalities are realized. To be noted, besides people owning the knowledge, industrial migration approaches use other sources of the knowledge if available. For instance, to identify the pre-existing architecture of the system they mainly ask the architects or maintainers, review the existing documentation, examine the legacy system interfaces or finally read the code.

Contrast with theory. In the SLR, all the approaches focus on capturing the knowledge by documenting it. As such, they are in-line with codification strategy addressing explicit documentation of the knowledge. The results of this study, however, suggests the importance of personalization. As such, research is needed to improve elicitation techniques, especially targeted for SOA migration, supporting personalization strategy.

4.3.4 Service Extraction by Defining the Ideal Services

Reflection Box 4.

- **F4.1.** The main driver in extraction of the legacy assets for migration is the representation of ideal service.
- **F4.2.** Approaches emerged out of more experience portray the ideal services in more detail.

**Finding F4.1.** The migration approaches, inherently, embrace trade-off analysis between the level of reuse of legacy elements and characteristics of the ideal services. We observed that, in this trade-off analysis, the industrial approaches assign considerably higher weight to the later rather than the former. To do so, first they determine the ideal services during the forward engineering process.

Later, those ideal services are re-shaped in a way that the reuse of pre-existing assets are realized. This way, the representation of the ideal service is the main

---

2With the term ideal service we mean a service that represents a repeatable (business) activity, has a specified outcome, is loosely coupled, is self-contained (state-less and adhering to a service contract), and yields distributed ownership.
CHAPTER 4. ON THE DIFFERENCES BETWEEN ACADEMIC AND INDUSTRIAL MIGRATION APPROACHES

driver of service extraction. That is, the services identified from the pre-existing capabilities would likely be substantially different in the absence of that representation of the ideal service. This is in-line with our other finding that all the migration approaches are bowl-shaped meaning that the To-Be candidate services guide the analysis and transformation of the As-Is legacy elements.

Contrast with theory. A characteristic of the bowl-shaped approaches is having the ideal services (To-Be state) as the main driver in service extraction. As such, this finding points out the same gap between theory and practice as discussed in finding F2, namely inadequate support of To-Be driven migration.

Finding F4.2. We further observed that, industrial approaches vary in the level of detail in which they portray their ideal services. Some of the approaches only define the capability of the desired services at conceptual level (e.g. order business service), while some others also provide the design of such services along with its associated service contract (e.g. order software service design). Some of the approaches externalize the constraints which each service should meet, while some others do not explicitly consider any constraints. Interestingly, we observed that the companies with more experience in providing service-based solutions tend to define the ideal services more detailed compared to the ones with less experience. Hence, we argue that the extent to which the ideal service is codified is an indicator of the maturity of the migration approach.

Contrast with theory. Detailed description of the ideal services is a best practice that companies have developed with experience. Interestingly, we could not trace back this best practice to the academic approaches.

4.4 Discussion

The theory and practice gap is a prominent and yet an unsolved problem in the software engineering field. The premier conference on software engineering featured in 2011 a panel on “What Industry Wants from Research” discussing the current gaps between theory and practice, and how to address them. All panel members in one way or another hinted the following cause of such gap: what research proposes does not fit the fundamental problems, goals, strategies and weaknesses of practice. We argue that, this chapter is a step towards filling the theory and practice gap as it sheds light on how migration is performed in practice and further contrasting it with how academic research addressed the migration problem. By identifying the characteristics which make these approaches favorable for practice, we could identify directions for future research that have better chance of adoption by practitioners.

I) Migration approaches fitting core activities. Getting back to finding F1, we argue that core activities can act as a frame of mind confining the migration approaches that are more aligned with practice. From that perspec-
tive, one would see that, for instance, the approaches addressing wrapping the applications as a whole are more in-line with practitioners concerns, compared to the ones addressing the automatic recovery of the legacy architecture. Hence, this frame of mind pinpoints the types of industry-relevant research in SOA migration methodologies and techniques.

II) To-Be driven migration approaches. As noted in finding F2, inadequate support for the bowl-shaped approaches in academia highlights promising opportunities for research to focus on how to support To-Be driven migration. For instance, future research can focus on addressing the following challenge of the practitioners: how to systematically elicit and capture the migration drivers and how to shape the migration approach using those drivers.

III) Legacy understanding without reverse engineering. Although reverse engineering is not covered in industrial migration approaches (see finding F3), elicitation of the knowledge about the legacy asset is crucial for a successful migration. In this regard, research can benefit practice by providing methods, techniques, or guidelines that facilitate elicitation of migration-relevant knowledge from different sources of such knowledge.

IV) Legacy evaluation from multiple perspectives. As noted, companies evaluate and extract the legacy assets for migration to SOA by depicting their ideal services. This is, however, done in an ad-hoc manner, which may hinder successful service extraction. An immediate concern calling for further research is how to systematically evaluate pre-existing legacy assets based on different aspects of the ideal services. The ability to define sound metrics for assessing pre-existing assets is essential to realize such systematic service extraction. While some work is being done in this respect (e.g. Reddy et al. (2009)), most metrics address the technical perspective (e.g. loose coupling, reusability) rather than the business perspective (e.g. agility, auditability, concept extendibility). Those business-related aspects though play a significant role in legacy asset extraction in practice.

4.4.1 Threats to Validity

Below, we discuss the validity threats of this work and how we addressed them. **Internal validity.** The interviews are mainly conducted by a single researcher and hence subjective interpretations might exist. To mitigate this threat, the interview guide was checked and validated by senior researchers experienced in software engineering, empirical studies and SOA. Moreover, two of the interviews were coached by professional expert in the field of ‘interviewing in qualitative research’, followed by two reflection sessions to review the execution of the interviews. Threat to validity of the analysis is in the general applicability of the
CHAPTER 4. ON THE DIFFERENCES BETWEEN ACADEMIC AND INDUSTRIAL MIGRATION APPROACHES

codes used for characterizing and classifying migration approaches. An assuring factor in this regard is that the start-list of codes is extracted from a conceptual framework published in a service-oriented computing forum, after being peer reviewed by experts in the field (Razavian and Lago, 2010). This framework stems from existing theory on reengineering and architectural recovery while it is is constantly refined through our coding procedure. This further consolidates its general applicability. The final threat considered is whether or not the academic approaches involved in our analysis adequately represent SOA migration approaches in the field. An assuring factor in this regard is the strength of SLRs as a research method in both minimizing the bias in the review and identifying all available research relevant to SOA migration.

External validity. Possible threat to validity is the generalizability of the findings. To mitigate this threat interviewees were chosen from international companies that are geographically distributed. In addition, we intentionally chose senior architects as they are the stakeholders who are aware of the key characteristics of the migration projects, have long-lasting experience in multiple projects and know the trends and current practice in use in the company. In order to cover different but relevant perspectives on the subject matter, we chose both enterprise and technical architects as interviewees.

4.5 Conclusions

This chapter explored the types of migration approaches employed by leading SOA solution providers in practice. Results show that by supporting similar set of activities, process organization, and best practices, industrial migration approaches mature towards a similar approach to SOA migration.

In spite of what academics think, practitioners still face difficulties in consolidating to a successful yet cost-effective migration approach. The many available approaches often prove to be abstract or commercial to be applicable. By contrasting the industrial migration approaches and the academic ones, this chapter emphasizes important gaps between theory and practice and consequently sketches the promising industry-relevant research directions. Those research directions enable finding solutions to problems that industrial practice confronts in real-world migration cases and is tailored to individual needs.
A Lean & Mean Approach for SOA Migration

This chapter builds upon our previous results of the interview survey presented in Chapter 4. The purpose of this chapter is twofold: 1) to discover the migration approaches that industrial practice adopts 2) to identify the benefits of making such approaches explicit. As a result, we generalize the practice of industrial migration into a Lean & Mean SOA migration approach. In addition, the uses of the approach pinpoint promising industry-relevant research directions.

5.1 Introduction

Chapter 4 showed that the industrial approaches are considerably different from the ones originated in academia. By discussing these differences with practitioners we were suggested that such differences pinpoint an undesired gap between theory and practice. To fill this gap, there is a need to better understand the fundamental aspects of industrial migration approaches. It is thus of significant interest to understand the commonalities that exist between different migration approaches and to develop a general model of those approaches. Such a model would provide us a common ground for developing new migration approaches better suited to specific problems and needs of industrial practice.

To this end, we further analyzed the interview survey introduced in the previous chapter. The interview survey was further followed by a panel of experts. Despite the diversity of participating enterprises, this study revealed that they all converged to the same, one, common SOA migration approach: all use similar activities, and similar knowledge input/output to carry out migration. This suggests that with experience enterprises mature toward a similar migration approach. From the common practices of industrial approaches, this chapter generalizes a core industrial migration approach.
CHAPTER 5. A LEAN & MEAN APPROACH FOR SOA MIGRATION

The panel of experts that followed the interview survey investigated the benefits of such core approach. The panel envisioned to use this approach as a general tool to guide and steer migration projects. We further elicited a list of extensions to such tool, that would address the recurring problems in industrial migration, namely identification of the costs and risks of migration projects, and deciding on the best migration approach to mitigate them. The core approach with extensions that emerges from the panel resembles the Lean & Mean approach of Kruchten (Kruchten, 2011): it is both mean (covering what really matters for migration) and lean (screening out details specific to the project at hand). The Lean & Mean migration approach, further, points to interesting directions for industry-relevant research.

5.2 Industrial Migration Approach

With the purpose of understanding the practice of industrial migration, we elicited the two views of knowledge and activity of the industrial migration approaches. Following the coding procedure presented in §3.3.3, for each interview we codified the migration approach used at its associated enterprise. Despite the many differences between the participating enterprises, the analysis revealed a great deal of similarity between their industrial migration approaches. The study, interestingly, shows that the approaches converge to, one, common approach. More precisely, industrial migration approaches perform similar activities, and use and produce similar knowledge elements. We see that this approach, which is working in practice, has emerged out of experience. In other words, with experience enterprises have matured toward a similar approach to SOA migration. In the reminder of this section we will describe the details of the analysis results with respect to the two views of activity and knowledge. Quotes from the interviews are included, too.

5.2.1 Core Activities

As noted in Chapter 4 by analyzing the migration activities that emerged from the interviews and the subsequent mappings we concluded the following: industrial approaches conceptually cover the same set of activities for migration. This implies that various companies, regardless of their company type (i.e., consultancy vs. in-house) and market segment (e.g. telecom, banking, energy), share the same set of migration activities. Here, we dig into the fundamentals of those activities as well as the “how” and “why” of each activity. These activities, called

---

1According to Oxford Dictionary, the term “lean and mean” refers to a person or an organization who is fit and ready for hard, efficient work.
Figure 5.1: Industrial Migration Approach

core activities, are shown in Figure 5.1. In the following we explain the core activities and discuss the related findings.

Migration Planning Traditionally, there are two generic categories of migration lifecycles: the ones addressing incremental migration (e.g. chicken little), and those with complete, sudden, migration (e.g. cold turkey) [Brodie and Stonebraker 1995]. It was, however, only the first category, incremental migration, that emerged from our study.

CXB (see Table 4.1) says: “... gradual migration is usually the only option in practical situations. This is because, to be competitive, some of the services need to be introduced to the market fast”.

Not surprisingly, migration in industrial approaches starts with lifecycle planning. The plan must reflect important decisions such as the number of increments, or the order by which existing assets are migrated. Interestingly we found that

2 The grey activities in Figure 5.1 illustrate the activities of the framework that are not covered in the industrial approaches.
to support these decisions, some of the surveyed companies have developed a number of local practices. For instance, we observed that in-house companies, who migrate their existing system to services, have developed practices such as highest-value-service-first or highest-change-first. Using these two practices, the first legacy assets migrated to services are those with highest value for the market or the ones that are prudent to change and consequently need the flexibility offered by SOA to cope with changes. The consultancy companies, on the other hand, devised practices such as easiest-service-first or selection-using-enterprise-architecture. In easiest-service-first practice the legacy assets that are easiest and consequently fastest to be ported, are first extracted and migrated to services. The selection-using-enterprise-architecture practice suggests forming the sequencing of migrating services using business and information architecture of the regarding domain. As such, they can select independent increments to migrate (i.e., portions of legacy elements that can be migrated independently of each other), the sequence of the increments to achieve the desired goal, and dealing with inter-dependencies among legacy elements. Although it was not evident from the interviews why the consultancies and in-house companies differed in their practices, we deduced the following hypotheses. In-house companies look for solutions that bring them direct value (such as retaining market position or keeping up with market changes) and that is why they have adopted practices highest-value-service-first or highest-change-first. In the case of consultancy companies, on the other hand, the main motivation is to meet the agreements with their customers. In the commercial setting of fixed price IT projects, the legacy assets to be migrated are selected by the customer with a fixed price. In this case, consultancies devise the increments that enable them to migrate the legacy assets within the fixed cost and time of the contract with the customer. This makes practices such as easiest-service-first more appealing. Moreover, enterprise architecture can provide a common ground for consultants, possibly new to the domain of the customer, to select independent legacy assets for migration.

Understanding As-Is and To-Be states Similar to any reengineering effort, industrial migration also achieves understanding of the As-Is (i.e legacy assets) and To-Be states (i.e services). Related to this activity we found two interesting observations discussed in the following:

(i) As noted in Chapter 4 to gain the required understanding of the legacy assets, none of the enterprises used reverse engineering techniques. The industrial migration approaches elicit the relevant knowledge by directly asking the stakeholders, who own, developed, or maintained that system. As such, understanding is achieved by person-to-person knowledge transfer.

PXB from an in-house company says: "we do not need to do reverse-engineering to understand the architecture or business processes of our systems. We still have
colleagues who were involved in creating the today's legacy systems, we just go and ask them. Of course some knowledge might be lost, but reverse-engineering is too costly”.

RXB from a consultancy company says: “we usually do not do reverse-engineering because we are consultancies and we cannot have tools for reverse-engineering of the different programming languages of all possible customers. We gain understanding about the legacy systems with practices similar to those of requirements engineering such as workshops and interviews with the customers”.

(ii) To gain understanding about the As-Is and To-Be states all enterprises extract the relevant portion of enterprise architecture (EA) of both legacy assets and target service-based systems. As such, the industrial approaches (partially) capture the As-Is and To-Be states in terms of EA elements, i.e. business architecture, data architecture and technology architecture. Business architecture represents the structural and behavioral architecture of the business. The first represents the key business capabilities and the interrelationships among them. Examples of business architecture are legacy functional blueprint (CXB), business service pool (RVS) and domain architecture (RXB). The business behavioral architecture represents the main behavior of the business domain in terms of business processes and business rules. Data architecture represents the data entities representing the business. Finally, technology architecture articulates the embodied software, middleware and hardware technologies used in As-Is and To-Be states.

**Gap analysis** Once the As-Is and To-Be states are understood, gap analysis at EA level is carried out. Gap analysis aims at understanding the gaps between the legacy assets and the target services.

PXB says: “This helps us to position our current state with our target business goals and IT investments plans”.

GWH says: “When we look for EA landscape of a customer, in effect, we can compare current customer's business, information and technical architecture. This enables us to align the goals of migration with the customer landscape. The important point is to scope and extract the portion of EA that is relevant for migration and not the overall EA”.

Once these gaps are identified then the architects decide about what to reuse from the legacy assets and what services has to be newly developed.

**Forward engineering** In general terms, forward engineering includes analysis, design and implementation of software services. In doing this, the architects move from the characteristics of To-Be services (e.g. requirements and goals) to a set of implemented software services. Forward engineering in the specific case of migration needs to embrace an additional factor, namely, trade-off analysis
between the degree of legacy leverage (i.e. As-Is state) and characteristics of the ideal services (i.e. To-Be state) (Tilley and Smith 1995). As an example, the trade-offs between reuse of the monolithic and large legacy components on one hand, and well-granular services on the other hand have to be frequently handled in forward engineering. When making compromises, one can give priority to To-Be state and as such the forward engineering is driven by characteristics of ideal services instead of reuse of legacy assets. We found that in all surveyed companies the priority is given to To-Be state. In this regard, some of the participants stated that in order to reach the migration goals they need to have the To-Be situation as the primary shaping force behind the migration.

CXB says: “We start the migration by defining the target blueprint (instead of identifying what are the pre-existing capabilities), otherwise we cannot ensure achieving the level of flexibility we envision”.

This is inline with the finding presented in Chapter 4, namely, migration approaches are To-Be driven (i.e. bowl-shaped). We further found that in all surveyed companies identification of ideal services is carried out prior to deciding what software services to be migrated or newly made. Having identified the ideal services, the next step is to extract the software service model, i.e., the actual services. When designing the actual services the ideal services are re-shaped in a way that the reuse of pre-existing assets are realized.

Transforming legacy assets to services All elicited migration approaches include transforming the pre-existing applications as-a-whole to new target services (i.e. Application Wrapping activity in Figure 5.1.I). Transformation here entails wrapping the legacy assets without decomposing them. We observed that the wrapping fulfills one of the following two purposes: a) integrating the legacy assets, or b) realizing a (set of) conceptual services identified during the forward engineering process. In addition to transforming the applications as-a-whole, in some cases the legacy assets are decomposed and segments of them (e.g. components or modules) are wrapped and transformed to services (i.e. Legacy Segment Wrapping activity in Figure 5.1.I). We found the three following purposes for this type of transformation: (i) porting the legacy elements that realize a specific functionality as a service, (ii) modifying segments of the legacy assets to support the changes to requirements (both functional and non-functional), and (iii) eliminating segments of legacy assets that are going to be replaced by COTS packages or external services.

5.2.2 Core Knowledge

By following step 2 of the coding procedure, for each industrial approach we identified (i) the knowledge elements that are used as input, and (ii) the conversions
among those elements. The categories of knowledge emerged from the coding procedure revealed that there is a set of knowledge elements that are prevalent in any SOA migration endeavor. Those knowledge elements are shown in Figure 5.2.II. The figure shows that industrial migration approaches share three main knowledge outputs, and also use similar input knowledge elements for producing each output. These three main knowledge outputs are: (i) migration plan, (ii) ideal services, and (iii) software services. For each of them we observed the following commonalities:

Knowledge to create migration plan  As indicated in Figure 5.2.II we found the following generic input knowledge used for defining the migration plan: business goals, risks, costs/investments and constraints. Not surprisingly, the decisions related to migration increments are made in-line with business goals (e.g., achieving flexibility to react to requirements change), costs (i.e., development and operation cost of services) and already made investments (e.g. in creating the legacy assets). In addition, when deciding on the migration plan all enterprises took into account, in one way or another, organizational/technical constraints such as using a specific platform (due to e.g., a business partnership with a vendor) or operational risks (e.g., available staff for operating the services).

Knowledge to create ideal services  To define ideal services, practitioners use as input the business capabilities offered by legacy assets and the desired capabilities in the service-based application. These capabilities are mainly captured using business processes or business services. Data architecture is also considerably used in identification of the business services addressing the functionality of business data entities. The business capabilities and business data entities are generalized to As-Is and To-Be enterprise architecture. Besides the As-Is and To-Be EA, to identify the ideal services, practitioners use as inputs new requirements, and service-specific characteristics (e.g. granularity).

Knowledge to create software services  The representation of the ideal services as well as the knowledge about the design of legacy assets are the main knowledge inputs for service extraction. This is shown in Figure 5.2.II where the input knowledge elements are representation of ideal services as well as design-and code-related knowledge of the legacy assets (i.e. design models, design constraints, qualities, and the code itself).
5.3 A Lean & Mean Migration Approach

To represent and steer migration, the industrial survey identifies a general tool, common to all seven participating companies, focusing on two core aspects: (i) core activities reflects what needs to be done in SOA migration and (ii) core knowledge highlights the types of knowledge that shape and drive the migration. Whereas this result clearly helps giving structure to a migration project, we wanted to understand if formalizing it would help industry in solving crucial problems. In other words, would the industrial migration approach provide concrete benefits? And what benefits?

To further dig into this question, we organized a panel of experts. We invited four senior architects from three companies different from the seven participating in the industrial survey. We intentionally chose senior architects as they are the stakeholders who are aware of the key characteristics of the migration projects, have long-lasting experience in multiple projects and know the trends and current practice in use in the company. As a co-product of the panel we gathered further evidence on the commonality of the industrial migration approach. Most importantly we elicited a list of concrete benefits that the core activities and the core knowledge could offer for their own company.

The benefits are discussed in the following. Overall, we observed that the way the panel of experts (or the panel in short) sees the industrial migration approach resembles the idea of a lean & mean process model [Kruchten (2011)]. In his paper, Kruchten observed that the process models developed in the last decades are too rich, hence hindering process support rather than providing guidance. He argues that more meaningful (i.e. mean) and much simpler (i.e. lean) models would be better and wider applicable, and they could be extended and customized only in case of need. In a similar vein, our industrial migration approach is both mean (covering what really matters for migration) and lean (screening out details specific to the project at hand). Interestingly enough, the benefits emerged during the panel discussion identify a whole set of extensions to the core that would eventually offer customizations reusable by companies only if they need them. For each of the core activities and core knowledge the following discusses the extensions elicited from the panel.

5.3.1 How Core Activities Benefit Practice

Core activities as a reference model. With regards to core activities, the panel found them conforming with their “way-of-working” in their migration projects. What was indicated as most useful was to set the core activities as the reference model of their migration process and further derive their process model using this reference. Furthermore, this reference model was indicated to be
5.3. A LEAN & MEAN MIGRATION APPROACH

helpful in describing the essentials of their existing and future types of migration so that they can be compared, evaluated, and tailored.

**Extensions.** The experts emphasized the importance of making explicit various concerns about the activities the most important being:

1. *Costs and risks of activities.* The panel emphasized that what is especially important for them is to explicitly know the costs and risks of each migration activity. Associating costs and risks to core activities makes the core an even more powerful tool for planning how to do migration. For instance, by knowing the costs of automatic reverse engineering, one might decide to gain understanding about the legacy asset using alternative techniques (e.g. asking stakeholders).

2. *Practices related to different activities.* The panel also suggested to link the core activities to practices. This helps practitioners to select the practices that are emerged out of experience and are proved to be beneficial for carrying out specific activities. This will result in saving considerable amount of time and costs. In summary, this extension would help to bring order to existing practices and further facilitating their selection, and reuse in similar situations. Chapter 6 develops this extension.

5.3.2 How Core Knowledge Benefits Practice

**Core knowledge as a checklist.** By specifying the main input and output knowledge elements, the core knowledge acts in practice as a checklist. The panel stated that the most important use of this checklist is not to identify what knowledge is available, but to focus on what is not available. In this way, practitioners can analyze early enough the costs and risks of eliciting the missing knowledge and eventually decide whether it is cost-effective to do it or not.

**Extensions.** The panel also identified various extensions to the core that provides benefits to migration. In the following, we describe the most important of those extensions and their related benefits.

1. *What is the source of knowledge: documents or people?* The panel found that the core knowledge should make the sources of knowledge explicit. The problem is that most of the times knowledge is not written in documents, but is kept in people's mind. A major contribution of core knowledge in this case is to highlight where a knowledge resides (documents or people's mind). Knowing this is essential for planning the right activities for knowledge elicitation. In this way, again, practitioners can better analyze and manage the costs and risks of knowledge elicitation activities. It should be noted that, the distinction between knowledge in documents and knowledge in
people’s mind confirms Nonaka and Takeuchi’s two modes of knowledge: explicit and tacit [Nonaka and Takeuchi (1995)].

