Governance of Innovation Project Management

Necessary and Neglected

Christoph Johann Stettina

The studies described in this thesis were performed at the Centre for Innovation at the Faculty Campus The Hague and the Leiden Institute of Advanced Computer Science at the Faculty of Science, Leiden University, the Netherlands.

This research project has been supported by the Living Lab The Hague project co-funded with support from the European Regional Development Fund of the European Union.

Governance of Innovation Project Management
Necessary and Neglected

Cover design by Pascal Schilp [www.thepassle.nl], Royal Academy of Art, The Hague

Cover photographs: Back left: Detroit Industry Murals, Diego Rivera - Middle: Chemiefaserwerk Guben, Meister der Zentralwerkstatt, Bundesarchiv Bild 183-G0925-0036-001 - Front right: Agile Portfolio Management Game, Daniel van den Hoven

ISBN 978-90-8555-098-3
NUR 754

Governance of Innovation Project Management
Necessary and Neglected

PROEFSCHRIFT

ter verkrijging van
de graad van Doctor aan de Universiteit Leiden,
op gezag van de Rector Magnificus prof.mr. C.J.J.M. Stolker,
volgens besluit van het College voor Promoties
te verdedigen op donderdag 21 mei 2015
klokke 16:15 uur

door

Christoph Johann Stettina
geboren te Peiskretscham, Silezië
in 1980

Samenstelling promotiecommissie:

promotor:	prof. dr. B.R. Katzy	Universiteit Leiden (LIACS)
promotor:	prof. dr. J. de Vries	Universiteit Leiden (CDH)
co-promotor:	dr. L.P. Groenewegen	Universiteit Leiden (LIACS)
overige leden:	prof. dr. J.N. Kok	Universiteit Leiden
	prof. dr. H.J. van den Herik	Universiteit Leiden
	prof. dr. A. Plaat	Universiteit Leiden
	prof. dr. P.K. Abrahamsson	Norwegian University of Science and Technology
	prof. dr. M. Martinsuo	Tampere University of Technology

To my grandfather
and the remainder of my family

Contents

List of Figures

List of Tables

List of Definitions

Knowledge workers

... contribute to the development of economic value through their application of analytical skills and theoretical knowledge. In contrast to the manual workers of the industrial age, knowledge workers need to be intelligent and possess a large degree of formal education (cf. Drucker (1994)).

Team

.. is a group of individuals who collaborate towards a common goal (cf. Salas, Sims and Burke (2005)).

Project

"..is a temporary organization to which resources are assigned to undertake a unique, novel and transient endeavour managing the inherent uncertainty and need for integration in order to deliver beneficial objectives of change." (Turner & Müller, 2003).

Project-based organization

...uses projects as the primary unit of organizing the main functions, the primary business mechanism (cf. Hobday (2000)).

Routines

..are *"repetitive, recognizable patterns of interdependent actions, carried out by multiple actors"* (Feldman & Pentland, 2003). Organizational routines are live, change in context and play an integral role in the development of capabilities (cf. Salvato (2009)).

(Project) Governance

..determines rights, responsibilities, power of actors, defines *"the rules of the game"* of project work (cf. Ahola, Ruuska, Artto and Kujala (2013)).

Agile Knowledge Worker Organizations

..Agile knowledge worker project organizations are those that learn fast and are effective in delivering value.

List of Studies

This thesis is the result of intense research that followed a research agenda which focused on understanding how agile innovation-driven project organizations are governed. The research has been implemented as a series of studies using a variety of research methods. Each of the studies has been published as given in the list below. Chapters of this thesis are revisions of these studies.

1. Stettina, C. J., & Heijstek, W. (2011). **Necessary and neglected? An empirical study of internal documentation in agile software development teams.** In Proceedings of the 29th ACM international conference on Design of communication (pp. 159-166). ACM.

2. Stettina, C. J., & Heijstek, W. (2011). **Five agile factors: Helping self-management to self-reflect.** In Systems, Software and Service Process Improvement (pp. 84-96). Springer Berlin Heidelberg.

3. Stettina, C. J., Groenewegen, L. P., & Katzy, B. R. (2011). **Structuring Medical Agility.** In HEALTHINF 2011 (pp. 614-617).

4. Stettina, C. J., Heijstek, W., & Fægri, T. E. (2012). **Documentation work in agile teams: the role of documentation formalism in achieving a sustainable practice.** In Agile Conference (AGILE), 2012 (pp. 31-40). IEEE.

5. Stettina, C. J., Groenewegen, L. P., & Katzy, B. R. (2012). **Towards flexibility and dynamic coordination in computer-interpretable enterprise process models.** In Enterprise Interoperability V (pp. 105-115). Springer London.

6. Katzy, B. R., Stettina, C. J., Groenewegen, L. P., & de Groot, M. J. (2011). **Managing weak ties in collaborative work.** In Concurrent Enterprising (ICE), 2011 17th International Conference on (pp. 1-9). IEEE.

7. Stettina, C. J., & Kroon, E. (2013). **