2. **Validity period for knowledge elements.** In addition to the previous point, the panel observed that the core knowledge should make explicit if the availability of some knowledge elements is temporal, i.e. if it has an *expiration time*. Quite often migration projects last a long period of time, and people that participate in the beginning of the project leave the company or retire before it is finished. As people are typically important sources of knowledge, and knowing if they will become unavailable before project completion is essential to ensure knowledge transfer in time, hence avoiding delays and economic losses.

3. **What can change.** One interesting observation made by panel was that in the lifetime of a project some knowledge may change. The issue is that in some cases changes are frequent and have important impacts on migration. As such, if not planned those changes can result in extra costs and efforts. The panel commented that the core knowledge should be an instrument to identify and highlight which knowledge elements might undergo changes. In this way, practitioners can analyze early enough what can change and how to mitigate the related risks.

### 5.3.3 Specification of the Lean & Mean Migration Approach

So far, we have introduced the basic idea behind the Lean & Mean migration approach, i.e. separating the context-generic elements of migration from the context-specific ones. Furthermore, the benefits emerged during the panel identified a whole set of extensions to the core. This section presents the specification for the Lean & Mean migration approach.

The Lean & Mean approach collects the context-generic elements into *core* and context-specific elements into *extensions*. The metamodel presented in Figure 5.2 represents the constructs of the Lean & Mean approach and their relationships. In the following, we further explain these constructs in the core and extensions. In *italic* we point out to those constructs.

**Core** The core includes the essential elements that are common in every migration approach. Core is lightweight and simple and is represented in two views: *Activity View* and *Knowledge View*. Figure 5.3 represents the Two-View representation of the core for migration to services. Knowledge view represents a set of stripped-down *Knowledge Elements* (i.e. the 15 knowledge elements shown in Figure 5.3) that are input/output to the migration approach as well as the *Knowledge Conversions* among them. Knowledge view can be customized for each migration
5.3. A LEAN & MEAN MIGRATION APPROACH

Figure 5.2: Metamodel of the Lean & Mean Migration Approach

project. This customization includes specification of the knowledge elements or addition of some more input/output knowledge elements. Chapter 6 provides an example of the customization of knowledge view. Activity View encompasses a set of generic Activities representing what needs to be done in a migration project. These activities mainly transform a number of input Knowledge Elements. Table 5.1 presents an overview of the core activities and their input/output knowledge elements. Similar to knowledge view, activity view can be customized for each migration case. Such customization entails specifying, adding or removing activities based on needs or context of the migration case (see Chapter 6 for examples).

Extension To support the more advanced needs of a certain migration project core should be supported with extensions. Extensions, represented in terms of Extension Views, are tied to Core Views and as such enrich that view. The extensions identified in §5.3.1 all address more advanced needs of migration projects and can be supported by extending knowledge or activity views. For instance, to
deal with changes incurring during migration, an extension to knowledge view is proposed in Chapter 7. This extension, represented in a change view, enriches the knowledge view with the concepts that are essential for identifying the impact of change.

## 5.4 Threats to Validity

Below, we discuss the validity threats of this work and how we addressed them.

**Internal validity.** Internal validity aims at ensuring that the collected data enables the researchers to draw valid conclusions (Creswell 2003). In this survey, the interviews are mainly conducted by a single researcher and hence subjective interpretations might exist. To mitigate this threat, the interview guide was checked and validated by senior researchers experienced in software engineering, empirical studies and SOA. Moreover, the first two interviews were coached by a professional consultant expert in the field of ‘interviewing in qualitative research’,
### 5.4. THREATS TO VALIDITY

#### Table 5.1: Overview of Lean & Mean Migration Activities and Input/Output Knowledge Elements

<table>
<thead>
<tr>
<th>Activity</th>
<th>Input Knowledge</th>
<th>Output Knowledge</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Migration Planning</td>
<td>Business Goals, Risks, Migration Plan Cost/Investments, Constraints</td>
<td></td>
<td>Within this activity different aspects of migration are planned, such as increments, and legacy elements to be migrated.</td>
</tr>
<tr>
<td>Understanding As-Is and To-Be States</td>
<td>As-Is New Requirements, As-Is and To-Be Enterprise Architecture, Technology, As-Is and To-Be Functionalities</td>
<td>Data Architecture, As-Is and To-Be Enterprise Architecture</td>
<td>This activity achieves understanding of the As-Is (i.e legacy assets) and To-Be states (i.e services).</td>
</tr>
<tr>
<td>Gap Analysis</td>
<td>As-Is Enterprise Architecture, To-Requirement Gaps, Functionality Gaps, Data Gaps</td>
<td></td>
<td>Gap analysis aims at understanding the gaps between the legacy assets and the target services.</td>
</tr>
<tr>
<td>Service Analysis</td>
<td>New Requirements, As-Is and To-Ideal Services</td>
<td></td>
<td>During this activity candidate services are identified from business processes and business services.</td>
</tr>
<tr>
<td>Service Design</td>
<td>Ideal Services</td>
<td>Software Service</td>
<td>Service design incorporates design of software services and service compositions.</td>
</tr>
<tr>
<td>Service Implementation</td>
<td>Software Service</td>
<td>Service Implementation</td>
<td>During service implementation the software services are realized.</td>
</tr>
<tr>
<td>Design Element</td>
<td>As-Is Quality Aspect, As-Is Design Software Service</td>
<td>Service Implementation</td>
<td>Transforming the segments of the legacy system to services.</td>
</tr>
<tr>
<td>Transformation</td>
<td>Constraints, As-Is Design Models</td>
<td></td>
<td>This activity transforms pre-existing applications as a whole to new target services. Transformation here, entails wrapping the legacy system without decomposing the existing system.</td>
</tr>
<tr>
<td>Code Transformation</td>
<td>As-Is Code</td>
<td>Service Implementation</td>
<td></td>
</tr>
</tbody>
</table>
followed by two reflection sessions to review the execution of the interviews. Threat to validity of the analysis is in the general applicability of the codes used for characterizing and classifying migration approaches. An assuring factor in this regard is that the start-list of codes is extracted from a conceptual framework published in a service-oriented computing forum, after being peer reviewed by experts in the field (Razavian and Lago, 2010). This framework stems from existing theory on reengineering and architectural recovery while it is constantly refined through our coding procedure. This further consolidates its general applicability. An additional threat to validity of analysis relates to generic codes that hide important differences between the approaches. An assuring factor in this regard is that the same set of codes was used in the systematic literature review (Chapter 3) in which very different categories of migration were identified. An additional threat is that the analysis is performed by a single researcher. Nevertheless, bias and omission issues is (partially) mitigated by having the codification results double-checked by both a second researcher and the interviewees. Finally to assure accuracy of findings, the industrial migration approach and initial findings were iterated with the interviewees to be confirmed.

**External validity.** External validity defines to what extent findings from the study can be generalized (Creswell, 2003). In this regard, a possible threat is that the survey is relatively small and the companies are mainly situated in the Netherlands and Belgium. As such, the architecture culture of Dutch enterprises might have influenced the results to be “architecture centric”. To mitigate this threat, the interviewees were chosen from international companies that are geographically distributed. Nevertheless, as a follow-up study we plan to geographically extend the survey to further investigate the generalizability of results. In addition, to increase generalizability we intentionally involved both in-house and consultancy companies. Moreover, we chose senior architects as they are the stakeholders who are aware of the key characteristics of the migration projects, have long-lasting experience in multiple projects and know the trends and current practice in use in the company. In order to cover different but relevant perspectives on the subject matter, we chose both enterprise and IT architects as interviewees. Finally, after completion of the interview survey we organized a panel with experts other than the ones participated in our study. This further consolidates the general applicability of the results. In addition, the results have been presented in a tutorial held at ICSE 2012 where industrial participants further confirmed the results in their own practice.

### 5.5 Conclusions

This chapter presents the Lean & Mean migration approach to be used as a tool to steer and drive migration. In any migration approach there exist elements
that are common for all migration efforts, independent from the context or the project at hand. The Lean & Mean approach separates those context-generic, stripped-down elements into the core, whilst it supports addition of extensions based on need. The core and extension, respectively, are defined based on the results of the interview survey followed by the panel.

The results of the survey show that despite the diversity of enterprises participating in the study and of their market position, their migration converges to one, common, approach - using core activities, driven by core types of knowledge elements, fundamentally To-Be driven, and with little to no attention to reverse engineering. Further, the panel envisioned a number of reusable extensions to each of the core knowledge and core activities.

Chapter 6 applies the Lean & Mean approach and evaluates in two industrial studies. In addition, two extensions to the core activities and the core knowledge are defined in Chapter 6 and 7, respectively. The extension to core activities binds practices to the core activities. The extension to the core knowledge complements the core knowledge with a change viewpoint to address dealing with changes in knowledge elements. These extensions and the way in which they guide migration decision making is discussed in the following chapters.
An Industrial Study on Lean & Mean Migration

Like today’s software developments, migration projects are faced with steadily increasing demands for efficiency: migration has to be carried out faster, better and cheaper. At the same time, migration complexity increases and migration projects remain challenging, despite the companies’ extensive experience.

To cope with such efficiency demands, companies need a well-defined migration approach. In Chapter 5, we have defined such a migration approach, i.e., Lean & Mean approach. In this chapter, following technical action research, we apply our Lean & Mean approach, and evaluate it, in two large industrial case studies in the field of data migration. Furthermore, to support practice reuse an extension to the activity view is defined. The use of the Lean & Mean in our case studies reveals the benefits of the Lean & Mean approach for meeting the efficiency demands.

6.1 Introduction

Many companies migrate their legacy assets to modern paradigms such as Service Orientation. This happens frequently as a result of mergers and reorganizations. Think of migration examples like the ABN-Amro/Fortis banks, KLM/Air France, or Océ/Canon mergers, involving thousands of applications in their IT portfolio, and the data of millions of customers. These migrations typically have to deal with many challenging problems, such as data model inconsistency, functional overlap, and architectural and platform inconsistency. Moreover, in carrying out migration projects, companies have to address various demands for high quality, fast delivery, and decreasing costs. Hence, migration projects remain challenging, despite the companies’ extensive experience.

Logica, the consultancy company involved in this study, had the problem
of achieving efficiency in their migration projects. One of the main roots of this problem, emphasized by managers of the business migration sector, was inadequate reuse of past experiences. In this regard, a manager said: “essentially there are many similarities in migration projects, but usually when a new project arrives we start all over again. We need to avoid repeating what we already know”. As noted by the managers, reusing the existing assets can reduce time and cost while ensuring proven quality. This is, however, not easily achieved in the presence of many different complexities such as incompatibility between data models that lead to inefficiency.

This chapter addresses the efficiency problem of Logica by applying a migration approach that has two features: (i) reusing past experience, and (ii) employing a light-weight migration approach. Our first solution for achieving efficiency is to reuse past experience. Consultancy companies are faced with similar migration projects and therefore have vast experience in migration. As shown by various works (Basili et al., 1994; Johnson et al., 2012), reusing experience in the form of activities or knowledge is essential for efficient and cost-effective migration. To be able to reuse experience, it needs to be explicitly captured. Capturing the experience can be achieved by a general model for migration approaches that enables reuse by identifying the fundamental commonalities between migration projects.

Our second solution to address demands for efficiency relates to light-weight migration. Light-weight software development approaches have been adopted by many companies over the past decade (Ebert et al., 2012). Although some claim that without a complex, concise and rich software process efficiency cannot be achieved (Jacobson et al., 1999; Boehm, 1986; Rose, 1992), many studies have shown that light-weight approaches are in-line with the team’s way-of-working, focus on the most important tasks, and lead to efficiency (Johnson et al., 2012; Nord et al., 2012; Kruchten, 2011).

In Chapter 5 we defined the Lean & Mean migration approach. Inspired by having the both above features i.e., (i) supporting reuse of the common practice, and (ii) being light-weight, this work proposes such approach as a solution for the efficiency problem of Logica. The basic idea behind the Lean & Mean approach is to separate the fundamental knowledge and activities common among any migration project from the project-specific ones. Following this idea, we develop an empirically grounded Lean & Mean approach and evaluated it in two case studies in the field of data migration. Data migration, according to Gartner (Friedman, 2009), is the most important problem in legacy migration, while failed data migration efforts are common. By acknowledging that most migration projects will require migration of data, managers of Logica proposed data migration as the focus of this study.

In this study, we serve two goals: (i) Logica’s goal to improve their efficiency
6.2. INDUSTRIAL CASE STUDIES

and (ii) our research goal to improve the relevance of our Lean & Mean approach by trying it out in two large industrial case studies. To achieve these goals, we follow technical action research (TAR) \cite{Wieringa and Moral} 2012. Related to our first goal, we develop the Lean & Mean approach for data migration \cite{6.5}. In addition, the uses of Lean & Mean approach reveal the following benefits each related to efficiency: (i) increased reuse (ii) improved problem-solving (iii) capturing and supporting existing-way-of-working \cite{6.6}. With respect to our second goal, the industrial perspective of this study brings insight about the generalizability of our Lean & Mean approach \cite{6.7}. Moreover, interesting observations and lessons learned further contribute to the novelty of our approach \cite{6.9}.

This chapter is organized as follows. We introduce the notion of Lean & Mean approaches in § 6.3. § 6.4 introduces study design. In § 6.5 we present the Lean & Mean approach for data migration. Then § 6.6 explains how such an approach facilitates efficiency. § 6.7 the generalizability of such an approach. In § 6.8 we discuss threats to validity. § 6.9 presents our additional observations followed by conclusion of our chapter.

6.2 Industrial Case Studies

The case studies in this work are in the area of data migration. Data migration is an important and usually overlooked aspect of migration to services. According to Gartner \cite{Friedman} 2009, failure in data migration is an important problem in migration efforts, including SOA migration. In other words, in order to have a successful SOA migration, migration approaches need to specifically care for migration of data (i.e. legacy asset), which existing legacy systems and/or To-Be services work with.

Data migration is an integral part of migration to services and plays a key role in many migration projects. This is exemplified in the following three scenarios, which are emerged out of our collaborations with industrial practice. First, in many migration projects one of the goals is to allow new service chains which include consuming services provided by other parties. This usually requires integration of data with the service providers. Second, when enterprises want to improve their business process and integrate multiple, redundant, and fragmented systems, they need to consolidate the associated data structures. Third, usually legacy systems do not have a cohesive design and for example the business rules are mixed with data in the data bases. Thus there is a need for separating business rules from data. In all these scenarios the migration approach needs to support understanding, transforming and moving the data to the To-Be services. In this chapter we focus on this aspect of SOA migration and apply the Lean & Mean migration approach to our data migration case studies.
6.2.1 Case Descriptions

This work includes two case studies, referred to as Case I and Case II. Case I was a post-mortem analysis of a data migration project, whereas, Case II was studied while it was still active. In the following, we briefly introduce the two case studies. More details about these two cases and the adopted migration approaches follow in the remainder of this chapter.

Our first case study (Case I) was a successfully completed data migration project addressing the legal merger of two large banks. These two banks were merged with the main goal of creating synergies. Logica together with another consultancy company was in charge of the data migration of their nearly 1.6 million retail banking customers. The key distinguishing factors of this project included: high volume of data, complex interdependencies between the target applications, and strict time constraints (e.g. data had to be loaded only in weekends).

The second case study (Case II) was an ongoing data migration project of Logica. Similar to Case I, data migration was due to merger of the private banking sections of two banks. In this project Logica was in charge of full migration of more than 140,000 private banking customers’ data. The key distinguishing factors of this project included: strict business constraints (e.g. trust and ownership), flexibility in data conversion, and high level of business involvement. With regards to strict business constraints, for instance, only the product manager dedicated to each customer was allowed to access customer’s data. This made the role of product managers, as trusted parties, important in negotiations required for data mappings. Related to flexibility in data conversion, the migration needed to embed flexibility offered in private banking products in the data mappings and data conversions too.

6.3 Lean & Mean Migration Approach: A Treatment For Efficiency

With the purpose of increasing efficiency in mind, we proposed Logica the idea of capturing past experience in a reference model, which would indicate the fundamental elements repeating in every migration project. This idea led us to apply the notion of Lean & Mean migration approach to Logica. Chapter 5 explained the basic concepts of the Lean & Mean approach, shown in Figure 5.3 hereafter referred to as Lean & Mean-G (where “G” stands for “Generic”). Here, we introduce our proposed treatment for Logica that is based on the Lean & Mean approach.
6.4. STUDY DESIGN

Treatment Our proposed treatment for Logica constituted (i) identifying the core and the extension(s) specific to Logica’s data migration approaches (DM) and organizing it into a Lean & Mean approach (referred hereafter as Lean & Mean-DM) (ii) for new projects define their migration approach based on the Lean & Mean-DM. Our hypothesis was that having the Lean & Mean-DM at hand, definition of new migration can be carried out more efficiently.

6.4 Study Design

In order to define and apply the Lean & Mean approaches for Logica, we followed a technical action research approach (TAR), introduced by Wieringa and Morali (2012). Our primary goal for choosing TAR is in what they call practice goal in TAR, namely, to collaboratively improve a real situation, and to learn from it. In the context of this work, the situation refers to efficiency problem in migration projects of Logica. In addition to our primary goal, we had a secondary one that complies with research goal in TAR. The research goal in TAR relates to increasing the relevance of research artifact (e.g. Lean & Mean migration approach in this study) by trying it out in real situations (e.g. migration projects of Logica).

In this work we desired to investigate whether the Lean & Mean-DM is generalizable by applying it to migration projects of Logica. To pursue the two goals, we had to play two distinct roles: (a) designer, design a Lean & Mean approach for improving problem of Logica, and (b) researcher, draw lessons learned about the Lean & Mean approach.

By distinguishing between the practice and research goals as well as designer and researcher roles, TAR provides rigor in addressing both goals simultaneously. For each of the research and practice goals a separate action research cycle is dedicated, namely research cycle and engineering cycle respectively. TAR is a nested, cyclical process in which each cycle could be of type research or engineering cycle. The nested structure of TAR means that there is a cycle within a cycle. This is evident from Figure 6.1 where for instance there is a research cycle within the first engineering cycle.

The engineering cycles, addressing practice goal, include three phases: 1) problem investigation, 2) treatment design, and 3) treatment execution and evaluation. Problem investigation refers to identification of the primary problems. Treatment design refers to designing improvement solutions as treatments for the identified problems. Treatment execution and evaluation first applies the treatment and then evaluates its application. The research cycles entail the typical phases for research design including: 1) research design investigation, 2) validate research design, 3) execute research and 4) analysis.

TAR has been used in our industrial study as illustrated in Figure 6.1. As shown in this figure we carried out three engineering cycles and one research cycle.
Table 6.1 provides an anonymous profile of the participants of our study. In the following, we summarize the goals and the main results of each of these cycles. More details about the study design of the cycles can be found in Appendix C.

We carried out Engineering Cycle 1 aiming at, firstly understanding the context and needs of Logica, and secondly deciding on the treatment plan. These two goals were achieved following phase 1: problem investigation and phase 2: treatment design, respectively. In short, the treatment plan constituted first defining the Lean & Mean-DM based on a past project of Logica (i.e. Case I) then applying it in a new project (i.e. Case II). Phase 3: treatment execution and evaluation, initiated the research cycle.

The Research Cycle was carried out with the goal of trying out our past research artifact, Lean & Mean-G in real situations and learn about its uses and its generalizability. Accordingly, this cycle addresses the following two research questions:

- (RQ-I) What would be the effect of having the Lean & Mean migration approach in place?
- (RQ-II) Is the Lean & Mean migration approach generalizable\(^1\)?

To answer the research questions, in phase 1: research design investigation we designed our research (for details see Appendix). In phase 2: validate research design we identified the threats to validity of this study and took actions to mitigate them (see §6.8). Phase 3: research execution initiated two engineering cycles each conducting a different case study (i.e. Case I and Case II). After completion of the last two engineering cycles in phase 4: analysis of the results was carried out. The regarding details such as unite of study and data analysis method are represented in Appendix.

The main goal of Engineering Cycle 2 was to elicit the migration approach of Case I and organize it into a Lean & Mean approach. This cycle resulted in the first version of the customized Lean & Mean-G for data migration, so called Lean & Mean-DM. We constructed the core by: (i) mapping the migration approach of Case I on the Lean & Mean-G using the coding procedure presented in Appendix C and (ii) refining the mapping based on the stakeholders feedbacks received in feedback sessions. We constructed an extension to the core based on the feedbacks we received from the stakeholders as well as by analyzing the project documents. Lean & Mean-DM was further refined in the next cycle.

In Engineering Cycle 3, we aimed at identifying the benefits of the Lean & Mean-DM when determining the approach of the new project, i.e., Case II. To this end, we organized a workshop with the team members. Throughout this workshop

---

\(^1\)By generalizability we mean if the core knowledge and core activities are common to every migration approach.
we observed the ways in which the Lean & Mean-DM helped stakeholders in determining the Case II migration approach. The observed benefits were further discussed with the stakeholders in the feedback sessions.

![Research Method Diagram](image)

**Figure 6.1: Research Method**

**Table 6.1: Technical Action Research Participants**

<table>
<thead>
<tr>
<th>Participant</th>
<th>Role</th>
<th>Experience (years)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LNR</td>
<td>Sector Manager</td>
<td>12</td>
</tr>
<tr>
<td>PTW</td>
<td>Chief Architect</td>
<td>20</td>
</tr>
<tr>
<td>WDE</td>
<td>Business Analyst</td>
<td>8</td>
</tr>
<tr>
<td>MAF</td>
<td>Architect</td>
<td>13</td>
</tr>
<tr>
<td>MAM</td>
<td>Architect</td>
<td>13</td>
</tr>
<tr>
<td>MTF</td>
<td>Project Manager</td>
<td>5</td>
</tr>
<tr>
<td>MTM</td>
<td>Project Manager</td>
<td>5</td>
</tr>
</tbody>
</table>

### 6.5 Lean & Mean Approach for Data Migration

The Lean & Mean-DM was constructed with the data gathered from the last two cycles of this study. Altogether there were 208 minutes of feedback sessions and workshops and around 100 pages of project documents. The Lean & Mean-DM captures the kinds of activities that are fundamental in data migration (in
CHAPTER 6. AN INDUSTRIAL STUDY ON LEAN & MEAN MIGRATION

Figure [6.2 II] and the core knowledge elements to be made available in data migration projects (in Figure [6.3]). In addition, the Lean & Mean-DM also provides an extension to the core that binds domain-specific practices to the core activities (see Figure [6.5]). The practices were identified and codified through close collaboration with stakeholders. In the following, we explain the core and the extension.

6.5.1 Core of Lean & Mean-DM

This section describes the core and its organization in two views of activity and knowledge.

Core Activities

Figure [6.2 II] shows the core activities that we found being carried out in data migration endeavors. These are the activities that are not only shared in both Case I and Case II, but are also perceived by practitioners being fundamental in data migration approaches. While the core activities of Lean & Mean-DM are grounded on our two case studies, they are constructed by being mapped on the core activities of Lean & Mean-G. In other words, core activities of Lean & Mean-DM are specialization of the core activities of Lean & Mean-G. This is evident from Figure [6.2 II] where some of the activities of Lean & Mean-G (see reffig:Lean-Mean-CoreActivity. I) are repeated in Lean & Mean-DM (e.g. Migration Planning or Gap Analysis in Figure [6.2 II]) while some are specialized for data migration (e.g. Data Model Transformation, instead of Legacy Element Transformation). In the following, we briefly introduce the core activities of Lean & Mean-DM and our related findings.

- **Migration planning.** Migration typically starts with planning. RNL said, “we start planning the migration during our bidding process and at this stage we need to take a lot of important decisions about how to address the customers requirements while remaining in time and budget limits”. We found that the plan reflects decisions such as the number of increments, the migration timeline, toolsets, level of automation, and the level of business user involvement. Further in the project the plan helps the migration team to monitor the progress against the plan.

- **Understanding the As-Is and To-Be states.** Obtaining the just-enough understanding of the As-Is and To-Be states was indicated as one of the key challenges of migration projects. To do this, Logica extracts the relevant portion of the enterprise architecture (EA) of these two states. WDE said:
Figure 6.2: Core Activities of Lean & Mean-G (I) and Lean & Mean-DM (II)
“In migration projects EA provides us the starting-point and the destination.” The As-Is and To-Be states are captured in terms of EA elements, i.e. business architecture, data architecture and technology architecture.

- **Gap Analysis.** Once the As-Is and To-Be states are understood, gap analysis at the EA level is carried out. Gap analysis aims at understanding the gaps between the business structure, data model and technology architecture of As-Is and To-Be states. Once these gaps are identified, the migration team decides how to fill them. For instance, gaps between the conceptual data models of the two banks indicated the differences in the banking products offered by the two banks. To fill this gap the migration team uses an internal practice, called mapping workshops, designated for negotiations and handling gaps.

- **Data Analysis.** Within this activity the To-Be business data model is identified. We found that to carry out this activity, typical requirements engineering techniques such as conceptual data modeling are used to define the ideal data model after migration.

- **Data Design.** This activity relates to design of the To-Be data model. In doing this, the practitioner has to do trade-offs between the desired To-Be data model and the existing data model of both the source and target systems.

- **Data Model Transformation.** Within this activity As-Is data model is transformed to To-Be data model.

- **Data Conversion.** This activity relates to automatic conversion of the source data to the target data.

- **Data Validation and Distribution.** The practitioners emphasized that minimizing the cascading effects of rollbacks in case of a faulty data insertion is an important challenge in data migration. Data validation addresses this challenge by assuring the correctness of the data before data load. After validation, data is distributed into smaller sets to meet timeliness and consistency.

### Core Knowledge

Figure 6.3 captures the key knowledge elements involved in data migration projects and as such represents the essential information to work with. Each of the core knowledge elements addresses a different aspect of the migration. For instance, some knowledge address the “why” question that underpins a project (e.g. business goals motivating the migration), while other knowledge addresses “what”
(e.g. source and target EA) and “how” (e.g. migration type in the migration plan). Regardless of their type, these knowledge elements are the ones that repeatedly occur in data migrations. Knowing these knowledge elements, practitioners can assess early enough the costs and risks of eliciting the required knowledge, and plan accordingly.

The arrows in Figure 6.3 reveal that to create a certain output, there are certain types of knowledge that should be available. For instance, to create the migration plan the input knowledge such as business goals, data volume, time constraints, and risks have to be made available (see the left part of Figure 6.3). This addresses a key issue emphasized by practitioners i.e., discovering what needs to be made available too late in the process (Razavian and Lago 2012b). The core knowledge includes three main output knowledge elements: migration plan, ideal target data, and target data. To create the output knowledge elements, all together, 22 knowledge elements were used as input.

![Figure 6.3: Core Knowledge of Lean & Mean-DM](image)

Core knowledge does not represent all the required information in a certain data migration project, but it only represents the bare essentials. As noted in §6.3 core knowledge can be customized to represent the key input/output knowledge.
elements of a certain project. Figure 6.4 shows an example of such customization for Case II. This customization can be realized by adding new knowledge elements to the core or refining the existing ones into more detailed elements. The black boxes in Figure 6.4 illustrate the added knowledge elements and the grey ones represent the refined ones. As an example, for Case II the migration team underlined that to make the core knowledge more expressive, a number of important output knowledge such as migration phase or source data should be added to the core. In this addition, the pre-existing knowledge elements of the core (i.e. the grayed out boxes in Figure 6.4) act as anchors hooking the new elements to the core. In the same vein, some knowledge elements were required to be expressed in more detail and consequently were refined. As an example, the migration team emphasized that Key Stakeholders Responsibilities is too generic and has to be refined into two categories of Key Responsibilities for Negotiations and Key Responsibilities for Validation. As their main rationale for such refinement they noted that business stakeholders would take many responsibilities in the whole migration and distinguishing those responsibilities early in migration...
6.5. LEAN & MEAN APPROACH FOR DATA MIGRATION

is essential. In short, based on what is fundamental in each project the core knowledge should be enriched. In this enrichment the core acts as an anchor to add new knowledge or to specify a knowledge in more details.

6.5.2 Extensions

The core activities suggest ‘what to do’ without providing any specific guidance on ‘how to do it’. On one hand, the lightweight description of the activities provide directions in carrying out migration, while letting practitioners adopt their way-of-working. On the other hand, practitioners are not supported with sufficient guidance on how to do migration. This constituted one of the main topics of discussion in the feedback sessions, namely, there is a need to enrich the core activities with more guidance. This need is addressed in extension part of the Lean & Mean approach in which a set of practices are identified.

We provided an extension that binds specific practices to the core activities. Figure 6.5 illustrates this extension where practices are shown using rectangles and are bound to core activities using binds link. As noted, a practice provides a solution for a particular migration activity or aspects of that activity. Examples are practices for carrying out gap analysis or practices for data validation and distribution.

To explicitly document the practices, we devised a template exemplified in Table 6.2. We used the work of Clerc (Clerc, 2008) as a basis to define this template. Using this template one can specify a practice in terms of its main intent, the business goals that it supports, the situation in which the practice is emerged and applied, and finally the relationships between the practices. In § 6.6.1 we will describe how elements of this template help practitioners having a well supported choice between practices for application in data migration projects.

We found 14 distinct practices out of which 7 were present in both projects analyzed in the case studies. In Figure 6.5 each practice is labeled with case studies that the practice was present in (c1 for Case I and c2 for Case II). In addition, this Figure shows that some of the activities were supported by multiple practices. These practices were either complementary (i.e. they can be adopted together) or exclusive (i.e. they represent alternative ways of conducting the activity). For instance, two practices of Go/No-Go decision and pre-migration validation and distribution, used together in both projects, were complementary. On the contrary, semi-automated transformation and fully-automated transformation were exclusive. Regardless of being complementary or exclusive, the practices package past experience in carrying out core activities.
6.6 Implications of Lean & Mean Migration Approach (RQ-I)

Previous section presented the Lean & Mean-DM. This section answers the question of what would be the effect of having Lean & Mean-DM in place, when defining the migration approach of a project (RQ-I). To answer RQ-I we carried out the Case II cycle (see Figure 6.1) in which we applied the Lean & Mean-DM to the still active project of Case II. In the workshop of this cycle, together with practitioners, we defined the migration approach of Case II. In doing this, we observed various benefits of the Lean & Mean organization of the migration approaches, all of which contribute to efficiency. These benefits fall into three main categories: (i) increased reuse, (ii) improved problem-solving, and (iii) support-
Table 6.2: Example of codified practice

<table>
<thead>
<tr>
<th>Best Practice Name</th>
<th>Migration partitioning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intent</td>
<td>This practice intends to cope with volume of data and/or data interdependencies by partitioning the full migration</td>
</tr>
<tr>
<td>Overview</td>
<td>Data migration is happening in an extended period of time (e.g. one week) instead of a point-in-time fashion. Migration is partitioned in to a number of partitions (e.g. three partitions for each of data entities including customer, product, and financial information). After each of these partitions a baseline of data consistency is reached in the target system that allows the next partition to be inserted correctly</td>
</tr>
<tr>
<td>Business Goals</td>
<td>Timeliness and Correctness</td>
</tr>
<tr>
<td>Context</td>
<td>Where point-in-time migration is not possible because of extreme data volumes, complex data inter-dependencies, or other business requirements, segmentation of data into partitions is a good option. This practice can be used by Freeze/Sweep practice to avoid the possibility of the same piece of data being accessed and updated in both the source and target structures and systems. As such there is no concern about collisions (concurrent updates in both locations)</td>
</tr>
<tr>
<td>Relationships</td>
<td>Freeze/Sweep</td>
</tr>
</tbody>
</table>

...ing the existing way-of-working. These benefits are listed in three key findings presented in the following sections. Each finding is summarized in a Reflection Box, and is followed by a detailed discussion of the finding.

### 6.6.1 Increased Reuse

Reflection Box.1. Organizing migration approaches into Lean & Mean increases reuse.

- **F1.1.** The core facilitates reuse of know-how.
- **F1.2.** The extension facilitates reuse of concrete solutions.

In this study we found that organizing migration approaches into Lean & Mean increases reuse of migration approaches among migration projects. The direct relation between reuse of approaches and efficiency has been widely acknowledged [Boehm 1999; Basili et al. 1994; Henninger 1997]. According to Basili et al. [1994], efficiency is very often achieved by reusing and modifying over and over the same approaches. This is in-line with what was perceived by practitioners in this study; by facilitating reuse of (parts of) previous migration approaches, Lean & Mean-DM helps decreasing the effort needed for defining migration approaches. Interestingly enough, we found that the core and the extension of Lean & Mean-DM contributes to two distinct types of reuse: (i) reuse of know-how, and (ii) reuse of ready-to-use solutions.
Core Facilitates Reuse of Know-how (F1.1)

Prior to this study, the know-how of migration (i.e., knowledge of how to do migration) remained mostly tacit in stakeholders’ minds. Nevertheless, in the feedback sessions of the last cycle (Engineering Cycle 3) the migration team emphasized that the core not only externalizes such know-how, but also combines it into a coherent whole. This led us to the finding F1.1, i.e., the core serves as a consolidated know-how. For instance, WDE referred to such consolidated know-how as “a skeleton of activities and knowledge”. In this regard he said: “... it is true that there will be more or less a lot of uncertainties at this point but this core makes a good starting point for us. It shows the skeleton of activities and information”. We further found that the way two parts of the core (core activities and core knowledge) facilitate such reuse of know-how is very different. In the following we represent our findings in this regard.

Core activities as a reference model of migration activities During the workshop of the last cycle, we observed that the migration team took the core activities of the Lean & Mean-DM and built the project activities of Case II on top of those activities. In doing this they repeatedly, (i) selected an activity, (ii) recollected their past experience related to that activity, (iii) contextualized the activity for the specific project of Case II. This is evident in the following quote where after selecting the data conversion activity (see Figure 6.2 II), MTM noted their past experience related to this activity and further discussed how it needs to be contextualized: “… for data conversion we usually use mapping tables and conversion rules, and we have tools and procedures that use them, like in Case I. But in Case II this [mapping tables and conversion rules] will not be enough. We need a mechanism for letting business users to review and eventually override the data conversions”.

In this way, the core activities of Lean & Mean-DM were used as a reference model isolating the know-how related to ‘what needs to be done’. The migration team used this reference model as a reminder of the generic activities that happen in data migration, the activities that most probably need to be carried out in Case II as well.

Core knowledge to characterize migration projects Core knowledge, inherently, is a repository of consolidated information that are key drivers in migration. During our workshop we observed that practitioners reuse the knowledge elements in this repository to characterize different migration approaches. Characterization of migration projects is a prerequisite for identifying reuse opportunities (Basili et al. 1994). Such characterization further practitioners to relate different projects. For instance, MAM selected data volume from the core
6.6. IMPLICATIONS OF LEAN & MEAN MIGRATION APPROACH (RQ-I)

knowledge to relate Case I and Case II. In this regard (s)he commented: “Case I had the data volume of nearly 1.5 million whereas Case II deals with more than 140,000. This probably means that we need fewer increments in Case II”. Likewise, RNL pointed out level of business-user involvement as a characterizing factor of Case II and said “Case II is different from Case I because of level of business-user involvement which has to be much higher”.

The migration team further emphasized that the core knowledge enables framing the class of migration approaches suitable for certain characteristics. This enables systematic reuse of migration approaches. In this regard WDE said: “the core knowledge can help us define the prerequisites of certain migration approaches. For example, approaches suitable for data migrations with a high level of business involvement, or approaches for acquisitions with a single data owner”.

Extension Facilitates Reuse of Concrete Solutions (F1.2)

The migration team found the extension as a repository of concrete solutions for their known problems. Each practice in the extension captures a repeatable solution for recurring problems. For instance, migration partitioning practice, represented in Table 6.2 addresses a solution for the problem of extreme data volumes and complex inter-dependencies between data entities. Such solutions are emerged out of experience in various data migration projects and are likely to be reused in future projects too. By linking these solutions to the core activities, the extension addresses a key issue emphasized by practitioners, i.e., what are the options for carrying out a certain activity. For instance, for the validation and distribution activity, we identified four practices of dependency-based sort/cluster, pre-migration validation and distribution, go/no-go decision, and freeze/sweep (see Figure 6.5). By bringing order in the practices available for validation and distribution activity, the migration team was able to readily decide on use of these practices in the Case II as well.

As a co-product of this study, the migration team found the way practices are formed being useful for deciding ‘when’ or ‘how’ to reuse a practice. For instance, we observed that to decide on ‘when’ to reuse a solution, they referred to Intent and Context in Table 6.2. In the same vein, the Overview of practices initiated discussions addressing how to reuse a specific solution.
6.6.2 Improved Problem-Solving

Reflection Box.2. Organizing migration approaches into Lean & Mean helps problem-structuring.

- **F2.1.** The core knowledge facilitates identifying the problems.
- **F2.2.** The core activities facilitate framing the problems.

Defining a migration approach is a type of problem-solving process that includes recognizing problems, and finding ways to devise an approach to solve these problems. To efficiently define the “right” migration approach, practitioners need to make many well-reasoned decisions in a timely manner. As such, improving problem-solving of practitioners would positively affect the efficiency in carrying out migration. In this study we found that organizing migration approaches into Lean & Mean would improve practitioner’s problem-solving.

Schön (1991) suggests that the way designers frame problems determines the features they focus on. Zannier et al. (2007) found that the more structured the problem space, the more rational is the approach taken by designers. In this study we observed that the migration team used the core of Lean & Mean-DM to systematically identify the migration problems and structure them. We expect that such improvement in problem structuring would improve practitioners problem-solving as well. We further found that the core knowledge and the core activities lend themselves in such problem-solving in different manners. In the following, we will present our findings in this regard.

Core Knowledge Facilitates Problem Identification (F2.1)

Some designers might not explicitly identify the problems that they need to solve in migration. Instead, they think about the solutions. For instance, instead of exploring the issues that the migration faces, they might jump into fully-automated data mapping because it is perceived as a best practice. There are some negative implications with this approach. Firstly, the problems may be oversimplified and not explored thoroughly. Secondly, a designer may anchor on a solution, and is reluctant to shift even in view of contradictory information. This has been found in the behavior of professional designers by Tang and van Vliet (2012). One way to avoid this behavior is to use techniques or tools that support problem identification. In this study we observed that the migration team used the core knowledge as a tool for identifying the migration problems of Case II.

In the workshop, the migration team used the core knowledge as a starting point for identifying the problems. We found that they used the core knowledge as a checklist, that helps them not to miss important information. As noted in §6.6.1 the migration team used the core knowledge to characterize the project of
Case II. Such characterization was often followed by recognizing the peculiarities and consequent problems that the project faces. For example, after identifying confidentiality of data as an important characteristic of Case II, MAM said: “the level of confidentiality of data is an issue. We need to find out how to test the data because we are not allowed to check the data of VIP customers”. MTM, referring to business constraints, continued: “business constraints raise issues too, for example we should keep in mind that after a point-in-time there will be no acquiree bank as a legal entity. So we cannot expect any activity from acquiree bank after that time”.

In this way, the core knowledge worked as a tool for steering problem identification. Using this tool, practitioners can reflect the potential gaps in their understanding of the problem. In this way, overlooking key issues in design could be minimized.

Core Activities Facilitate Framing The Problems (F2.2)

After identifying the generic problems, the migration team needed to decide how to approach the specific problems in carrying out the migration activities. Throughout the workshop, they alternated between raising problems and proposing solutions. The solutions were mainly of type ‘how to carry out (a part of) an activity’. For some problems they had an immediate solution, whereas for others they left the solution open. In doing this, inherently, they followed the famous problem solution co-evolution (Dorst and Cross, 2001). In software engineering it is well known that solutions are discovered and evolved based on problem structuring. Software engineering design methods such as twin peaks (Nuseibeh, 2001) and problem frames (Jackson, 2005) support co-evolving of problems and solutions. In this study, however, how the migration team structured the problem was new. We observed that they framed these problems using the core activities. In other words, they used activities as containers to contextualize and accommodate their problems. This is evident in the following quote where MTM frames the issue of maximizing business involvement in the scope of gap analysis activity: “maximizing business involvement for gap analysis means that gaps have to be identified with active involvement of business stakeholders. The issue is to involve whom. It needs to be someone who has overall view on the domain and is aware of important gaps.”

To summarize, we found that core activities enable focusing on a certain activity and identifying or translating the migration problems for that activity. This facilitates a focused design discourse that helps to formulate the problems and find solutions (e.g. practices) efficiently.
6.6.3 Supporting Existing Way-of-working

Reflection Box.3. Organizing migration approaches into Lean & Mean facilitates support of existing way-of-working.

In this study we aim at efficient support of the ways people work. This goal resembles what happened in general in software process field, where they moved from processes that are strict, precise, and comprehensive (e.g., RUP (Jacobson et al., 1999), Spiral Model (Boehm, 1986), SSADM (Rose, 1992)) to lightweight, agile, and highly iterative approaches (e.g., Scrum (Schwaber and Beedle, 2001), Extreme Programming (Beck, 2005) or Lean (Ebert et al., 2012)). In this emerging perspective, being efficient comes together with being simple and being compliant with existing-way-of-working of practitioners (e.g., Ebert et al. (2012); Johnson et al. (2012); Kruchten (2011)). More precisely, this perspective follows the idea that more elaborate, precise, and comprehensive approaches cannot support practitioners’ challenges. Instead, approaches that are reduced to absolute bare necessities and adopt existing way-of-working, deliver the required quality without increasing costs and efforts (Ebert et al. 2012). However, in order to achieve these benefits it is of great importance to bring order and systemize such existing-way-of-working. This is what our Lean & Mean approach does.

Existing way-of-working of practitioners lies in the heart of the Lean & Mean approach. The practices, captured in extension, are systematic representations of the solutions that practitioners already adopted. By providing a very light-weight core that can be extended by practices, the Lean & Mean approach systemizes what practitioners already do. What was indicated as most useful was to set the core as a reference model of their migration process and tailor their migration processes using this reference. Since this reference model emerged out of existing projects, it reflected actual way-of-working. Furthermore, this reference model was indicated to be helpful since it fitted the social and organizational dynamics of migration teams in Logica.

6.7 Generalizability of the Lean & Mean Migration Approach (RQ-II)

As noted, our research goal was to try out the Lean & Mean-G in the real-world migration cases and see whether the core elements are generalizable. Generalizability in this study pertains to the degree to which the knowledge elements and the activities of the core in the Lean & Mean-G are common in the migration efforts. Lean & Mean-G was the result of an industrial survey in different types of companies (in-house and consultancies) and different market positions. In the
two case studies discussed in this chapter we confirmed its general applicability, at least in the specifics of data migrations. Further generalizability issues are discussed in § 6.8.

By following TAR, the Lean & Mean-G was changed in such a way that it matches the approach taken in the regarding approaches of Case I and Case II. In addition, the feedback sessions resulted in further changes in order to assure that the core was fundamental in the data migration approaches. These changes resulted in deviation from the Lean & Mean-G. The results of the analysis revealed that almost all the changes were of type addition or refinement of the core elements. This makes Lean & Mean-DM being specification of the Lean & Mean-DM. In the following, we will review the changes with respect to core activities and core knowledge.

6.7.1 Core Activities

Stakeholders agreed the processes from the Lean & Mean-G (Reverse Engineering, Transformation, and Forward Engineering) should remain largely unchanged and that two activities of migration planning and understanding the As-Is and To-Be states should occur prior to these processes. Activities of forward engineering process were refined into data analysis, data design, and data validation and distribution. This is to be expected given that the focus of these activities in Lean & Mean-G is on analysis and design of services whereas Lean & Mean-DM focuses on a certain aspect of services, namely, data. In the same vein, the activities of data model transformation and data conversion are specification of legacy element transformation and legacy translation respectively. This confirms that the activities of Lean & Mean-G are core and the activities of Lean & Mean-DM are specific forms of the Lean & Mean-G.

6.7.2 Core Knowledge

The complete list of changes made to the knowledge elements of the Lean & Mean-G are listed in Table 6.3. Out of 15 knowledge elements of Lean & Mean-G, 7 were repeated without any changes in the Lean & Mean-DM, 7 were refined, and 1 was removed.

It should be noted that, the knowledge elements that were repeated in Lean & Mean-DM, were the ones that stakeholders confirmed being fundamental in their migration projects. They further confirmed that these knowledge element are self-contained and do not need further refinements.

Some of the knowledge elements were too generic to adequately express the knowledge needed in the data migration projects. For instance, while stakeholder confirmed migration plan being fundamental output knowledge, they felt that it is too generic and it needs to be supported with more details. Accordingly, it was
refined into the following set of knowledge elements: increments, timeline, migration type, level of automation and tool support. With respect to migration type, MAF said “. . . what I am missing here is the type of migration. Typically, in our migration projects we have one (or a combination) of three approaches: big-bang data migration, incremental data migration, event-based data migration and bi-directional synchronization. Based on the characteristic of the project we choose one of those. In the Case I well in advance we chose incremental-bigbangs”.

Other examples of the refined knowledge elements pertain to the differing focus of service migration as compared to data migration. Service migration focuses on creating services as output, whereas, data migration focuses on creating data. Consequently, the two knowledge elements of ideal service and software service were changed to ideal target data and target data, respectively.

The stakeholders emphasized that the knowledge elements in Lean & Mean-G are fundamental, but minimal. In the feedback sessions, as the assessments of the core knowledge proceeded, some knowledge elements were added to the core knowledge. Much knowledge elements that were previously left implicit were made explicit later by adding more elements such as key stakeholders responsibilities or data owner. For example, related to key stakeholders responsibilities MTF said: “When we are in the planning stage, we need to decide about to what extent stakeholders have to be involved. For instance, in Case I business stakeholders were decided to be involved mainly in the early phases of understanding the context, as well as in the validation. But in Case II we need much more involvement. We need to plan this in early stages of the project”.

Only one of the input knowledge elements of Lean & Mean-G (As-Is code) was removed. This was because the stakeholders found As-Is code of being out of scope for data migration projects. In summary, we can conclude that the core knowledge of Lean & Mean-G is generalizable.

6.8 Threats to Validity

Lincoln and Guba (Lincoln and Guba, 1985) argue that qualitative research should be internally and externally valid. Below, we discuss the validity threats of this work and how we addressed them.

**Internal validity.** This validity relates to “how” the research is carried out, and whether the used methods are credible. Yin (Yin, 2008) suggests triangulation as tool for ensuring internal validity. The basic idea is to gather different types of evidence to support conclusions. Internal validity of this study is supported by triangulation at data source level. Data source triangulation was achieved since data (e.g. activities carried in migration) were gathered multiple times, from documents, from interviews, and from observations in the feedback.
### 6.8. THREATS TO VALIDITY

Table 6.3: Mapping Knowledge Elements of Lean & Mean-DM on Lean & Mean-DM

<table>
<thead>
<tr>
<th>Knowledge Element (Lean &amp; Mean-G)</th>
<th>Knowledge Element (Lean &amp; Mean-DM)</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input Knowledge</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Business Goals</td>
<td>∼</td>
<td>no change</td>
</tr>
<tr>
<td>Risks</td>
<td>∼</td>
<td>no change</td>
</tr>
<tr>
<td>Costs/Investments</td>
<td>∼</td>
<td>no change</td>
</tr>
<tr>
<td>As-Is Enterprise Architecture</td>
<td>∼</td>
<td>no change</td>
</tr>
<tr>
<td>To-Be Enterprise Architecture</td>
<td>∼</td>
<td>no change</td>
</tr>
<tr>
<td>New Requirements</td>
<td>∼</td>
<td>no change</td>
</tr>
<tr>
<td>As-Is Quality Aspect</td>
<td>∼</td>
<td>no change</td>
</tr>
<tr>
<td>Constraints</td>
<td>Business Constraints</td>
<td>refined</td>
</tr>
<tr>
<td></td>
<td>Time Constraints</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Data Volume</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Data Model Complexity</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Inter-dependencies</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Between Applications</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Source and Target Data Stores</td>
<td></td>
</tr>
<tr>
<td>Service-specific Characteristics</td>
<td>Data Characteristics</td>
<td>refined</td>
</tr>
<tr>
<td>As-Is Design Models</td>
<td>As-Is Data model</td>
<td>refined</td>
</tr>
<tr>
<td>As-Is Design Constraint</td>
<td>Data and Domain Interdependencies</td>
<td>refined</td>
</tr>
<tr>
<td></td>
<td>Infrastructure Capability</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Key Stakeholders Responsibilities</td>
<td>added</td>
</tr>
<tr>
<td></td>
<td>As-Is and To-Be Conceptual Models</td>
<td>added</td>
</tr>
<tr>
<td></td>
<td>Data Owner</td>
<td>added</td>
</tr>
<tr>
<td></td>
<td>Source and Target Data Model</td>
<td>added</td>
</tr>
<tr>
<td></td>
<td>Data Scope</td>
<td>removed</td>
</tr>
<tr>
<td>As-Is Code</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Output Knowledge</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Migration Plan</td>
<td>Increments</td>
<td>refined</td>
</tr>
<tr>
<td></td>
<td>Timeline</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Migration Types</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Tool Support</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Level of Automation</td>
<td></td>
</tr>
<tr>
<td>Ideal Services</td>
<td>Ideal Target Data</td>
<td>refined</td>
</tr>
<tr>
<td>Software Service</td>
<td>Target Data</td>
<td>refined</td>
</tr>
</tbody>
</table>
sessions and workshops.

In this study, the interviews were mainly conducted by a single researcher and hence subjective interpretations might exist. To mitigate this threat, the interview guide was checked and validated by senior researchers experienced in software engineering, empirical studies and migration. The role distribution of participants in this study in Table 6.1 shows that a population selection bias based on role is unlikely. We also intentionally chose senior practitioners with considerable experience in migration projects. This gives confidence that participants are familiar with migration projects throughout Logica.

Threat to the validity of the analysis is in general applicability of the codes used for characterizing and classifying migration approaches. An assuring factor in this regard is that the start-list of codes is extracted from the Lean & Mean-G published in a service-oriented computing forum, after being peer reviewed by experts in the field. In addition Lean & Mean-G stems from interview with seven SOA solution provider companies. This further consolidates its general applicability. An additional threat is that the analysis is performed by a single researcher. Nevertheless, bias and omission issues is (partially) mitigated by having the codification results double-checked by both a second researcher and the interviewees. Finally to assure accuracy of findings, we used member checking method that is getting feedback on the findings from the subjects who provided the data in the first place. Member checking was extensively used as the Lean & Mean-DM and initial findings were refined and confirmed in the feedback sessions of two last engineering cycles. In addition, adopting TAR with the main focus on the practitioners point of view on usability and effects provided a sound correction of the bias of the researchers.

**External validity.** In this regard, a possible threat, inherent in any study based of case study research, is that the results are subjective. Yin points out that external validity can be improved by using replicated study-designs. In this study report, we present results that are based on findings made in two different case studies. This reduces the influence of the concrete cases. A threat to generalizability of results is that the study was conducted at one company which means the findings are specific to this study. In order to cover different but relevant perspectives on the subject matter, we chose practitioners with different roles of chief architect, sector manager, business analyst and project manager. In addition, we chose senior stakeholders who hold extensive experience and are aware of characteristics of migration projects. This pats in favor of generalizability of our results.
This chapter presents a Lean & Mean approach for data migration experimented in technical action research in industry. The research presented here has addressed a novel approach for organizing migration approaches, called the Lean & Mean approach. This approach is based on the idea of separating the core from the extensions. To the best of our knowledge this is the first study of the kind in the context of migration.

Most related work is Semat (Software Engineering Method and Theory), which has the very broad focus of software engineering in general. Semat is an initiative that was introduced at the end of 2009 by Ivar Jacobson, Bertrand Meyer and Richard Soley. This initiative recognizes the important issues being in the field of software engineering such as the abundance of unique methods that are hard to compare, and the gap between academic research and its practical application in industry. To solve these issues, Semat envisions a solution, which is based on the idea of separating the essence of software engineering from complex context-dependent details. The essence is captured in Semat kernel and the context-dependent detail are represented in practices. Our approach is similar. We worked together with practitioners to identify what is the essence of data migration and represented it in terms of core. In addition, we developed the extension that brings order in the data migration practices and provides a common ground for comparing and defining practices. In this way, we have addressed above mentioned issue of abundance of unique, incomparable methods. In addition, by following TAR we took a step towards filling the theory and practice gap.

The industrial perspective of this study brought interesting observations and lessons learned and as such contributed to the novelty and industrial relevance of our approach. The reactions to the Lean & Mean approach led to some usage implications that we as researchers did not expect them. In addition to the effects discussed in §6.6 we identified the following lessons learned:

- **Core is the starting point.** During this study practitioners repeatedly emphasized that although each project is unique, and the migration approaches used in them cannot be simply repeated, starting from core can save them major efforts. They confirmed that core is generic enough to have wide applicability and as such it helps them avoid starting from scratch.

- **Lean & Mean supports creativity.** The practitioners acknowledged that by bringing structure on relevant knowledge and activities, Lean & Mean should help not overlooking important information when defining migration approaches. In addition, by providing only fundamental information, and not providing a prescriptive, step-by-step procedure Lean & Mean offers
room for being creative.

- *Lean & Mean supports teamwork.* Having a shared mental model of the approach for carrying out migration was emphasized to be crucial for successful teamwork. Practitioners acknowledged that the Lean & Mean approach can act as such a shared mental model and as such it encourages teamwork.

The essence of our approach is thus: the core elements that are repeating in different projects must be separated from the project-specific ones; the advanced needs of certain projects should be supported as an extension to the core; and the core and the extensions should be under constant evolution. Making the fundamental elements visible provides the necessary information for making informed decisions about what is expensive to change, such as what should drive the migration (business goals or To-Be architecture) or what activities must be performed.

This study discussed how organizing migration approaches in form of Lean & Mean positively affects efficiency of migration. This result is based on the experience with the effects of Lean & Mean-DM in Case II. While initial results are promising, future work includes empirical validation of the effects of Lean & Mean in efficiency to be measured, in terms of cost to value ratio. This requires engagement of a broader set of projects.
Change Viewpoint: An Extension for Dealing with Change

The panel (Chapter 5) emphasized that although enterprises perceive the importance of change, practitioners often report that dealing with changes occurred during migration is still a challenge. One way of addressing change is to guide the decision making of architects using a viewpoint. This chapter presents a change viewpoint as an extension to the knowledge view of the Lean & Mean approach introduced in Chapter 5. The change viewpoint frames the concerns of architects related to change. In an attempt to better understand what the architects’ concerns are and the kind of support they need for dealing with change during migration, we have conducted a case study in the Dutch Electronic Health Record system. Based on the results of this case study, the change viewpoint, aiming at guiding the architects in their reasoning, has been defined and discussed.

7.1 Introduction

The notion of change lies in the heart of any migration effort including migration to services. Migration to services has been widely established in industrial practice. As discussed in Chapter 4, researchers have proposed various approaches for migration to services for over a decade, but those approaches often do not fit the fundamental problems and goals of practice. One of these problems, repeatedly emphasized by practitioners (see Chapter 5), revolves around the issue of dealing with change. Changes can occur in at least two ways. On one hand, migration can be itself interpreted as a series of changes. Such changes follow a path bringing an existing system to a target service-based system. On the other hand, changes, potentially originated outside the migration projects, can occur during migration. For instance, if the technology to be used by the target system changes (due to e.g. a business partnership with a different vendor), then
some knowledge (e.g. design decisions) about the target system has to change, too. Likewise, the migration approach stating the path towards the target system needs to change. This chapter focuses on the second type of change. In our work, this type of change was considered of being especially critical by practitioners. If not dealt with appropriately, these changes can result in extra costs and efforts, or even project failure.

We investigated the problem of dealing with change in the Electronic Health Record System (EHRS for short) migration project in the Netherlands. During the course of this migration project one of the key laws related to healthcare was changed. This change initiated a domino effect in introducing other changes ranging from business drivers to design decisions. To apply these changes, the EHRS architects needed to adapt their migration approach. This led them to various concerns such as how change in legislation affects their business goals and whether design decisions are still viable.

To address these concerns, in this chapter we extend the Lean & Mean approach with the change viewpoint. We chose case study as our research approach because we sought for close up reality of what it is like when important changes are imposed during migration. The case study further enabled us to observe the architects mental model of ‘how to deal with change’. The shared mental model revealed that architects deal with changes in an iterative manner. In each iteration, they explore the sources of change, select one source and focus on the chain of changes triggered by that source. Based on this mental model and the concerns of architects we defined the change viewpoint.

The change viewpoint helps architects reasoning when dealing with change. For instance, architects regularly face complex networks of dependencies between knowledge elements. Being unsupported, it is very difficult to reason about the impact of a change (e.g. change of a business goal) on other knowledge elements (e.g. design decisions). The change viewpoint facilitates managing complexity by scoping to one selected chain of changes at a time, and by filtering out the knowledge and dependencies that do not matter, facilitates managing complexity. In short, the change viewpoint improves architects reasoning by guiding them to (i) identify only the types of knowledge that are relevant for change impact analysis, (ii) consider the related changes simultaneously, instead of in isolation, (iii) determine the sequence of activities to be followed in order to deal with change.

7.2 Context and Groundwork

This chapter provides an extension to the knowledge view of the Lean & Mean SOA migration approach presented in Chapter 7. By providing a list of knowledge elements that need to be made available, the knowledge view guides the
7.3. NATION-WIDE ELECTRONIC HEALTH RECORD

architects in understanding the source (As-Is) and target (To-Be) states. When during the course of migration changes occur in either of these two states, the knowledge view cannot fully support dealing with change. This is because in this view, the relations between the knowledge elements are left implicit. As a result, the reasoning of the architect for determining the impact of change is left unsupported. For example, when the business drivers change, the knowledge view cannot guide the architect in determining whether requirements or design decisions would change. In this chapter we fill this gap by proposing a change viewpoint framing the key concerns of architects related to change.

The change viewpoint is developed in the context of the EHRS case study. In the following, we shortly introduce the three steps taken for dealing with change in this case study.

1. To understand the EHRS before change, we extracted its associated knowledge view. In doing this, we elicited the knowledge elements that are instrumental for change analysis. These knowledge elements are explained in §7.4.

2. To support dealing with change, we defined change viewpoint that frames the key change-related concerns of architects. The definition of this viewpoint, as mentioned, is based on the shared mental model of the EHRS architects. §7.5 describes the change viewpoint.

3. To apply the change viewpoint we used change scenarios in the EHRS. Each scenarios represents a set of steps for dealing with specific changes in the EHRS. Using the scenarios, the change viewpoint is examined to see whether it supports the required line of reasoning needed for dealing with change. This is discussed in §7.6.

7.3 Nation-wide Electronic Health Record

In this section we review a real-world case of change in a large-scale migration project, namely, the EHRS. The Dutch government envisaged the EHRS to serve as the nation-wide channel for exchanging patient health records information. The main goal of the system is to make patients’ medical history accessible on a nation-wide scale, thus optimizing healthcare services and reducing treatment errors. Development of EHRS was planned as a migration project, aiming at integrating existing healthcare information systems. That is, a migration from a number of silo systems (source state) to the EHRS service-based system (target state). One of the most notable characteristics of this migration is the influence of legislation on the design decisions. For instance, EHRS was decided to be developed as a partially decentralized service-based system, because legislation
does not favor storage of patient information in a central infrastructure or outside (the control of) a healthcare provider (van’t Noordende 2010).

To support migration to EHRS, the Dutch healthcare Minister introduced the LSP law, mandating all healthcare providers to connect to EHRS for exchanging health records. The migration project was thus planned and started with the LSP law as one of its key underlying assumptions. This assumption had to be later changed, however, as the LSP law fell into disgrace by upper house (Senate): the EHRS programme was no longer mandated and care providers now migrate to the national EHRS only on a voluntary basis. In response to this change, which originated outside the migration project, the EHRS migration approach needed to be adapted.

The architects were now faced with the following questions: how these changes impact the target service-based system and to what extent already partially migrated services can be used. In the rest of this chapter we will explain our approach to answering these two questions.

7.4 Understanding EHRS using Knowledge View

For dealing with change, we adopt a knowledge-based approach. For this, we first elicit the key knowledge elements that describe both the problem and the solution space of the target system. To elicit the knowledge, we used the knowledge view introduced in Chapter 5 as a checklist and asked the architects “which knowledge is relevant to be externalized in order to deal with change”. In answering this question, the knowledge view helped us to (i) focus on core knowledge elements that really matter in the EHRS migration case, and leave out irrelevant elements (ii) bring structure in what needs to be externalized (iii) identify the gaps in our
UNDERSTANDING EHRS USING KNOWLEDGE VIEW

understanding. Figure 7.1 shows the types of knowledge that were elicited in this case study and the relationships among them.

Assumptions. Withdrawal of LSP law imposed changes in the (hidden) assumptions behind EHRS design decisions. To identify these assumptions, we looked for basic information taken for granted as being factual by architects. As a result, the following were identified:

- **BSN as unique ID**: Patients are identified using a unique number (BSN, known as nation-wide unique citizen number), which can be looked up by means of a separate BSN verification service.

- **EHRS as single communication path**: According to the LSP law all health-care providers have to connect and communicate via EHRS. As such all interactions required for finding and accessing patient records go through the EHRS.

- **Index right**: By signing a contact with the healthcare providers, EHRS is qualified to keep an index on patients data.

- **Opt-out Policy**: Data will be automatically made accessible for all patients, unless the patient has registered an objection against electronic exchange.

- **Governmental Financial Support**: EHRS is subject to financial support from government.

Business goals. Business goals were identified as a key type of knowledge that might undergo change. The architects listed the following business goals underpinning EHRS. To represent the goals we borrowed the idea of business goal scenarios [Clements and Bass, 2010] and expressed them in terms of four elements: goal subject, goal object, goal measure and the goal itself. As such, for each business goal we were able to assess the constituent elements and pinpoint the ones that were subject to change.

- **Promote evidence-based medicine**: Ministry of Health (goal subject) desires that healthcare professionals (goal object) benefit from better decision support (goal) and will be satisfied if the comprehensive clinical data of patients are provided to healthcare providers in a just-in-time manner (goal measure).

- **Improve quality of care**: Ministry of Health (goal subject) desires that Dutch citizens (goal object) benefit from better quality of care (goal) and will be satisfied if administration costs are reduced (goal measure) which then in turn can free up time and money for patient care.
Privacy: Ministry of Health (goal subject) envisions that patients (goal object) benefit from privacy (goal) and will be satisfied if relevant healthcare professionals can access their health record data (goal measure).

Reduction of costs: Ministry of Health (goal subject) desires that Dutch citizens (goal object) benefit from cheaper medical services (goal) and this will be satisfied if less unnecessary hospital visits happens (goal measure).

Support mobility: Ministry of Health (goal subject) desires that Dutch citizens (goal object) receive equally effective medical care in mobility and this will be achieved if the comprehensive clinical data of patients are provided to healthcare providers at nation-wide scale (goal measure).

Support innovative healthcare: Ministry of Health (goal subject) desires that Dutch citizens (goal object) benefit from innovative healthcare support and this will be satisfied if healthcare providers could retrieve data using articulated complex queries (goal measure).

Quality requirements. In order to assess the impacts of change we needed to recognize what quality requirements had influenced the design decisions of the EHRS. The following quality requirements are the key ones:

- Confidentiality ensures that access to information/services is granted only to authorized subjects.
- Authentication ensures that the indicated author/sender is the one responsible for the information.
- Integrity which guarantees that information is not corrupted.
- Availability ensures that EHRS provides the complete patients’ data to the relevant healthcare providers in a timely manner.

Design issues and decisions. In order to assess whether design decisions are still viable, we needed to recover the decisions taken and the issues they resolve. Furthermore, for each issue we needed to recover the alternative solution options considered by the architects (see design options in Table. 7.1). Besides, for both chosen options (i.e. decision) and rejected ones it is important to recover the pros and cons considered at the time of decision making. In other words, we needed to know for each issue what criteria were initially used in constraining options and making design decisions (see Selection Criteria in Table. 7.1). Finally we needed to identify the priorities and the rationale behind each decision. To identify the design issues, regarding solution options and decisions we used a template proposed in [Gu et al., 2010], and filled it in together with the architects.
7.5. USING CHANGE VIEWPOINT TO DEAL WITH CHANGES

Table 7.1: EHRS Design Space

<table>
<thead>
<tr>
<th>Design Issue</th>
<th>Design Options</th>
<th>Selection Criteria</th>
<th>Design Decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>D2: How to delegate authorization to patients data?</td>
<td>Opt1: decentralized delegation Opt2: centralized delegation</td>
<td>Confidentiality, Integrity</td>
<td>Opt1</td>
</tr>
<tr>
<td>D3: How to authenticate healthcare providers?</td>
<td>Opt1: UZI Cards, Opt2: user name/password</td>
<td>Confidentiality, Integrity, Usability</td>
<td>Opt1</td>
</tr>
</tbody>
</table>

Appendix D shows an example of detailed description of design issues, options and decisions. In total, we identified 6 main design issues (summarized in Table. 7.1).

Change in EHRS is captured as change in its constituent knowledge elements, in the change viewpoint. § 7.5 describes the change viewpoint, and § 7.6 exemplifies how this viewpoint illustrates the impact of change on the EHRS knowledge elements.

7.5 Using Change Viewpoint to Deal with Changes

The withdrawal of the LSP law caused architects various concerns, such as “what changes” and “how to incorporate the changes”. To address these concerns, the architects needed to look at EHRS from the specific standpoint of change. For this purpose, we defined the change viewpoint that frames and visualizes the key concerns of architects related to change. To define the viewpoint, we followed the advice of Kruchten (2011) and Holt (2002), and started with observing what architect do when dealing with change. This led us to find the EHRS architects’
mental model for dealing with change. This mental model is basis of the change viewpoint. This way, instead of devising a complex viewpoint addressing the concerns that we think are important, we provide a viewpoint that is compliant with architects’ conceptualization of the approach for dealing with change. §7.5.1 explains the architects’ mental model and §7.5.2 describes the change viewpoint.

7.5.1 Architects’ Mental Model of How to Deal with Change

To understand how the architects of EHRS typically deal with change we examined the key changes together with three lead architects and we asked them how do they determine the impact of changes. As a result, we found that architects tend to have a common mental model of dealing with change. To explain this mental model we first use the example of impact of withdrawal of LSP law on design decisions, described in Box. 1.

**Box. 1.** The architect explores the business drivers and selects the assumption withdrawal of LSP law as the source of change and carries out the following steps:

1. The architect finds the requirement(s) that reflect the changing assumption.
2. The architect changes the requirement to the new version.
3. The architect finds the design issues that are influenced by the changing requirement as well as their associated options, selection criteria, rationale and decisions. Furthermore, she/he re-evaluates the options and aligns the rationale behind selection/rejection of an option.
4. If needed she/he changes the design decision.
5. The architect finds other decisions that are influenced by the changed design decision.

The approach presented in Box. 1 is illustrated in Figure 7.2 which as we later show, forms the basis of change viewpoint’s visualization. As shown in Box. 1 the mental model embraces a sequence of two activities of find and change. In Figure 7.2 this sequence of find and change is represented as the set of arrows forming a “zigzag”. In addition, Figure 7.2 shows the knowledge elements before the changes (on the left hand side) and after the changes (on the right hand side). The dashed arrows represents inter-dependencies between the knowledge elements related to different categories of knowledge.

In general terms, the shared mental model, called the find & change approach, works as follows: explore and select one source of change at a time, scope and find only the elements that are influenced by that source, make decision about the required changes and eventually change them. In the following section we will
7.5. USING CHANGE VIEWPOINT TO DEAL WITH CHANGES

explain how the find & change approach is used to define the change viewpoint.

The change viewpoint is defined based on the ISO/IEC/IEEE 42010 standard for documenting views and viewpoints (International Organization Of Standardization, 2011). Accordingly, the following describes the change viewpoint’s purpose, concerns, metamodel, and conforming notation.

Purpose: The purpose of this viewpoint is to guide the reasoning required for dealing with change.

Concern 1: What is the source of change? Identifying the source of change is a key issue in managing change. It is fundamental, for instance, to recognize whether a change is internal to the system (e.g., change of an architectural decision), or it roots in a change in a wider environment (e.g., change of rules or regulations); whether it is a change in a business driver that answers the fundamental “why” question that underpins a project, or it is a change in requirements that address the “what” and “how” of the project. Of course changes in requirements need to be managed, but a change in business drivers is more complex as they provide context for both requirements and design decisions and can so have a broader potential impact. Thus to support the architects the change viewpoint is required to both explicitly represent the source of change, and distinguish whether this source is external or internal to the system context.

Concern 2: What is the impact of change? To manage the changes ar-
chitects are concerned with understanding the impact of change on other knowledge elements. To address this concern they often need to “trace” the source of change to its dependent knowledge elements. For example, when the assumption LSP law changes in EHRS case, the architects would like to know which requirements and design issues were inspired by this assumption, and if eventually change affects them. Being able to trace the dependent knowledge elements is essential for determining the implication of a change. Thus the change viewpoint needs to visualize these trace links. However, should all possible trace links between the knowledge elements be visualized?

Between knowledge elements there can be different types of dependencies. To improve the understandability of change viewpoint, we decided that instead of providing the complete picture of all traces between knowledge elements, customize the traceability for the situation at hand. To this aim, we borrowed the idea of scoped traceability (Lago et al., 2009) which is to use only those traces needed for the activities carried out by stakeholders. In our work this means to use only the traces supporting the find & change activities.

The “zigzag” in the middle of Figure 7.2 represents the traceability path for find & change activities. The arrows labeled f and c in the traceability path, represents the trace links to be traversed in find and change activities respectively. In the following, we describe our reasoning related to this traceability path. The f traces are needed because throughout the find activity the architect has to trace the elements that are influenced by the changing element. For example, if design decision central service broker (see Figure 7.2) is influenced by confidentiality, then a change in confidentiality, will likely change this decision too. During the change activity the architect needs to keep track of traces between elements that are changing (i.e. related by link c). As noted, the architects’ mental model embraces subsequent activities of find and change carried out iteratively. Consequently, the trace links graphically form the “zigzag” shown in Figure 7.2.

### 7.5.3 Change Viewpoint: Metamodel and Conforming Notation

The metamodel presented in Figure 7.3 presents the constructs of the change viewpoint and their inter-relations. The conforming notation of this metamodel is presented in Figure 7.4. Knowledge Elements are presented by squares; the gray one represents the Changed Knowledge Element. Note that Knowledge Clusters, distinguished by the types of knowledge elements presented in the ontology in Figure 7.1 constitute other clusters or knowledge elements. The Change Impact Path, constituting a series of Find Trace and Change Trace, links the clusters in Before and After. Note that change views shall use the notation shown in
7.5. USING CHANGE VIEWPOINT TO DEAL WITH CHANGES

Figure 7.3: Metamodel of Change Viewpoint

Figure 7.4: Conforming Notation of Change Viewpoint
Figure 7.4 and adhere to the metamodel presented in Figure 7.3. Figure 7.2 illustrates an example of the change view.

The change viewpoint specifically addresses Concern 1 by visualizing the source of change and scope. The scope box represents the boundary of migration. Any knowledge element within the box represents a piece of information that is internal to migration problem at hand and constitutes the context of migration. Knowledge elements are categorized into knowledge clusters. In this way we represent different types of knowledge and their hierarchical relationships using the ontology represented in Figure 7.1.

The change viewpoint also addresses Concern 2 by illustrating the impact of change. For this it visualizes the change impact path between the knowledge clusters. The change impact path, as noted, pinpoints the path to be taken to assess the impact of change (through link $f$) and apply it (through link $c$) in an iterative manner. The dependent knowledge elements are represented using link is influenced by. The panes Before and After indicate the state of knowledge elements, namely before or after change.

### 7.6 Applying the Change Viewpoint in EHRS

In this section we exemplify how the change viewpoint guides dealing with changes in the EHRS case. We introduce two scenarios of EHRS representing the following two tasks of the architects: (i) scoping change in problem space and (ii) evaluating impact on design decisions. In the following, we describe the scenarios using the template proposed in (Tang et al., 2010). For each scenario we will explain how its associated change view helps to draw reasoning required for carrying out the scenarios.

#### 7.6.1 Scenario 1: Scoping Change in Problem Space.

**Description:** An architect wants to assess if business drivers would change due to a change in a wider environment.

**Problem:** By withdrawal of LSP law the architects are now faced with the task of determining how this change, external to EHRS, would affect the EHRS business drivers.

**Solution:** The change view, shown in Figure 7.5 illustrates the withdrawal of LSP law as external source of change. By addressing concerns 1 and 2, we expect such explicit representation of source of change to help the architects to (i) bring focus to change impact analysis due to withdrawal of LSP law only, (ii) justify the consequent changes (iii) facilitate prioritization of the changes based on their importance or risks related to the change. For instance, in our study the
EHRS architects argued that because compliance with legislation is obligatory accommodating the change in LSP law gets high priority.

By linking the withdrawal of LSP law to knowledge clusters (see link f in Figure 7.5), the change view highlights the types of knowledge that need to be assessed against change (assumptions and business goals). This brings the architects attention to the related knowledge elements directly related to the change.

In evaluating the assumptions, the EHRS architects found that except BSN as unique ID, the rest of pre-existing assumptions were subject to change (shown in gray in Figure 7.5). The following explains the reasoning behind the changed assumptions. The withdrawal of LSP law implies that the EHRS has no longer the legal rights to keep index to patients files. As a result, the assumption index right is no longer valid and has to change to no index right. Besides, after the change in LSP law the healthcare providers will now only link to EHRS on a voluntary basis and we will have multiple communication paths instead of single communication path. Also as a side effect of LSP law withdrawal, the opt-out policy changes to opt-in policy.

The detailed elements describing each business goal are goal subject, goal object, goal measure or the goal itself (see Figure 7.1). These detailed elements are not shown in Figure 7.5. However, during the reasoning the architects analyzed each business goal and found that promote evidence-based medicine and support mobility change only in the way these goals need to be measured (i.e. goal measure). Regarding the promote evidence-based medicine, the architects pointed
that goal measure should be changed to supporting partial information instead of comprehensive information. This is because linking to EHRS is optional now and as a result nation-wide support of patients information is not viable. As shown in the change view (see pane After in Figure 7.5), the changes in business driver become internal sources of change. Hence, each calls for a distinct change impact analysis using the change view themselves. Scenario 2 addresses the impact analysis related to one of these internal sources of change, namely No index right assumption.

7.6.2 Scenario 2: Design Decision Impact Evaluation.

Description: An architect wants to evaluate the impact of a change of a business driver on the design decisions.

Problem: EHRS is no longer allowed to index to patients’ information (change of index right to no index right). The architects need to evaluate and identify the impact of this changing assumption on design decisions, so that the design decisions and new situation are consistent.

Solution: The change impact path in the change view shown in Figure 7.2 represents the find & change activities to be carried out in this scenario. We expect that such explicit representation of the change impact path helps architects in (i) identifying the requirements and design decisions that need to be assessed against change impact, and (ii) determining the sequence in which these knowledge elements are assessed.

By linking no index right to the quality requirements, the change view helps architects to scope the impact analysis to the quality requirements that depend on this assumption. In evaluating the quality requirements, the architects found that confidentiality was the only one influenced by this assumption. The following explains the rationale behind change of confidentiality. Initially, when EHRS was legally allowed to know the location of patients info, confidentiality was limited to ensure that access to patients information is granted only to authorized healthcare providers (i.e. the ones that have treatment relationship with the patients). With the change of index right, confidentiality should embrace the location of patients data as well. This means that only healthcare providers owning the patients data know where the data is located.

The next step is to analyze the impact of change in confidentiality on the Design cluster (indicated by f link leaving confidentiality). Here, we need to assess against change the design issues that directly concern confidentiality or have the confidentiality as part of their selection criteria. The change view in Figure 7.2 shows the 5 design issues directly influenced by confidentiality. Out of these issues only the decision related to D6 (i.e. How to retrieve patient data?) was affected by change. According to this decision, namely central service
broker, patients information are retrieved using a service provided by EHRS that acts as a broker and locates patients information. By withdrawal of LSP law, however, EHRS is not allowed to index to patients’ information and as such this decision is not viable. This further results in selecting the other option of **P2P information sharing** that satisfies confidentiality as it eliminates the role of third-party and enables information exchange amongst healthcare providers as peers. **P2P information sharing** focuses on interconnecting the healthcare providers using peer-to-peer (P2P) communication technology and supporting information exchange via broadcasting the queries. Further, although issue D2 (i.e. How to delegate authorization to patients data?) directly addresses confidentiality, the change does not impact the decision. This is because the decision (i.e. UZI cards for authorization) is not in conflict with the new definition of confidentiality. Similarly, D3 and D4 also remain unchanged.

Note that the change of D6 decision, becomes itself a source of change in other knowledge elements. Impact of this change is in turn visualized with a new change view with **P2P communication** decision as a source of change. In evaluating the impacts of this source, new change decisions potentially also invalidate the previous change decisions.

### 7.7 Discussion

In the following we summarize how the change viewpoint helps architects reasoning when dealing with change.

*Supporting focus on chain of changes.* The change viewpoint brings attention to the chain effect of a certain source of change. For instance, the pane After in Figure 7.2 visualizes the domino effect of no index right. This enables the architects to focus on the complete chain of changes triggered (or linked to) a certain source of change. While this brings attention to one and only the changes that are related, it also seamlessly compels the architects to consider the changes simultaneously (i.e. in the same view). In this way, we solve a problem practitioners often report, namely considering changes in isolation and discovering their dependencies too late in the process.

*Bringing structure on relevant knowledge.* According to [Tang 2011](#), important information that is required for design reasoning may be overlooked by designers. We argue that knowledge clusters by bringing structure on what needs to be evaluated, help to avoid overlooking the important information. In this way, a key issue emphasized by practitioners (§ 5.3) is addressed i.e., what types of knowledge might undergo changes. In short, by presenting architects with all required knowledge related to change, the change viewpoint should enrich architects reasoning.
CHAPTER 7. CHANGE VIEWPOINT: AN EXTENSION FOR DEALING WITH CHANGE

Supporting logical thinking. The change impact path structures the find & change activities without imposing a strict step-by-step process. It follows the natural mindset of the architects that participated in our study. As such, we expect the change viewpoint to facilitate architects logical thinking, while it still offers room for being creative.

7.8 Related Work

We categorize our related work to three areas: tool support, change in software architecture and requirements engineering communities, and taxonomy of sources of change. These areas are discussed in the following.

Tool support for change viewpoint. The change viewpoint needs to be assisted by tool support. The amount of knowledge elements and knowledge clusters could be overwhelming, which makes creation and analysis of change views ineffective or even impossible in absence of supporting tools. To the best of our knowledge no existing tool can fully support this type of viewpoint. In the following, we will describe the requirements for such a tool and further discuss the extent to which existing tools address those requirements.

Our proposed notation for the change viewpoint provides a schema for visualization that directs attention to change-related concerns. Existing tools can only partially support this schema. For instance, ontology-based knowledge visualization tools such as Aduna Cluster Maps [Fluit et al., 2005] can hierarchically illustrate knowledge elements in form of ontologies (e.g. the ontology presented in Figure 7.1). However, we need another tool (and another type of visualization) to determine the impact of change. This is because to illustrate the impact of change we need to visualize not only the distribution of knowledge elements in clusters but also how these elements are inter-dependent.

There are few tools that can capture the interdependencies, but are not fully suitable for change viewpoint. Lee and Kruchten (2008) propose a tool for visualizing architectural design decisions. This tool captures which decisions may be impacted by a change, using a graph. This tool does not support two features of the change viewpoint: (i) The change viewpoint visualizes all types of relevant knowledge, while the mentioned tool only visualizes design decisions, and (ii) The change viewpoint decreases complexity by scoping impact relations to a specific source of change, while this tool captures the impact relations for all sources of change in a global picture.

The tool for change viewpoint needs to support the iterative find & change approach presented in this chapter. To this end, it has to facilitate dynamic selection of sources of knowledge and automatic retrieval of the knowledge elements that
7.8. RELATED WORK

have to go through impact evaluation. A tool that partially supports the find activities is AREL (Tang et al., 2007). AREL models the causal relationships between architectural elements and decisions, and supports traceability among them. Similar to the find activity, AREL filters out unwanted traces when tracing the architectural elements. To sufficiently support the find activity, however, AREL has to be extended to perform filtering based on the ontology presented in Figure 7.1. In particular, the traces have to be scoped to (i) specific knowledge clusters (i.e., the classes in Figure 7.1) and (ii) the knowledge elements that are influenced by the change.

**Change in software architecture** In software engineering community according to a systematic literature review (Williams and Carver, 2010), 130 primary studies address change in architecture. Most of these approaches, however, are based on the traditional perception of architecture i.e., decomposition of a system into a number of interacting components. This implies that the change impact analysis is also limited to how components and their interactions change. Recent work in-the-field shows a shift in how software architecture is perceived: from architecture as the structure of an IT solution to architecture as a set of knowledge elements (Dingsøyr and van Vliet, 2009). Following this emerging perspective, our approach addresses change of any kind of architectural knowledge and as such dealing with change is a part of knowledge management.

**Change in requirements engineering** In requirements engineering community researchers provide various approaches for keeping problem space (goals, functional and non-functional requirements) and solution space (decisions and solution options) traceable and consistent under change (Lamsweerde, 2009; Sommerville and Sawyer, 1997). Although these approaches support moving between problem- and solution space (e.g., by guidelines or by automatic generation of trace links) they do not address the problem of supporting the reasoning process of architects when dealing with change. Providing such reasoning support is central to our change viewpoint.

**Taxonomy of sources of change** Several studies have presented classifications of sources of change. Many focus on classifying source code changes (e.g., Kim et al. (2005)), others identify organizational, management, and external factors as source of change (Madhavji, 1991). Our taxonomy in change viewpoint is the ontology presented in Figure 7.1. This taxonomy has emerged out of practice. For its construction, we used the knowledge types that we had found common among SOA migration projects (i.e., knowledge elements in core knowledge of Lean & Mean approach), and further refined them for the specific
CHAPTER 7. CHANGE VIEWPOINT: AN EXTENSION FOR DEALING WITH CHANGE

In this chapter we devised a viewpoint that helps architects reason better when dealing with change. We further showed the architects’ shared mental model, and defined the change viewpoint based on this model. The change viewpoint lends itself to architects’ natural way of thinking. According to Tang and van Vliet (Tang and van Vliet, 2012), architects use one or a combination of two thinking strategies: breadth-first and depth-first. Breadth-first thinking means exploring the whole design space broadly, whereas depth-first thinking means focusing on a particular problem area and addressing it in depth. Accordingly, the shared mental model presented in this chapter suggests that architects, in dealing with change, think in a combination of breadth-first and depth-first. Architects first explore and scope the sources of change broadly. Next, they select one change, go deep in both the problem and the solution space, and find the associated chain of changes. They often switch between depth- and breadth-first thinking. While scoping the impact of a change in depth, architects regularly switch to breadth-first and explore the context of change again, and vice versa. The change viewpoint supports the combination of breadth- and depth-first thinking, too, hence providing effective support for change management. This is exemplified in EHRS change scenarios. In Scenario 1 (see Figure 7.5), architects first identify the context of change in breadth, for scoping the effect of change in assumptions and business goals (breadth-first). In Scenario 2 (see Figure 7.2), however, they choose one source of change in particular and find the associated chain of changes (depth-first).

By focusing on a specific chain of changes, the change viewpoint identifies the relevant knowledge and context related to those changes. Therefore, for instance, it helps avoiding conflicting decisions. In particular, we expect that change viewpoint supports the two aspects of a sound reasoning (Tang, 2011): (i) explaining accurately the basic facts and assumptions, and (ii) making valid arguments. The first aspect is supported by explicit representation of the knowledge elements in Before pane of the change view. The second aspect, is facilitated by the change impact path, which highlights arguments such as “change in knowledge X might cause change in knowledge Y”.

In this study we have relied on input and feedback from architects of the EHRS to examine whether the change viewpoint supports their reasoning. The feedback, although informal, has been positive. The consensus was that the change viewpoint brings attention to what really matters in dealing with change, and that the iterative find & change approach helps their design thinking process.
Future work includes empirical validation of the effects of the change viewpoint in architects reasoning. This requires engagement of a broad community of architects of SOA migration projects. In addition, for broader adoption of the change viewpoint, the ontology of knowledge elements emerged out of the case study will be aligned with ISO/IEC 42010 ontology.
When Service Orientation was introduced as the solution for retaining and rehabilitating legacy assets, both researchers and practitioners proposed techniques, methods, and guidelines for SOA migration. With so much hype surrounding SOA, it is not surprising that academics and practitioners interpreted the concept differently and proposed different approaches to SOA migration. Accordingly, soon there were abundance of unique methods that were hard to compare and eventually adopt. Against this backdrop, in this thesis we investigated the academic and industrial approaches to SOA migration, generalized the practice of SOA migration in industry, and used such generalized practice to define the Lean & Mean approach to SOA migration. At the heart of Lean & Mean approach lies the core knowledge and activities that are common in SOA migration endeavors. As such, this approach provides the common ground to, among other things, guide practitioners (e.g., architects, designers, developers, project managers) in migration decision making. In this chapter, we revisit the research questions we listed in Chapter 1, summarize the answers our research provides, and discuss additional pointers to further research.

8.1 Contributions

In this thesis we have studied how SOA migration can be guided. As noted in Chapter 1, we have have elaborated this central research question (RQ) into four research sub-questions. Our answer to the central research question is therefore provided by the answers to the four sub-questions in the following four sub-sections.
CHAPTER 8. CONCLUSIONS

8.1.1 RQ-1: How to understand and characterize SOA migration?

In Chapter 2, we have presented the SOA migration framework (SOA-MF) for characterizing migration approaches. We showed that using the conceptual elements of SOA-MF one can characterize a migration approach based on processes it supports, artifacts included, activities carried out, and types of knowledge used and produced. Hence, as an answer to RQ-1, a general understanding of the migration approaches could be reached by mapping and positioning those approaches on SOA-MF.

Throughout this thesis, we used SOA-MF to characterize different academic and industrial SOA migration approaches. By representing those approaches in a unified manner, the framework assisted classification and comparison of such approaches. In comparing different approaches, thanks to expressiveness of SOA-MF, we were able to identify the meaningful categorization of migration approaches. For instance, the migration approaches that include conceptually similar activities, have graphically similar coverage patterns on SOA-MF. Such similarities helped categorizing migration approaches and further identifying SOA migration families in Chapter 3. In summary, by facilitating categorization of migration approaches, SOA-MF lends itself to achieve a general understanding of “what SOA migration entails”.

8.1.2 RQ-2: How is SOA migration perceived in academic research?

Chapter 3 presents a systematic literature review exploring and categorizing SOA migration approaches in academia. As a result, we provided a frame of reference for SOA migration categorizing the migration approaches with respect to two views of activity and knowledge. By studying the distribution of primary studies in the identified categories, we gained insight about how SOA migration is perceived in the field (i.e., RQ-2). In the following, we summarize our findings in this regard.

Firstly, we found that the dominant view on migration relates to using SOA as a technology enabler that facilitates integrating existing applications, rather than a paradigm that guides the development of business and software services from existing assets. Accordingly, the dominant approach for SOA migration pertains to reverse engineering existing legacy assets and modernizing them to services, rather than forward engineering added-value services while reusing pre-existing legacy assets. Furthermore, migration is mainly shaped and driven by the As-Is state, rather than the To-Be state, meaning that in trade-offs between leveraging legacy assets and ideal services, the heavier weight is given to the former rather
From the knowledge management perspective, academic approaches mainly focus on the explicit solution-related knowledge, and as such neglect the tacit knowledge residing in stakeholders minds. In the same vein, in knowledge conversions those approaches mainly rely on externalization and combination of knowledge, with little to no attention to socialization or internalization. Interestingly, our industrial studies (Chapter 6,7) revealed that knowledge conversions which include taking externalized knowledge and making it into individual tacit knowledge in the form of mental models or know-how are widely used in practice (i.e. internalization). In addition, transfer tacit knowledge to another person through observation or “learning by doing” (i.e. socialization) are very common in industrial practice.

8.1.3 RQ-3: How is SOA migration perceived and applied in industrial practice?

During our research, we have obtained a good overview of industrial SOA migration approaches. In Chapter 3, we explored the types of migration approaches employed by SOA solution providers in practice. As a result we found that unlike the majority of academic approaches, SOA migration in industry mostly neglects reverse engineering. Rather, migration follows a forward engineering approach initiated by identifying the ideal state (e.g. ideal business services), which is taken as a reference to extract and transform legacy elements to services. Understanding of legacy assets is achieved through personalization rather than reverse engineering techniques. Finally, industrial migration occurs mostly gradually in a number of increments, rather than in a sudden migration which is prevalent in academic approaches.

Our research further revealed that despite the diversity of participating enterprises, they all converged to the same, one, common SOA migration approach: all use similar activities, and similar knowledge input to carry out migration. This suggests that with experience enterprises mature toward a similar migration approach. In Chapter 5, we introduced such convergent approach. We further presented this industrial approach to a panel of experts, a tutorial with practitioners, and applied it into two industrial studies. This brought us with confidence that the industrial migration approach is conforming with “how migration is carried out in practice”.

143
8.1.4 RQ-4: How to guide SOA migration so that specific problems and needs of practice are addressed?

The research presented in this thesis has addressed a novel approach for guiding SOA migration, called the Lean & Mean approach. The essence of our approach is that the core elements that are repeating in SOA migration are separated from the project-specific ones and the advanced needs of certain projects are supported as an extension to the core. The core of Lean & Mean, emerged out of the results of the industrial survey, represents the knowledge elements and activities that were prevalent the industrial approaches. We applied the Lean & Mean in two industrial studies presented in Chapters 6 and 7. The usage implications of the Lean & Mean revealed the following ways in which this approach guides migration.

Firstly, Chapter 6 showed that the core guides determining “what knowledge has to be made available” and “what activities needs to be carried out”. Chapter 7 described how core knowledge guided us to readily focus on the key knowledge elements and leave out irrelevant elements; identify what knowledge needs to be externalized; and identify the gaps in our understanding. In short, in determining the required knowledge and activities core acts as the starting point, which helps to avoid starting from scratch.

Secondly, the extensions to the core support the more advanced needs of migration. In Chapter 7 we defined an extension that binds specific practices to the core activities. This extension guides reusing concrete solutions for carrying out a certain activity. An extension to the core knowledge was provided in Chapter 7 to guide practitioners to deal with changes during migration. In short, extensions guide practitioners in dealing with more context- or project-specific problems or needs.

Finally, in Chapter 6 and 7 we learned that the Lean & Mean approach guides practitioners reasoning in migration decision making. According to [2011] sound decision making includes three aspects: (i) identifying the important and relevant information that matter to decision making; (ii) exploring and developing problems; (iii) exploring the possible solution options and evaluation those options to make a decision. Related to the first aspect, both industrial studies (Chapter 6 and 7) showed that the core knowledge helps identifying and not overlooking relevant knowledge that shape and drive the migration. Related to second aspect, in Chapter 6 we found that the core helped practitioners to systematically identify the migration problems and structure them. We further found that the core knowledge and the core activities lend themselves in such problem-solving in different manners. Core knowledge helps to raise and identify migration issues, whereas core activities were used to further explore and structure those problems. In the same vein, Chapter 7 presented that change viewpoint (i.e., an extension
to Lean & Mean) structures the problem by bringing structure on relevant knowledge. With respect to the third aspect, exploring solution options, the extension for practice reuse in Chapter 6 supports exploring concrete solutions. Likewise, Chapter 7 presented chain of changes as different solution options for dealing with change. All in all, by supporting these three aspects the Lean & Mean approach guides practitioners reasoning in migration decision making process.

8.2 Future Research

Some ideas for future work have already been discussed in the previous chapters. In this section we conclude with a discussion of additional pointers to future research directions.

8.2.1 Enriching SOA Migration with Knowledge Management

This thesis has shown the combination of knowledge management and SOA migration is particularly helpful to provide necessary foundation for guiding migration decision making. Such combination led us to identify typical types of knowledge that shape and drive the migration, and address certain challenges and issues in SOA migration using knowledge management practices. While in this research the first steps towards enriching SOA migration with knowledge management practices are taken, further research is needed to fully bring the advancements of knowledge management discipline to SOA migration. In this regard, we see three future research directions linked to our results.

(i) Isolating and supporting the important migration drivers: In this thesis we have identified and isolated knowledge elements that shape and drive the migration and as such represent the essential information to work with. Each of the core knowledge elements addresses a different aspect of the migration. For instance, some knowledge address the “why” question that underpins a project (e.g. business goals motivating the migration), while other knowledge address “what” (e.g. what legacy assets) and “how” (e.g. how to transform). Some of those knowledge, however, are more important than others and need to be managed more carefully. For instance in Chapter 7 we showed that how modified assumptions have major effects in the migration projects. This is because assumptions, being knowledge addressing the “why” of the migration project, are more important than for instance a knowledge about a legacy component addressing “what”. Distinguishing the important knowledge elements in the specifics of SOA migration and providing suitable means for sharing, distributing, creating, capturing and understanding those knowledge can have significant influence in improvement
of SOA migration decision making. This constitutes the future research of this thesis.

(ii) Guiding selection of suitable knowledge management strategies: In our research we learned that there are many different knowledge management strategies that can benefit migration. Software companies who want to enrich their migration approaches with knowledge management practices need to decide on which approach is most useful for them. This can depend on wether the knowledge is of a kind, which can be shared between people, or is it of such a kind that some form of codification is needed. The size of the company and number of people will be important when choosing a strategy. Larger volumes will in most cases require a greater need for codification. However, the nature of the knowledge to be shared is also important in selecting the suitable knowledge management strategies. For instance, knowledge addressing migration drivers such as business goals or design decisions might be of a form, which makes codification more efficient than for person-to-person knowledge communication.

(iii) Improving communication in migration teams: The Lean & Mean approach developed in this thesis supports achieving a global view on what migration entails. This is in-line with one of the characteristics of learning organizations\footnote{Learning organizations is a prominent model on how knowledge is transferred or learned at an individual and organizational level proposed by Senge (1990).}: the ability of “systems thinking” - to see more than just parts of a system. In our work we created such global view by involving practitioners to develop a “shared mental model” (e.g. a shared mental model for how to deal with change). Such shared mental model is a common ground for “how to do migration” which showed being able to improve communication and teamwork in our research. Another way of improving communication in the migration team is to work on inserting other aspects of learning organizations with the approach for migration. Examples are “personal mastery”; that people make use of their creativity and abilities for certain tasks of migration, or “group learning”; to enhance communication and openness in the migration team.

8.2.2 Industry-relevant SOA Migration

In software engineering as an applied science, research should in principle serve the final purpose of being applied in practice. The extent to which this principle is supported by research, however, has been subject to debate for decades, and remains an unsolved problem. Different initiatives in software engineering (e.g., Semat, ICSE 2011 panel on “What Industry Wants from Research”) have a consensus on the following cause of such gap: what research proposes does not fit the fundamental problems, goals, strategies and weaknesses of practice. Not surprisingly, this research has revealed that such a theory-practice gap also exists
in the field of SOA migration. A significant step towards filling such a gap is our Lean & Mean approach as it is emerged out of practice of industrial migration, while it cares for extensions directly addressing important problems of practice. There are, however, many needs that industrial practice faces that are yet not supported by research. These needs, explained in-depth in this thesis, draw the following promising directions for industry-relevant research:

(i) Aligning risks, costs and value with migration approach: One of the issues that was repeatedly stated by the practitioners was the importance of risks and cost management in SOA migration decision making. We found risks and cost management to be in fact one of the main drivers of migration and influential on most decisions. This further confirms a recent interest toward risk-, cost- and value-aware methods (Poort and van Vliet 2011; Brown et al. 2010) that needs further research.

(ii) Providing decision making tools to support selection of migration approaches: In addition to the previous point, practitioners emphasized that industry needs tools to support planning and decision making for migration. For this purpose, the practitioners indicated as very beneficial to associate the core or extensions of Lean & Mean approach with typical risks, costs and pre-requisites. This calls for empirical research studies to associate the constituent elements of Lean & Mean approach with important decision criteria such as risks, costs and pre-requisites.
Samenvatting

Kennisgedreven migratie naar diensten

Service-oriëntatie beoogt reeds bestaande legacy elementen in vroegere staat te herstellen door hen onderdeel te maken van diensten die waarde toevoegen. Het migreren van dergelijke legacy elementen naar diensten die naadloos met moderne technologie kunnen samenwerken is een uitdagende en complexe taak. Deze complexiteit uit zich in een verschuiving van de wijze waarop de legacy elementen zijn ontwikkeld naar de wijze waarop wij diensten momenteel ontwerpen: van één groot systeem naar een verzameling van kleinere, uitwisselbare diensten die nog eigendom, noch onderdeel zijn van een “monolitisch” systeem. Diensten zijn helder gespecificeerde, los gekoppelde en samenhangende uitwisselbare elementen. Legacy elementen zijn daarentegen vaak sterk gekoppeld en niet samenhangend; voorts ondersteunen zij meerdere bedrijfsfunctionaliteiten. Deze inherente verschillen leiden ertoe dat het op diensten voorbereid maken van legacy elementen tot een complexe en veeleisende taak verwordt. Het resultaat van de migratie wordt, een dergelijke complexe omgeving indachtig, sterk beïnvloed door de manier waarop de migratie wordt begeleid. Dit proefschrift behandelt de wijze waarop migratie naar service-georiënteerde architectuur kan worden begeleid.

Het onderzoek dat beschreven is in dit proefschrift begon met het verkrijgen van begrip over wat migratie naar service-georiënteerde architectuur met zich meebrengt. Het verkrijgen van begrip in dit onderzoeksveld dat nog in opkomst is en aan scherpte kan winnen is moeilijk. Om dit begrip niettemin te creëren hebben wij een raamwerk ontwikkeld, SOA-MF genaamd, dat het mogelijk maakt verschillende migratie-aanpakken te karakteriseren. Het relateren en positioneren van verschillende migratie-aanpakken ten opzichte van SOA-MF heeft geleid tot het gewenste begrip.

Vervolgens hebben wij een systematisch literatuuronderzoek uitgevoerd om te begrijpen hoe migratie naar service-georiënteerde architectuur wordt gepositioneerd in het wetenschappelijke onderzoeksveld. Met behulp van het raamwerk hebben wij verschillende wetenschappelijke migratie-aanpakken gecategoriseerd. Dit heeft geresulteerd in een referentiekader voor migratie naar service-georiënteerde architectuur waarin de verschillende migratie-aanpakken zijn gecategoriseerd op basis van de in de aanpakken uitgevoerde activiteiten en gebruikte of geproduceerde kennisstructuren.

Om inzicht te verkrijgen in de wijze waarop migratie naar service-georiënteerde architectuur door de industrie wordt waargenomen, is een industriege richtte vragenlijst afgenomen bij zeven organisaties die hun klanten oplossingen bieden op het gebied van service-georiënteerde architectuur. Uit de resultaten bleek dat in de industrie gebruikte migratie-aanpakken aanzienlijk verschillen van wetenschappelijke migratie-aanpakken.
De verschillen tussen de industriële en wetenschappelijke aanpakken spoorden ons aan tot het verkrijgen van een beter begrip van de industriële aanpakken. Op basis van een verdere analyse van de resultaten van de industriegerichte vragenlijst hebben wij de praktijk van industriegerichte migratie gegeneraliseerd tot een Lean & Mean aanpak voor migratie naar service-georiënteerde architectuur.

Dit proefschrift presenteert de Lean & Mean aanpak als een generiek hulpmiddel om migratieprojecten te begeleiden en te besturen. De essentie van deze methode betreft het scheiden van kernelementen, die bij migratie naar service-georiënteerde architectuur terugkeren, van de projectspecifieke elementen en daarnaast het ondersteunen van geavanceerde behoeften van bepaalde projecten via een uitbreiding op die kern. Wij hebben de Lean & Mean aanpak voor migratie naar service-georiënteerde architectuur toegepast in twee industriële studies. De implicaties voor het gebruik van de Lean & Mean aanpak voor migratie naar service-georiënteerde architectuur bracht aan het licht op welke manieren deze aanpak de migratie kan begeleiden. Ten eerste helpen de kernelementen te bepalen “welke kennis beschikbaar gemaakt dient te worden” en “welke activiteiten uitgevoerd dienen te worden”. Ten tweede helpen de uitbreidingen op de kern de industriële professionals in het omgaan met meer context- of projectspecifieke problemen of behoeften. Op basis van de twee industriële studies hebben wij twee uitbreidingen op de Lean & Mean aanpak ontwikkeld: (i) Uitbreiding voor hergebruik van praktijken en (ii) Uitbreiding voor het omgaan met verandering. Op deze wijze hebben wij twee voornam problemen rond migratie naar service-georiënteerde architectuur uit de industriële praktijk geadresseerd: (i) hoe verkregen ervaringen te hergebruiken en (ii) hoe met wijzigingen gedurende migratie om te gaan. Ten slotte begeleidt de Lean & Mean aanpak industriële professionals bij het redeneren gedurende het nemen van beslissingen rondom de migratie.

Dit proefschrift toont ten slotte aan dat de combinatie van kennismanagement en migratie naar service-georiënteerde architectuur vooral behulpzaam is bij het bieden van de benodigde basis voor het begeleiden van het nemen van beslissingen rondom de migratie. Op basis van deze combinatie hebben wij kenmerkende kennisstypen geïdentificeerd die de migratie vormgeven en voortstuwen en die verschillende uitdagingen en kwesties in migratie naar service-georiënteerde architectuur met behulp van kennismanagement-praktijken adresseren. Hoewel met dit onderzoek de eerste stappen zijn ondernomen naar het verrijken van migratie naar service-georiënteerde architectuur met kennismanagement-praktijken, blijft verder onderzoek noodzakelijk om de binnen de kennismanagement-discipline geboekte resultaten volledig toe te passen op migratie naar service-georiënteerde architectuur.
Primary Studies of the Systematic Literature Review


Bibliography


Appendices
Overview of Primary Studies of the Systematic Literature Review

The overview of primary studies are presented in Table A.
## APPENDIX A. OVERVIEW OF PRIMARY STUDIES OF THE SYSTEMATIC LITERATURE REVIEW

<table>
<thead>
<tr>
<th>Study</th>
<th>Year</th>
<th>Validation</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>Ahn et al., 2008</td>
<td>2008</td>
</tr>
<tr>
<td>S2</td>
<td>Arcelli et al., 2008</td>
<td>2008</td>
</tr>
<tr>
<td>S3</td>
<td>Balis et al., 2008</td>
<td>2008</td>
</tr>
<tr>
<td>S4</td>
<td>Donnini and Forcato, 2002</td>
<td>2003</td>
</tr>
<tr>
<td>S5</td>
<td>Cantora et al., 2008, 2008a</td>
<td>2006, 2008</td>
</tr>
<tr>
<td>S6</td>
<td>dei Castino et al., 2009</td>
<td>2009</td>
</tr>
<tr>
<td>S7</td>
<td>Cetin et al., 2007a, b</td>
<td>2007</td>
</tr>
<tr>
<td>S8</td>
<td>Chen et al., 2004</td>
<td>2004</td>
</tr>
<tr>
<td>S9</td>
<td>Chen et al., 2005</td>
<td>2005</td>
</tr>
<tr>
<td>S10</td>
<td>Chen et al., 2006</td>
<td>2006</td>
</tr>
<tr>
<td>S11</td>
<td>Chuang et al., 2007</td>
<td>2007</td>
</tr>
<tr>
<td>S12</td>
<td>Cuadrado et al., 2008</td>
<td>2008</td>
</tr>
<tr>
<td>S13</td>
<td>El-Ramy et al., 2009</td>
<td>2009</td>
</tr>
<tr>
<td>S14</td>
<td>Gonzalez et al., 2009</td>
<td>2009</td>
</tr>
<tr>
<td>S15</td>
<td>Giro et al., 2009</td>
<td>2009</td>
</tr>
<tr>
<td>S16</td>
<td>Haidar and Aklallah, 2008</td>
<td>2008</td>
</tr>
<tr>
<td>S17</td>
<td>Holm et al., 2008</td>
<td>2008</td>
</tr>
<tr>
<td>S18</td>
<td>Huang et al., 2008</td>
<td>2008</td>
</tr>
<tr>
<td>S19</td>
<td>Huang, 2009</td>
<td>2009</td>
</tr>
<tr>
<td>S20</td>
<td>Hutchinson et al., 2007</td>
<td>2007</td>
</tr>
<tr>
<td>S21</td>
<td>Iik et al., 2008</td>
<td>2008</td>
</tr>
<tr>
<td>S22</td>
<td>Kamran and Srivastava, 2008</td>
<td>2008</td>
</tr>
<tr>
<td>S23</td>
<td>Lavay et al., 2004</td>
<td>2004</td>
</tr>
<tr>
<td>S24</td>
<td>Lee et al., 2004</td>
<td>2004</td>
</tr>
<tr>
<td>S26</td>
<td>Li et al., 2007</td>
<td>2007</td>
</tr>
<tr>
<td>S27</td>
<td>Li and Tahvildari, 2008</td>
<td>2008</td>
</tr>
<tr>
<td>S28</td>
<td>Liu et al., 2009</td>
<td>2009</td>
</tr>
<tr>
<td>S29</td>
<td>Liu et al., 2009</td>
<td>2009</td>
</tr>
<tr>
<td>S30</td>
<td>Liu et al., 2008</td>
<td>2008</td>
</tr>
<tr>
<td>S31</td>
<td>Lucu et al., 2008</td>
<td>2008</td>
</tr>
<tr>
<td>S32</td>
<td>Matos, 2008</td>
<td>2008</td>
</tr>
<tr>
<td>S33</td>
<td>Nguyen et al., 2009b</td>
<td>2009</td>
</tr>
<tr>
<td>S34</td>
<td>O'Hagan et al., 2005</td>
<td>2005</td>
</tr>
<tr>
<td>S35</td>
<td>Pant and Barrett, 2006</td>
<td>2006</td>
</tr>
<tr>
<td>S36</td>
<td>Sindhgatta and Ponnalagu, 2008</td>
<td>2008</td>
</tr>
<tr>
<td>S37</td>
<td>Sneed, 2006</td>
<td>2006</td>
</tr>
<tr>
<td>S38</td>
<td>Umar and Zordan, 2009</td>
<td>2009</td>
</tr>
<tr>
<td>S39</td>
<td>Varga et al., 2004</td>
<td>2004</td>
</tr>
<tr>
<td>S40</td>
<td>Xu et al., 2005</td>
<td>2005</td>
</tr>
<tr>
<td>S42</td>
<td>Zhang et al., 2008</td>
<td>2008</td>
</tr>
<tr>
<td>S43</td>
<td>Zou and Kostoglou, 2004</td>
<td>2000</td>
</tr>
<tr>
<td>S44</td>
<td>Iocola, 2007</td>
<td>2007</td>
</tr>
<tr>
<td>S45</td>
<td>Alahmari et al., 2010</td>
<td>2010</td>
</tr>
<tr>
<td>S46</td>
<td>Bao et al., 2010</td>
<td>2010</td>
</tr>
<tr>
<td>S47</td>
<td>Hassamieh et al., 2010</td>
<td>2010</td>
</tr>
<tr>
<td>S48</td>
<td>Chenghao et al., 2010</td>
<td>2010</td>
</tr>
<tr>
<td>S49</td>
<td>Chung et al., 2009</td>
<td>2009</td>
</tr>
<tr>
<td>S50</td>
<td>Djelouai et al., 2009</td>
<td>2009</td>
</tr>
<tr>
<td>S51</td>
<td>Koschel et al., 2009</td>
<td>2009</td>
</tr>
<tr>
<td>S52</td>
<td>Koun et al., 2010</td>
<td>2010</td>
</tr>
<tr>
<td>S53</td>
<td>Marchetto and Ricci, 2009</td>
<td>2009</td>
</tr>
<tr>
<td>S54</td>
<td>Vatnan, 2010</td>
<td>2010</td>
</tr>
<tr>
<td>S55</td>
<td>Mulcahy et al., 2010</td>
<td>2010</td>
</tr>
<tr>
<td>S56</td>
<td>Nakamura et al., 2009</td>
<td>2009</td>
</tr>
<tr>
<td>S57</td>
<td>Nasi et al., 2010</td>
<td>2010</td>
</tr>
<tr>
<td>S58</td>
<td>Proctor and Kerďouh, 2010</td>
<td>2010</td>
</tr>
<tr>
<td>S59</td>
<td>Wang et al., 2010</td>
<td>2010</td>
</tr>
<tr>
<td>S60</td>
<td>Zhang et al., 2010a, b</td>
<td>2010</td>
</tr>
<tr>
<td>S61</td>
<td>Zhou et al., 2010</td>
<td>2010</td>
</tr>
</tbody>
</table>
Overview of Interview Survey Results

The interview guid and the overview of interview survey results (shown in Table B.1) are described in the following.

Interview Guide

1. SOA Migration Project
   In which projects did you migrate your pre-existing business functionalities to SOA? Lets choose one of these projects (maybe the most recent one).
   
   (a) Can you describe your role within this project?
   (b) Can you describe the context and domain of this project?
   (c) Was the existing system cross-organizational or internal to the company?

2. SOA Migration Approach

   (a) Candidate Service Identification in the Pre-existing Legacy System
   What were the legacy assets that were identified as a candidate to be reused in (new) service-based system?
   
   i. What types of assets (e.g. component, code, infrastructure, business processes) of the legacy system were (re)-used in the SOA based systems?
   ii. What properties or criteria made this legacy element a candidate to be transformed to services?
   iii. Generally speaking, what type of assets do you find useful/feasible to be migrated to service based systems?
Appendix B. Overview of Interview Survey Results

(b) Understanding the Pre-existing Legacy Assets

In order to qualify existing assets as candidate services, how far did you need to go into understanding of the existing legacy system?

i. What types of information helped you to identify these candidate services? (e.g. code, functionality of existing system, existing structure of the system, high level architecture, requirements)

ii. From what sources did you obtain this information about the legacy system?

iii. What activities or technique did you use to come up to this understanding (if any)? What were the inputs and outputs to these activities?

iv. Were there any key quality attributes in the legacy system that you wanted to preserve into new service based system?

v. Can you think of any information or knowledge about the legacy system that was not available, which could have helped you with SOA migration task? Can you give examples?

vi. In your perception, does capturing the design decisions made (in the past) during the development of the legacy system, help the SOA migration process?

(c) Understanding the Target Services

In order to qualify existing assets as candidate services, how far did you need to go into understanding of the target service based environment?

i. What properties/features of the target SOA supported the selection of the pre-existing legacy elements as candidate services? (e.g. SOA principles level of granularity of services, constraints may be posed by SOA infrastructure, different types of services)

(d) Transformation to Services

Once the legacy elements as candidate services, were identified, how were they transformed to services?

i. What techniques are used to transform the extracted legacy elements to SOA?

ii. Did you have set of transformations at different levels of abstraction? (e.g. module/components level, architecture level or business processes level)?

iii. Do you reuse certain solutions (e.g. transformation patterns) possibly in different migration projects for transforming legacy elements to SOA?
iv. Do you think having a set of ready to use solutions would be helpful for migration to SOA? Does it work in practice?

(e) Development of New Service-based System

What was the process for design and development of the service based system based on the new requirements or goals (if any)?

i. Was the design of new services (out of new requirements) shaped by the migrated services and vice versa?

ii. Did the migrated services finally act as service (as new service)?

As an example, were there any challenges in service compositions constituting new and migrated services?

Interview Survey Results

The results of the interview survey are summarized in Table B.1.
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>RVS</td>
<td>Figure 4.1.II(a) bowl. understanding components functionalities, legacy architecture and existing functionalities by asking the stakeholders and existing documentation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BS.</td>
<td>reusability, granularity of services, cohesion</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CXB</td>
<td>Figure 4.1.II(b) bowl. understanding the legacy interfaces, data model and data flows via a set of interviews and workshops and reading the code</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BS.</td>
<td>loose coupling, granularity of services, cohesion</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SVD</td>
<td>Figure 4.1.II(b) bowl. understanding the business processes, data models and interactions using workshops, user manuals and existing documents</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BS.</td>
<td>none</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JXM</td>
<td>Figure 4.1.II(b) bowl. understanding the code, architecture and functionalities by asking the stakeholders, and existing documentation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BS.</td>
<td>reusability, granularity of services</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PXB</td>
<td>Figure 4.1.II(b) bowl. understanding the business and data architecture, business processes and system interfaces by asking the stakeholders and existing documentation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BS.</td>
<td>cohesion</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DXL</td>
<td>Figure 4.1.II(b) bowl. understanding existing data, transactions and messages by reading the code, interviews and prototypes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BS.</td>
<td>loose coupling, granularity of services, modifiability, flexibility</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GWH</td>
<td>Figure 4.1.II(b) bowl. understanding business processes, functionalities, architecture, and interfaces using a set of semi-structured interviews</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BS., SS.</td>
<td>loose coupling, granularity of services, cohesion, auditability, agility</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RXB</td>
<td>Figure 4.1.II(a) bowl. understanding business domain, data, legacy technology, and system interfaces using a set of brainstorming meetings, interviews and documents</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BS., SS.</td>
<td>reusability, ownership, service granularity</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JWV</td>
<td>Figure 4.1.II(b) bowl. understanding business domain and BS. business service software service</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RXB</td>
<td>Figure 4.1.II(a) bowl. understanding business domain and BS. business service software service</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CXB</td>
<td>Figure 4.1.II(b) bowl. understanding business domain and BS. business service software service</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GWH</td>
<td>Figure 4.1.II(b) bowl. understanding business domain and BS. business service software service</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JWV</td>
<td>Figure 4.1.II(b) bowl. understanding business domain and BS. business service software service</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Technical Action Research Study Design

The detailed study design of the technical action research presented in Chapter is illustrated in Figure C.1 and described in the following.

Figure C.1: Technical Action Research Study Design: Detailed

**Engineering Cycle 1**

**Problem investigation.** In this phase we came to an understanding of the stakeholder participating in our study, context and needs of Logica through a number of interviews. Interviews were used in order to collect qualitative data about the migration projects. Three stakeholders involved in different aspects of migration were interviewed: the chief architect, a business analyst and a sector...
manager. In order to remain as open as possible in identifying practitioners problems, we chose open interviews.

**Treatment design.** In this phase we decided on the treatment plan. As noted, the plan was to first recover the Lean & Mean approach of a past project and later use this approach in an on-going project. Accordingly, together with the stakeholders we chose one past and one on-going data migration project. These two projects are introduced in § 6.2.

**Treatment execution and evaluation.** Within this phase we executed the plan. This phase initiated the research cycle.

**Research Cycle 1**

**Research design investigation** In this phase we defined the *research questions*, *unit of study* and *data analysis method.* **Research questions:**

- RQ-I. What would be the effect of having the Lean & Mean migration approach in place?
- RQ-II. Is the Lean & Mean migration approach generalizable?

**Unit of study.** The data population of this study constituted the migration projects in the Logica. Data was collected by means of existing documentation, researcher diary, and audio recording and transcription of the interviews and feedback sessions. The interviews conducted were semi-structured with open-ended questions, which allow interviewers to ask follow on questions when necessary. This flexibility allows the interviewer to ask the interviewees detailed questions about their specific areas of expertise.

**Data analysis method.** To typify the migration approaches, we analyzed the interview and feedback session transcripts. The analysis resulted in the characterization of the migration approaches based on the activities carried out and the knowledge used and produced. To carry out the analysis systematically, we devised a coding procedure. For creating codes we followed the suggestion of Miles and Huberman (Miles and Huberman, 1994) and started with an initial set of codes, called ‘start-list’, and we refined it during the analysis. Our start-list stems from the Lean & Mean-G introduced in § 6.3. The coding procedure is described in the following:

1. *Filling in activities involved in migration:* coding the activities involved in the migration. Filling in refers to creating new codes as insights occur during the analysis.

2. *Surfacing knowledge elements:* identifying the input/output knowledge elements of migration activities as well as the conversions among them. This
way, the knowledge elements that are used and/or produced throughout the migration approach are extracted.

**Validate research design.** The research design was evaluated to get an impression of its external and internal validity. The details are discussed in § 6.8.

**Research execution.** This phase initiated the next engineering cycles each conducting a different case study (i.e. Case I and Case II).

**Analysis results** Using the coding procedure introduced above, we mapped each of the migration approaches of Case I and Case II on Lean & Mean-G. We further analyzed the feedback sessions and the workshop. The analysis resulted in answering our research questions.

**Engineering Cycle 2**

The main goal of this cycle was to recover the migration approach of Case I and organize it into a Lean & Mean approach.

**Problem investigation.** The main goal of this phase was to understand the context of the project through interviews and existing documents. For interviews, three stakeholders who had holistic view on the project were selected: the chief architect, the project manager and a business analyst.

**Treatment design.** Using our coding procedure we first mapped the approach of Case I on the Lean & Mean-G.

**Treatment execution and evaluation.** At this phase the Lean & Mean approach of Case I was evaluated within the feedback sessions. During the feedback sessions each of the stakeholders were asked to identify knowledge or activities that are core and are missing in the mapping. For example, for each of the activities they were asked:

- Should this activity occur in all data migration projects?

- What knowledge is used in or produced by this activity?

The purposes of feedback sessions were twofold: (i) evaluate whether the core is really prevalent in data migration projects, and (ii) identify the special needs of Logica that pinpoint the needed extensions to the core. In the feedback sessions we observed and asked exploratory questions to provoke reactions that differed from our expectations. The feedback sessions resulted in the first version of Lean & Mean approach of data migration projects, called Lean & Mean-DM. The Lean & Mean-DM was further challenged and refined in the engineering cycle 3. § 6.5 describes the Lean & Mean-DM.
Engineering Cycle 3

Our goal in the third engineering cycle was to identify the benefits of having the core and extensions of the Lean & Mean-DM when determining the approach of the new project, i.e., Case II.

**Problem investigation.** The first stage in this cycle constituted understanding the context of Case II. Similar to Case I, such understanding was gained through interviews and existing documents.

**Treatment design.** At this phase to determine the migration approach for Case II we organized workshops with project stakeholders: the project manager, the chief architect and a business analyst. These workshops resulted in defining the Lean & Mean approach of the project. Throughout these workshops we observed the ways in which the Lean & Mean-DM helped stakeholders in determining the Case II’s migration approach.

**Treatment execution and evaluation.** The observed benefits were further discussed with the stakeholders in the feedback sessions. Lean & Mean-DM was further revised based on the stakeholders feedback.
Documenting Design Space Using a Template

Table D.1 exemplifies the representation of a design issue, its regarding solution options, and decisions using the template proposed in (Gu et al., 2010).
# APPENDIX D. DOCUMENTING DESIGN SPACE USING A TEMPLATE

## Table D.1: Design Space of Issue D5

<table>
<thead>
<tr>
<th>Design issue</th>
<th>D5 What architectural style fits best the coordination of information exchange in EHRS?</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Context</strong></td>
<td>The EHRS is a service-based system that enables exchange of information to retrieve certain data about patients. In doing so, different healthcare providers need to communicate and eventually provide relevant data. To effectively coordinate those data exchanges while adhering to legislation, selection of the best fitting architectural style is essential.</td>
</tr>
<tr>
<td><strong>Selection Criteria</strong></td>
<td>Cr1: Confidentiality, Cr2: Integrity, Cr3: Scalability, Cr4: Interoperability.</td>
</tr>
<tr>
<td><strong>Options</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Identifier</strong></td>
<td>D5-Opt1 Nation-wide centralized architecture</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>This architecture is characterized by centralized HIB that coordinates the exchange of patient data between healthcare providers, such that the HIB controls the access to patient data using a national authorization protocol (Aut), record locator (Rec), access log (Log) and identification &amp; authorization service.</td>
</tr>
<tr>
<td><strong>Status</strong></td>
<td>This option has been decided</td>
</tr>
<tr>
<td><strong>Evaluation</strong></td>
<td>Cr1: Confidentiality is not negatively affected as patients data is not stored centrally, but instead remain stored in the information system of the healthcare providers and EHRS only keeps and index to patients data. Cr2: Because of its central role in authentication and authorization and the exchange of patient records, the EHRS could be an attractive target for attackers. Consequently, negatively affects integrity. Cr3: The nature of centralized architecture increases the possibility of arising bottleneck problem. While the response of one message includes a long history of medication, responding rest of messages could be postponed for a long while by the centralized system. Facing bottleneck problem represents the system is not able to manage incidental peak demands. Cr4: The centralized architecture has the advantage that the HIB could bridge a version difference between XIS exchanging patient data.</td>
</tr>
<tr>
<td><strong>Rationale</strong></td>
<td>This option was selected as it better supports interoperability which was an important problem in beginning of the project due to lack of standardization.</td>
</tr>
<tr>
<td><strong>Identifier</strong></td>
<td>D5-Opt2 Nation-wide distributed architecture</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>This architecture is characterized by decentralized exchange of information in which healthcare providers directly communicate. For authorization the regarding services are consumed directly by the healthcare providers.</td>
</tr>
<tr>
<td><strong>Status</strong></td>
<td>This option has been rejected</td>
</tr>
<tr>
<td><strong>Evaluation</strong></td>
<td>Cr1: Confidentiality is not negatively affected as patients data is not stored centrally, but instead remain stored in the information system of the healthcare providers and EHRS only keeps and index to patients data. Cr2: Since the entire healthcare data is directly exchanged between healthcare providers, the possibility of attacks are lower. Cr3: If part of the common service components are reused for other purposes, the capacity demand growth may certainly differ per service component. Cr4: The distributed services architecture scores low regarding version compatibility as XIS’es exchange data directly and they should support solutions for version compatibility themselves.</td>
</tr>
<tr>
<td><strong>Rationale</strong></td>
<td>The main reason that this option is rejected lies in lack of support for interoperability as data used by different providers are heterogeneous.</td>
</tr>
</tbody>
</table>

**Legend:** D#: design issue; Cr#: criteria; D#-Opt#: architectural design option.
SIKS Dissertation Series

1998
1998-1 Johan van den Akker (CWI)
   DEGAS - An Active, Temporal Database of Autonomous Objects
1998-2 Floris Wiesman (UM)
   Information Retrieval by Graphically Browsing Meta-Information
1998-3 Ans Steuten (TUD)
   A Contribution to the Linguistic Analysis of Business Conversations
   within the Language/Action Perspective
1998-4 Dennis Breuker (UM)
   Memory versus Search in Games
1998-5 E.W.Oskamp (RUL)
   Computerondersteuning bij Straftoemeting

1999
1999-1 Mark Sloof (VU)
   Physiology of Quality Change Modelling;
   Automated modelling of Quality Change of Agricultural Products
1999-2 Rob Potharst (EUR)
   Classification using decision trees and neural nets
1999-3 Don Beal (UM)
   The Nature of Minimax Search
1999-4 Jacques Penders (UM)
   The practical Art of Moving Physical Objects
1999-5 Aldo de Moor (KUB)
   Empowering Communities: A Method for the Legitimate User-Driven
   Specification of Network Information Systems
1999-6 Nick J.E. Wijngaards (VU)
   Re-design of compositional systems
1999-7 David Spelt (UT)
   Verification support for object database design
1999-8 Jacques H.J. Lenting (UM)
   Informed Gambling: Conception and Analysis of a Multi-Agent
   Mechanism for Discrete Reallocation.

2000
2000-1 Frank Niessink (VU)
   Perspectives on Improving Software Maintenance
2000-2 Koen Holtman (TUE)
   Prototyping of CMS Storage Management
2000-3 Carolien M.T. Metselaar (UVA)
   Sociaal-organisatorische gevolgen van kennis technologie;
   een procesbenadering en actopperceptief.
2000-4 Geert de Haan (VU)
ETAG, A Formal Model of Competence Knowledge for User Interface Design

2000-5 Ruud van der Pol (UM)
Knowledge-based Query Formulation in Information Retrieval.

2000-6 Rogier van Eijk (UU)
Programming Languages for Agent Communication

2000-7 Niels Peek (UU)
Decision-theoretic Planning of Clinical Patient Management

2000-8 Veerle Coup (EUR)
Sensitivity Analysis of Decision-Theoretic Networks

2000-9 Florian Waas (CWI)
Principles of Probabilistic Query Optimization

2000-10 Niels Nes (CWI)
Image Database Management System Design Considerations, Algorithms and Architecture

2000-11 Jonas Karlsson (CWI)
Scalable Distributed Data Structures for Database Management

2001

2001-1 Silja Renooij (UU)
Qualitative Approaches to Quantifying Probabilistic Networks

2001-2 Koen Hindriks (UU)
Agent Programming Languages: Programming with Mental Models

2001-3 Maarten van Someren (UvA)
Learning as problem solving

2001-4 Evgeni Smirnov (UM)
Conjunctive and Disjunctive Version Spaces with Instance-Based Boundary Sets

2001-5 Jacco van Ossenbruggen (VU)
Processing Structured Hypermedia: A Matter of Style

2001-6 Martijn van Welie (VU)
Task-based User Interface Design

2001-7 Bastiaan Schonhage (VU)
Diva: Architectural Perspectives on Information Visualization

2001-8 Pascal van Eck (VU)
A Compositional Semantic Structure for Multi-Agent Systems Dynamics.

2001-9 Pieter Jan ’t Hoen (RUL)
Towards Distributed Development of Large Object-Oriented Models, Views of Packages as Classes

2001-10 Maarten Sierhuis (UvA)
Modeling and Simulating Work Practice
BRAHMS: a multiagent modeling and simulation language
for work practice analysis and design

2001-11 Tom M. van Engers (VUA)
Knowledge Management:
The Role of Mental Models in Business Systems Design

2002
2002-01 Nico Lassing (VU)
Architecture-Level Modifiability Analysis

2002-02 Roelof van Zwol (UT)
Modelling and searching web-based document collections

2002-03 Henk Ernst Blok (UT)
Database Optimization Aspects for Information Retrieval

2002-04 Juan Roberto Castelo Valdueza (UU)
The Discrete Acyclic Digraph Markov Model in Data Mining

2002-05 Radu Serban (VU)
The Private Cyberspace Modeling Electronic Environments
inhabited by Privacy-concerned Agents

2002-06 Laurens Mommers (UL)
Applied legal epistemology;
Building a knowledge-based ontology of the legal domain

2002-07 Peter Boncz (CWI)
Monet: A Next-Generation DBMS Kernel For Query-Intensive Applications

2002-08 Jaap Gordijn (VU)
Value Based Requirements Engineering: Exploring Innovative
E-Commerce Ideas

2002-09 Willem-Jan van den Heuvel(KUB)
Integrating Modern Business Applications with Objectified Legacy Systems

2002-10 Brian Sheppard (UM)
Towards Perfect Play of Scrabble

2002-11 Wouter C.A. Wijngaards (VU)
Agent Based Modelling of Dynamics: Biological and Organisational Applications

2002-12 Albrecht Schmidt (Uva)
Processing XML in Database Systems

2002-13 Hongjing Wu (TUE)
A Reference Architecture for Adaptive Hypermedia Applications

2002-14 Wieke de Vries (UU)
Agent Interaction: Abstract Approaches to Modelling, Programming and
Verifying Multi-Agent Systems

2002-15 Rik Eshuis (UT)
Semantics and Verification of UML Activity Diagrams for Workflow Modelling

2002-16 Pieter van Langen (VU)
The Anatomy of Design: Foundations, Models and Applications

2002-17 Stefan Manegold (UVA)
Understanding, Modeling, and Improving Main-Memory Database Performance

2003
Ontology-Based Information Sharing in Weakly Structured Environments

Modal Action Logics for Reasoning About Reactive Systems

Human-Computer Interaction and Presence in Virtual Reality Exposure Therapy

Content-Based Video Retrieval Supported by Database Technology

Causation in Artificial Intelligence and Law - A modelling approach

Development and specification of virtual environments

Formal Explorations of Knowledge Intensive Tasks

Repair Based Scheduling

The resolution of visually guided behaviour

Electronic Business Negotiation: Some experimental studies on the interaction between medium, innovation context and culture

Reasoning under Uncertainty in Natural Language Dialogue using Bayesian Networks

Dutch speech recognition in multimedia information retrieval

Nosce Hostem - Searching with Opponent Models

Freezing Language: Conceptualisation Processes across ICT-Supported Organisations

Plan Merging in Multi-Agent Systems

Feature Grammar Systems - Incremental Maintenance of Indexes to Digital Media Warehouses

Extensions of Statecharts with Probability, Time, and Stochastic Timing

Learning Search Decisions

Virginia Dignum (UU)
A Model for Organizational Interaction: Based on Agents, Founded in Logic

2004-02 Lai Xu (UvT)
Monitoring Multi-party Contracts for E-business

2004-03 Perry Groot (VU)
A Theoretical and Empirical Analysis of Approximation in Symbolic Problem Solving

2004-04 Chris van Aart (UVA)
Organizational Principles for Multi-Agent Architectures

2004-05 Viara Popova (EUR)
Knowledge discovery and monotonicity

2004-06 Bart-Jan Hommes (TUD)
The Evaluation of Business Process Modeling Techniques

2004-07 Elise Boltjes (UM)
Voorbeeldig onderwijs; voorbeeldgestuurd onderwijs, een opstap naar abstract denken, vooral voor meisjes

2004-08 Joop Verbeek(UM)
Folijie en de Nieuwe Internationale Informatiemarkt, Grensregionale politiege gegevensuitwisseling en digitale expertise

2004-09 Martin Caminada (VU)
For the Sake of the Argument; explorations into argument-based reasoning

2004-10 Suzanne Kabel (UVA)
Knowledge-rich indexing of learning-objects

2004-11 Michel Klein (VU)
Change Management for Distributed Ontologies

2004-12 The Duy Bui (UT)
Creating emotions and facial expressions for embodied agents

2004-13 Wojciech Jamroga (UT)
Using Multiple Models of Reality: On Agents who Know how to Play

2004-14 Paul Harrenstein (UU)
Logic in Conflict. Logical Explorations in Strategic Equilibrium

2004-15 Arno Knobbe (UU)
Multi-Relational Data Mining

2004-16 Federico Divina (VU)
Hybrid Genetic Relational Search for Inductive Learning

2004-17 Mark Winands (UM)
Informed Search in Complex Games

2004-18 Vania Bessa Machado (UvA)
Supporting the Construction of Qualitative Knowledge Models

2004-19 Thijs Westerveld (UT)
Using generative probabilistic models for multimedia retrieval

2004-20 Madelon Evers (Nyenrode)
Learning from Design: facilitating multidisciplinary design teams
2005

2005-01 Floor Verdenius (UVA)
Methodological Aspects of Designing Induction-Based Applications

2005-02 Erik van der Werf (UM)
AI techniques for the game of Go

2005-03 Franc Grootjen (RUN)
A Pragmatic Approach to the Conceptualisation of Language

2005-04 Nirvana Meratnia (UT)
Towards Database Support for Moving Object data

2005-05 Gabriel Infante-Lopez (UVA)
Two-Level Probabilistic Grammars for Natural Language Parsing

2005-06 Pieter Spronck (UM)
Adaptive Game AI

2005-07 Flavius Frasincar (TUE)
Hypermedia Presentation Generation for Semantic Web Information Systems

2005-08 Richard Vdovjak (TUE)
A Model-driven Approach for Building Distributed Ontology-based Web Applications

2005-09 Jeen Broekstra (VU)
Storage, Querying and Inferencing for Semantic Web Languages

2005-10 Anders Bouwer (UVA)
Explaining Behaviour: Using Qualitative Simulation in Interactive Learning Environments

2005-11 Elth Ogston (VU)
Agent Based Matchmaking and Clustering - A Decentralized Approach to Search

2005-12 Csaba Boer (EUR)
Distributed Simulation in Industry

2005-13 Fred Hamburg (UL)
Een Computermodel voor het Ondersteunen van Euthanasiebeslissingen

2005-14 Borys Omelayenko (VU)
Web-Service configuration on the Semantic Web; Exploring how semantics meets pragmatics

2005-15 Tibor Bosse (VU)
Analysis of the Dynamics of Cognitive Processes

2005-16 Joris Graaumans (UU)
Usability of XML Query Languages

2005-17 Boris Shishkov (TUD)
Software Specification Based on Re-usable Business Components

2005-18 Danielle Sent (UU)
Test-selection strategies for probabilistic networks

2005-19 Michel van Dartel (UM)
Situated Representation

2005-20 Cristina Coteanu (UL)
Cyber Consumer Law, State of the Art and Perspectives

2005-21 Wijnand Derks (UT)
Improving Concurrency and Recovery in Database Systems by
Exploiting Application Semantics

2006

2006-01 Samuil Angelov (TUE)
Foundations of B2B Electronic Contracting

2006-02 Cristina Chisalita (VU)
Contextual issues in the design and use of information technology in organizations

2006-03 Noor Christoph (UVA)
The role of metacognitive skills in learning to solve problems

2006-04 Marta Sabou (VU)
Building Web Service Ontologies

2006-05 Cees Pierik (UU)
Validation Techniques for Object-Oriented Proof Outlines

2006-06 Ziv Baida (VU)
Software-aided Service Bundling - Intelligent Methods & Tools
for Graphical Service Modeling

2006-07 Marko Smiljanic (UT)
XML schema matching – balancing efficiency and effectiveness by means of clustering

2006-08 Eelco Herder (UT)
Forward, Back and Home Again - Analyzing User Behavior on the Web

2006-09 Mohamed Wahdan (UM)
Automatic Formulation of the Auditor’s Opinion

2006-10 Ronny Siebes (VU)
Semantic Routing in Peer-to-Peer Systems

2006-11 Joeri van Ruth (UT)
Flattening Queries over Nested Data Types

2006-12 Bert Bongers (VU)
Interactivation - Towards an e-cology of people, our technological environment, and the arts

2006-13 Henk-Jan Lebbink (UU)
Dialogue and Decision Games for Information Exchanging Agents

2006-14 Johan Hoorn (VU)
Software Requirements: Update, Upgrade, Redesign -
towards a Theory of Requirements Change

2006-15 Rainer Malik (UU)
CONAN: Text Mining in the Biomedical Domain

2006-16 Carsten Riggelsen (UU)
Approximation Methods for Efficient Learning of Bayesian Networks

2006-17 Stacey Nagata (UU)
User Assistance for Multitasking with Interruptions on a Mobile Device
2006-18 Valentin Zhizhkun (UVA)
Graph transformation for Natural Language Processing

2006-19 Birna van Riemsdijk (UU)
Cognitive Agent Programming: A Semantic Approach

2006-20 Marina Velikova (UvT)
Monotone models for prediction in data mining

2006-21 Bas van Gils (RUN)
Aptness on the Web

2006-22 Paul de Vrieze (RUN)
Fundamentals of Adaptive Personalisation

2006-23 Ion Juvina (UU)
Development of Cognitive Model for Navigating on the Web

2006-24 Laura Hollink (VU)
Semantic Annotation for Retrieval of Visual Resources

2006-25 Madalina Drugan (UU)
Conditional log-likelihood MDL and Evolutionary MCMC

2006-26 Vojkan Mihajlovic (UT)
Score Region Algebra: A Flexible Framework for Structured Information Retrieval

2006-27 Stefano Bocconi (CWI)
Vox Populi: generating video documentaries from semantically annotated media repositories

2006-28 Borkur Sigurbjornsson (UVA)
Focused Information Access using XML Element Retrieval

2007

2007-01 Kees Leune (UvT)
Access Control and Service-Oriented Architectures

2007-02 Wouter Teepe (RUG)
Reconciling Information Exchange and Confidentiality: A Formal Approach

2007-03 Peter Mika (VU)
Social Networks and the Semantic Web

2007-04 Jurriana van Diggelen (UU)
Achieving Semantic Interoperability in Multi-agent Systems: a dialogue-based approach

2007-05 Bart Schermer (UL)
Software Agents, Surveillance, and the Right to Privacy: a Legislative Framework for Agent-enabled Surveillance

2007-06 Gilad Mishne (UVA)
Applied Text Analytics for Blogs

2007-07 Natasa Jovanovic' (UT)
To Whom It May Concern - Addressee Identification in Face-to-Face Meetings

2007-08 Mark Hoogendoorn (VU)
Modeling of Change in Multi-Agent Organizations
2007-09 David Mobach (VU)
   Agent-Based Mediated Service Negotiation

2007-10 Huib Aldewereld (UU)
   Autonomy vs. Conformity: an Institutional Perspective on Norms and Protocols

2007-11 Natalia Stash (TUE)
   Incorporating Cognitive/Learning Styles in a General-Purpose Adaptive Hypermedia System

2007-12 Marcel van Gerven (RUN)
   Bayesian Networks for Clinical Decision Support: A Rational Approach to Dynamic Decision-Making under Uncertainty

2007-13 Rutger Rienks (UT)
   Meetings in Smart Environments; Implications of Progressing Technology

2007-14 Niek Bergboer (UM)
   Context-Based Image Analysis

2007-15 Joyca Lacroix (UM)
   NIM: a Situated Computational Memory Model

2007-16 Davide Grossi (UU)
   Designing Invisible Handcuffs.

2007-17 Theodore Charitos (UU)
   Reasoning with Dynamic Networks in Practice

2007-18 Bart Orriens (UvT)
   On the development an management of adaptive business collaborations

2007-19 David Levy (UM)
   Intimate relationships with artificial partners

2007-20 Slinger Jansen (UU)
   Customer Configuration Updating in a Software Supply Network

2007-21 Karianne Vermaas (UU)
   Fast diffusion and broadening use: A research on residential adoption and usage of broadband internet in the Netherlands between 2001 and 2005

2007-22 Zlatko Zlatev (UT)
   Goal-oriented design of value and process models from patterns

2007-23 Peter Barna (TUE)
   Specification of Application Logic in Web Information Systems

2007-24 Georgina Ramirez Camps (CWI)
   Structural Features in XML Retrieval

2007-25 Joost Schalken (VU)
   Empirical Investigations in Software Process Improvement

2008

2008-01 Katalin Boer-Sorbn (EUR)
   Agent-Based Simulation of Financial Markets: A modular,continuous-time approach
2008-02 Alexei Sharanskykh (VU)  
On Computer-Aided Methods for Modeling and Analysis of Organizations

2008-03 Vera Hollink (UVA)  
Optimizing hierarchical menus: a usage-based approach

2008-04 Ander de Keijzer (UT)  
Management of Uncertain Data - towards unattended integration

2008-05 Bela Mutschler (UT)  
Modeling and simulating causal dependencies on process-aware information systems from a cost perspective

2008-06 Arjen Hommersom (RUN)  
On the Application of Formal Methods to Clinical Guidelines, an Artificial Intelligence Perspective

2008-07 Peter van Rosmalen (OU)  
Supporting the tutor in the design and support of adaptive e-learning

2008-08 Janneke Bolt (UU)  
Bayesian Networks: Aspects of Approximate Inference

2008-09 Christof van Nimwegen (UU)  
The paradox of the guided user: assistance can be counter-effective

2008-10 Wauter Bosma (UT)  
Discourse oriented summarization

2008-11 Vera Kartseva (VU)  
Designing Controls for Network Organizations: A Value-Based Approach

2008-12 Jozsef Farkas (RUN)  
A Semiotically Oriented Cognitive Model of Knowledge Representation

2008-13 Caterina Carraciolo (UVA)  
Topic Driven Access to Scientific Handbooks

2008-14 Arthur van Bunningen (UT)  
Context-Aware Querying; Better Answers with Less Effort

2008-15 Martijn van Otterlo (UT)  
The Logic of Adaptive Behavior: Knowledge Representation and Algorithms for the Markov Decision Process Framework in First-Order Domains

2008-16 Henriette van Vugt (VU)  
Embodied agents from a user’s perspective

2008-17 Martin Op ‘t Land (TUD)  
Applying Architecture and Ontology to the Splitting and Allying of Enterprises

2008-18 Guido de Croon (UM)  
Adaptive Active Vision

2008-19 Henning Rode (UT)  
From Document to Entity Retrieval: Improving Precision and Performance of Focused Text Search

2008-20 Rex Arendsen (UVA)  
Geen bericht, goed bericht.
2008-21 Krisztian Balog (UVA)
People Search in the Enterprise

2008-22 Henk Koning (UU)
Communication of IT-Architecture

2008-23 Stefan Visscher (UU)
Bayesian network models for the management of ventilator-associated pneumonia

2008-24 Zharko Aleksovski (VU)
Using background knowledge in ontology matching

2008-25 Geert Jonker (UU)
Efficient and Equitable Exchange in Air Traffic Management Plan Repair using Spender-signed Currency

2008-26 Marijn Huijbregts (UT)
Segmentation, Diarization and Speech Transcription: Surprise Data Unraveled

2008-27 Hubert Vogten (OU)
Design and Implementation Strategies for IMS Learning Design

2008-28 Ilidko Flesch (RUN)
On the Use of Independence Relations in Bayesian Networks

2008-29 Dennis Reidsma (UT)
Annotations and Subjective Machines - Of Annotators, Embodied Agents, Users, and Other Humans

2008-30 Wouter van Atteveldt (VU)
Semantic Network Analysis: Techniques for Extracting, Representing and Querying Media Content

2008-31 Loes Braun (UM)
Pro-Active Medical Information Retrieval

2008-32 Trung H. Bui (UT)
Toward Affective Dialogue Management using Partially Observable Markov Decision Processes

2008-33 Frank Terpstra (UVA)
Scientific Workflow Design; theoretical and practical issues

2008-34 Jeroen de Knijf (UU)
Studies in Frequent Tree Mining

2008-35 Ben Torben Nielsen (UvT)
Dendritic morphologies: function shapes structure

2009

2009-01 Rasa Jurgeleitane (RUN)
Symmetric Causal Independence Models

2009-02 Willem Robert van Hage (VU)
Evaluating Ontology-Alignment Techniques

2009-03 Hans Stol (UvT)
A Framework for Evidence-based Policy Making Using IT
2009-04 Josephine Nabukenya (RUN)
  Improving the Quality of Organisational Policy Making using Collaboration Engineering

2009-05 Sietse Overbeek (RUN)
  Bridging Supply and Demand for Knowledge Intensive Tasks -
  Based on Knowledge, Cognition, and Quality

2009-06 Muhammad Subianto (UU)
  Understanding Classification

2009-07 Ronald Poppe (UT)
  Discriminative Vision-Based Recovery and Recognition of Human Motion

2009-08 Volker Nannen (VU)
  Evolutionary Agent-Based Policy Analysis in Dynamic Environments

2009-09 Benjamin Kanagwa (RUN)
  Design, Discovery and Construction of Service-oriented Systems

2009-10 Jan Wielemaker (UVA)
  Logic programming for knowledge-intensive interactive applications

2009-11 Alexander Boer (UVA)
  Legal Theory, Sources of Law & the Semantic Web

2009-12 Peter Massathe (TUE, Humboldt-Universitaet zu Berlin)
  Operating Guidelines for Services

2009-13 Steven de Jong (UM)
  Fairness in Multi-Agent Systems

2009-14 Maksym Korotkiy (VU)
  From ontology-enabled services to service-enabled ontologies
  (making ontologies work in e-science with ONTO-SOA)

2009-15 Rinke Hoekstra (UVA)
  Ontology Representation - Design Patterns and Ontologies that Make Sense

2009-16 Fritz Reul (UvT)
  New Architectures in Computer Chess

2009-17 Laurens van der Maaten (UvT)
  Feature Extraction from Visual Data

2009-18 Fabian Groffen (CWI)
  Armada, An Evolving Database System

2009-19 Valentin Robu (CWI)
  Modeling Preferences, Strategic Reasoning and Collaboration in
  Agent-Mediated Electronic Markets

2009-20 Bob van der Vecht (UU)
  Adjustable Autonomy: Controlling Influences on Decision Making

2009-21 Stijn Vanderlooy (UM)
  Ranking and Reliable Classification

2009-22 Pavel Serdyukov (UT)
  Search For Expertise: Going beyond direct evidence

2009-23 Peter Hofgesang (VU)
  Modelling Web Usage in a Changing Environment

2009-24 Annerieke Heuvelink (VUA)
Cognitive Models for Training Simulations

2009-25 Alex van Ballegooij (CWI)
RAM: Array Database Management through Relational Mapping

2009-26 Fernando Koch (UU)
An Agent-Based Model for the Development of Intelligent Mobile Services

2009-27 Christian Glahn (OU)
Contextual Support of social Engagement and Reflection on the Web

2009-28 Sander Evers (UT)
Sensor Data Management with Probabilistic Models

2009-29 Stanislav Pokraev (UT)
Model-Driven Semantic Integration of Service-Oriented Applications

2009-30 Marcin Zukowski (CWI)
Balancing vectorized query execution with bandwidth-optimized storage

2009-31 Sofiya Katrenko (UVA)
A Closer Look at Learning Relations from Text

2009-32 Rik Farenhorst (VU) and Remco de Boer (VU)
Architectural Knowledge Management: Supporting Architects and Auditors

2009-33 Khiet Truong (UT)
How Does Real Affect Affect Affect Recognition In Speech?

2009-34 Inge van de Weerd (UU)
Advancing in Software Product Management: An Incremental Method Engineering Approach

2009-35 Wouter Koelewijn (UL)
Privacy en Politiegegevens;
Over geautomatiseerde normatieve informatie-uitwisseling

2009-36 Marco Kalz (OUN)
Placement Support for Learners in Learning Networks

2009-37 Hendrik Drachsler (OUN)
Navigation Support for Learners in Informal Learning Networks

2009-38 Riina Vuorikari (OU)
Tags and self-organisation: a metadata ecology for learning resources in a multilingual context

2009-39 Christian Stahl (TUE, Humboldt-Universitaet zu Berlin)
Service Substitution – A Behavioral Approach Based on Petri Nets

2009-40 Stephan Raaijmakers (UvT)
Multinomial Language Learning: Investigations into the Geometry of Language

2009-41 Igor Berezhnyy (UvT)
Digital Analysis of Paintings

2009-42 Toine Bogers
Recommender Systems for Social Bookmarking

2009-43 Virginia Nunes Leal Franqueira (UT)
Finding Multi-step Attacks in Computer Networks using
Heuristic Search and Mobile Ambients
2009-44 Roberto Santana Tapia (UT)  
Assessing Business-IT Alignment in Networked Organizations

2009-45 Jilles Vreeken (UU)  
Making Pattern Mining Useful

2009-46 Loredana Afanasiev (UvA)  
Querying XML: Benchmarks and Recursion

2010

2010-01 Matthijs van Leeuwen (UU)  
Patterns that Matter

2010-02 Ingo Wassink (UT)  
Work flows in Life Science

2010-03 Joost Geurts (CWI)  
A Document Engineering Model and Processing Framework for Multimedia documents

2010-04 Olga Kulyk (UT)  
Do You Know What I Know? Situational Awareness of Co-located Teams in Multidisplay Environments

2010-05 Claudia Hauff (UT)  
Predicting the Effectiveness of Queries and Retrieval Systems

2010-06 Sander Bakkes (UvT)  
Rapid Adaptation of Video Game AI

2010-07 Wim Fikkert (UT)  
Gesture interaction at a Distance

2010-08 Krzysztof Siewicz (UL)  
Towards an Improved Regulatory Framework of Free Software. Protecting user freedoms in a world of software communities and eGovernments

2010-09 Hugo Kielman (UL)  
A Politie gegevensverwerking en Privacy, Naar een effectieve waarborging

2010-10 Rebecca Ong (UL)  
Mobile Communication and Protection of Children 2010-11 Adriaan Ter Mors (TUD) The world according to MARP: Multi-Agent Route Planning

2010-12 Susan van den Braak (UU)  
Sensemaking software for crime analysis

2010-13 Gianluigi Folino (RUN)  
High Performance Data Mining using Bio-inspired techniques

2010-14 Sander van Splunter (VU)  
Automated Web Service Reconfiguration

2010-15 Lianne Bodenstaff (UT)  
Managing Dependency Relations in Inter-Organizational Models

2010-16 Sicco Verwer (TUD)  
Efficient Identification of Timed Automata, theory and practice
2010-17 Spyros Kotoulas (VU)
Scalable Discovery of Networked Resources: Algorithms, Infrastructure, Applications

2010-18 Charlotte Gerritsen (VU)
Caught in the Act: Investigating Crime by Agent-Based Simulation

2010-19 Henriette Cramer (UvA)
People’s Responses to Autonomous and Adaptive Systems

2010-20 Ivo Swartjes (UT)
Whose Story Is It Anyway?
How Improv Informs Agency and Authorship of Emergent Narrative

2010-21 Harold van Heerde (UT)
Privacy-aware data management by means of data degradation

2010-22 Michiel Hildebrand (CWI)
End-user Support for Access to Heterogeneous Linked Data

2010-23 Bas Steunebrink (UU)
The Logical Structure of Emotions

2010-24 Dmytro Tykhonov
Designing Generic and Efficient Negotiation Strategies

2010-25 Zulfiqar Ali Memon (VU)
Modelling Human-Awareness for Ambient Agents: A Human Mindreading Perspective

2010-26 Ying Zhang (CWI)
XRPC: Efficient Distributed Query Processing on Heterogeneous XQuery Engines

2010-27 Marten Voulon (UL)
Automatisch contracteren

2010-28 Arne Koopman (UU)
Characteristic Relational Patterns

2010-29 Stratos Idreos (CWI)
Database Cracking: Towards Auto-tuning Database Kernels

2010-30 Marieke van Erp (UvT)
Accessing Natural History - Discoveries in data cleaning, structuring, and retrieval

2010-31 Victor de Boer (UvA)
Ontology Enrichment from Heterogeneous Sources on the Web

2010-32 Marcel Hiel (UvT)
An Adaptive Service Oriented Architecture: Automatically solving Interoperability Problems

2010-33 Robin Aly (UT)
Modeling Representation Uncertainty in Concept-Based Multimedia Retrieval

2010-34 Teduh Dirgahayu (UT)
Interaction Design in Service Compositions

2010-35 Dolf Trieschnigg (UT)
Proof of Concept: Concept-based Biomedical Information Retrieval

2010-36 Jose Janssen (OU)
Paving the Way for Lifelong Learning; Facilitating competence development through a learning path specification

2010-37 Niels Lohmann (TUE)  
Correctness of services and their composition

2010-38 Dirk Fahland (TUE)  
From Scenarios to components

2010-39 Ghazanfar Farooq Siddiqui (VU)  
Integrative modeling of emotions in virtual agents

2010-40 Mark van Assem (VU)  
Converting and Integrating Vocabularies for the Semantic Web

2010-41 Guillaume Chaslot (UM)  
Monte-Carlo Tree Search

2010-42 Sybren de Kinderen (VU)  
Needs-driven service bundling in a multi-supplier setting - the computational e3-service approach

2010-43 Peter van Kranenburg (UU)  
A Computational Approach to Content-Based Retrieval of Folk Song Melodies

2010-44 Pieter Bellekens (TUE)  
An Approach towards Context-sensitive and User-adapted Access to Heterogeneous Data Sources, Illustrated in the Television Domain

2010-45 Vasilios Andrikopoulos (UvT)  
A theory and model for the evolution of software services

2010-46 Vincent Pijpers (VU)  
e3alignment: Exploring Inter-Organizational Business-ICT Alignment

2010-47 Chen Li (UT)  
Mining Process Model Variants: Challenges, Techniques, Examples

2010-48 Milan Lovric (EUR)  
Behavioral Finance and Agent-Based Artificial Markets

2010-49 Jahn-Takeshi Saito (UM)  
Solving difficult game positions

2010-50 Bouke Huurnink (UVA)  
Search in Audiovisual Broadcast Archives

2010-51 Alia Khairia Amin (CWI)  
Understanding and supporting information seeking tasks in multiple sources

2010-52 Peter-Paul van Maanen (VU)  
Adaptive Support for Human-Computer Teams: Exploring the Use of Cognitive Models of Trust and Attention

2010-53 Edgar Meij (UVA)  
Combining Concepts and Language Models for Information Access

2011

2011-01 Botond Cseke (RUN)
Variational Algorithms for Bayesian Inference in Latent Gaussian Models

2011-02 Nick Tinnemeier (UU)
Organizing Agent Organizations.
Syntax and Operational Semantics of an Organization-Oriented Programming Language

2011-03
Jan Martijn van der Werf (TUE)
Compositional Design and Verification of Component-Based Information Systems

2011-04 Hado van Hasselt (UU)
Insights in Reinforcement Learning;
Formal analysis and empirical evaluation of temporal-difference learning algorithms

2011-05 Base van der Raadt (VU)
Enterprise Architecture Coming of Age - Increasing the Performance of an Emerging Discipline.

2011-06 Yiwen Wang (TUE)
Semantically-Enhanced Recommendations in Cultural Heritage

2011-07 Yujia Cao (UT)
Multimodal Information Presentation for High Load Human Computer Interaction

2011-08 Nieske Vergunst (UU)
BDI-based Generation of Robust Task-Oriented Dialogues
Contextualised Mobile Media for Learning

2011-10 Bart Bogaert (UvT)
Cloud Content Contention

2011-11 Dhaval Vyas (UT)
Designing for Awareness: An Experience-focused HCI Perspective

2011-12 Carmen Bratosin (TUE)
Grid Architecture for Distributed Process Mining

2011-13 Xiaoyu Mao (UvT)
Airport under Control. Multiagent Scheduling for Airport Ground Handling

2011-14 Milan Lovric (EUR)
Behavioral Finance and Agent-Based Artificial Markets

2011-15 Marijn Koolen (UvA)
The Meaning of Structure: the Value of Link Evidence for Information Retrieval

2011-16 Maarten Schadd (UM)
Selective Search in Games of Different Complexity

2011-17 Jiyin He (UVA)
Exploring Topic Structure: Coherence, Diversity and Relatedness

2011-18 Mark Ponsen (UM)
Strategic Decision-Making in complex games

2011-19 Ellen Rusman (OU)
The Mind ’ s Eye on Personal Profiles

2011-20 Qing Gu (VU)
Guiding service-oriented software engineering - A view-based approach

2011-21 Linda Terlouw (TUD)
Modularization and Specification of Service-Oriented Systems

2011-22 Junte Zhang (UVA)
System Evaluation of Archival Description and Access

2011-23 Wouter Weerkamp (UVA)
Finding People and their Utterances in Social Media

2011-24 Herwin van Welbergen (UT)
Behavior Generation for Interpersonal Coordination with Virtual Humans On Specifying, Scheduling and Realizing Multimodal Virtual Human Behavior

2011-25 Syed Waqar ul Qounain Jaffry (VU)
Analysis and Validation of Models for Trust Dynamics

2011-26 Matthijs Aart Pontier (VU)
Virtual Agents for Human Communication - Emotion Regulation and Involvement-Distance Trade-Offs in Embodied Conversational Agents and Robots

2011-27 Aniel Bhulai (VU)
Dynamic website optimization through autonomous management of design patterns

2011-28 Rianne Kaptein(UVA)
Effective Focused Retrieval by Exploiting Query Context and Document Structure

2011-29 Faisal Kamiran (TUE)
Discrimination-aware Classification

2011-30 Egon van den Broek (UT)
Affective Signal Processing (ASP): Unraveling the mystery of emotions

2011-31 Ludo Waltman (EUR)
Computational and Game-Theoretic Approaches for Modeling Bounded Rationality

2011-32 Nees-Jan van Eck (EUR)
Methodological Advances in Bibliometric Mapping of Science

2011-33 Tom van der Weide (UU)
Arguing to Motivate Decisions

2011-34 Paolo Turrini (UU)
Strategic Reasoning in Interdependence: Logical and Game-theoretical Investigations

2011-35 Maaike Harbers (UU)
Explaining Agent Behavior in Virtual Training

2011-36 Erik van der Spek (UU)
Experiments in serious game design: a cognitive approach

2011-37 Adriana Burlutiu (RUN)
Machine Learning for Pairwise Data, Applications for Preference Learning and Supervised Network Inference

2011-38 Nyree Lemmens (UM)
Bee-inspired Distributed Optimization

2011-39 Joost Westra (UU)
Organizing Adaptation using Agents in Serious Games
2011-40 Viktor Clerc (VU)
Architectural Knowledge Management in Global Software Development

2011-41 Luan Ibraimi (UT)
Cryptographically Enforced Distributed Data Access Control

2011-42 Michal Sindlar (UU)
Explaining Behavior through Mental State Attribution

2011-43 Henk van der Schuur (UU)
Process Improvement through Software Operation Knowledge

2011-44 Boris Reuderink (UT)
Robust Brain-Computer Interfaces

2011-45 Herman Stehouwer (UvT)
Statistical Language Models for Alternative Sequence Selection

2011-46 Beibei Hu (TUD)
Towards Contextualized Information Delivery: A Rule-based Architecture for the Domain of Mobile Police Work

2011-47 Azizi Bin Ab Aziz(VU)
Exploring Computational Models for Intelligent Support of Persons with Depression

2011-48 Mark Ter Maat (UT)
Response Selection and Turn-taking for a Sensitive Artificial Listening Agent

2011-49 Andreea Niculescu (UT)
Conversational interfaces for task-oriented spoken dialogues: design aspects influencing interaction quality

2012

2012-01 Terry Kakeeto (UvT)
Relationship Marketing for SMEs in Uganda

2012-02 Muhammad Umair(VU)
Adaptivity, emotion, and Rationality in Human and Ambient Agent Models

2012-03 Adam Vanya (VU)
Supporting Architecture Evolution by Mining Software Repositories

2012-04 Jurriaan Souer (UU)
Development of Content Management System-based Web Applications

2012-05 Marijn Plomp (UU)
Maturing Interorganisational Information Systems

2012-06 Wolfgang Reinhardt (OU)
Awareness Support for Knowledge Workers in Research Networks

2012-07 Rianne van Lambalgen (VU)
When the Going Gets Tough: Exploring Agent-based Models of Human Performance under Demanding Conditions

2012-08 Gerben de Vries (UVA)
Kernel Methods for Vessel Trajectories
2012-09 Ricardo Neisse (UT)
Trust and Privacy Management Support for Context-Aware Service Platforms

2012-10 David Smits (TUE)
Towards a Generic Distributed Adaptive Hypermedia Environment

2012-11 J.C.B. Rantham Prabhakara (TUE)
Process Mining in the Large: Preprocessing, Discovery, and Diagnostics

2012-12 Kees van der Sluijs (TUE)
Model Driven Design and Data Integration in Semantic Web Information Systems

2012-13 Suleman Shahid (UvT)
Fun and Face: Exploring non-verbal expressions of emotion during playful interactions

2012-14
Evgeny Knutov (TUE)
Generic Adaptation Framework for Unifying Adaptive Web-based Systems

2012-15 Natalie van der Wal (VU)
Social Agents. Agent-Based Modelling of Integrated Internal
and Social Dynamics of Cognitive and Affective Processes

2012-16 Fiemke Both (VU)
Helping people by understanding them -
Ambient Agents supporting task execution and depression treatment

2012-17 Amal Elgammal (UvT)
Towards a Comprehensive Framework for Business Process Compliance

2012-18 Eltjo Poort (VU)
Improving Solution Architecting Practices

2012-19 Helen Schonenberg (TUE)
What’s Next? Operational Support for Business Process Execution

2012-20 Ali Bahramisharif (RUN)
Covert Visual Spatial Attention, a Robust Paradigm for Brain-Computer Interfacing

2012-21 Roberto Cornacchia (TUD)
Querying Sparse Matrices for Information Retrieval

2012-22 Thijs Vis (UvT)
Intelligence, politie en veiligheidsdienst: verenigbare grootheden?

2012-23 Christian Muehl (UT)
Toward Affective Brain-Computer Interfaces:
Exploring the Neurophysiology of Affect during Human Media Interaction

2012-24 Laurens van der Werff (UT)
Evaluation of Noisy Transcripts for Spoken Document Retrieval

2012-25 Silja Eckartz (UT)
Managing the Business Case Development in Inter-Organizational IT Projects:
A Methodology and its Application

2012-26 Emile de Maat (UVA)
Making Sense of Legal Text

2012-27 Hayrettin Gurkok (UT)
Mind the Sheep!
User Experience Evaluation & Brain-Computer Interface Games

2012-28 Nancy Pascall (UvT)  
Engendering Technology Empowering Women

2012-29 Almer Tigelaar (UT)  
Peer-to-Peer Information Retrieval

2012-30 Alina Pommeranz (TUD)  
Designing Human-Centered Systems for Reflective Decision Making

2012-31 Emily Bagarukayo (RUN)  
A Learning by Construction Approach for Higher Order Cognitive Skills Improvement, Building Capacity and Infrastructure

2012-32 Wietske Visser (TUD)  
Qualitative multi-criteria preference representation and reasoning

2012-33 Rory Sie (OUN)  
Coalitions in Cooperation Networks (COCOON)

2012-34 Pavol Jancura (RUN)  
Evolutionary analysis in PPI networks and applications

2012-35 Evert Haasdijk (VU)  
Never Too Old To Learn – On-line Evolution of Controllers in Swarm- and Modular Robotics

2012-36 Denis Ssebugwawo (RUN)  
Analysis and Evaluation of Collaborative Modeling Processes

2012-37 Agnes Nakakawa (RUN)  
A Collaboration Process for Enterprise Architecture Creation

2012-38 Selmar Smit (VU)  
Parameter Tuning and Scientific Testing in Evolutionary Algorithms

2012-39 Hassan Fatemi (UT)  
Risk-aware design of value and coordination networks

2012-40 Agus Gunawan (UvT)  
Information Access for SMEs in Indonesia

2012-41 Sebastian Kelle (OU)  
Game Design Patterns for Learning

2012-42 Dominique Verpoorten (OU)  
Reflection Amplifiers in self-regulated Learning

2012-43 Withdrawn

2012-44 Anna Tordai (VU)  
On Combining Alignment Techniques

2012-45 Benedikt Kratz (UvT)  
A Model and Language for Business-aware Transactions

2012-46 Simon Carter (UVA)  
Exploration and Exploitation of Multilingual Data for Statistical Machine Translation

2012-47 Manos Tsagkias (UVA)
Mining Social Media: Tracking Content and Predicting Behavior

2012-48 Jorn Bakker (TUE)
Handling Abrupt Changes in Evolving Time-series Data

2012-49 Michael Kaisers (UM)
Learning against Learning -
Evolutionary dynamics of reinforcement learning algorithms in strategic interactions

2012-50 Steven van Kervel (TUD)
Ontology driven Enterprise Information Systems Engineering

2012-51 Jeroen de Jong (TUD)
Heuristics in Dynamic Scheduling; a practical framework with a case study in elevator dispatching

2013

2013-01 Viorel Milea (EUR)
News Analytics for Financial Decision Support

2013-02 Erietta Liarou (CWI)
MonetDB/DataCell: Leveraging the Column-store Database Technology for Efficient and Scalable Stream Processing

2013-03 Szymon Klarman (VU)
Reasoning with Contexts in Description Logics

2013-04 Chetan Yadati(TUD)
Coordinating autonomous planning and scheduling

2013-05 Dulce Pumareja (UT)
Groupware Requirements Evolutions Patterns

2013-06 Romulo Gonzalves(CWI)
The Data Cyclotron: Juggling Data and Queries for a Data Warehouse Audience

2013-07 Giel van Lankveld (UT)
Quantifying Individual Player Differences

2013-08 Robbert-Jan Merk(VU)
Making enemies: cognitive modeling for opponent agents in fighter pilot simulators

2013-09 Fabio Gori (RUN)
Metagenomic Data Analysis: Computational Methods and Applications

2013-10 Jeewanie Jayasinghe Arachchige(UvT)
A Unified Modeling Framework for Service Design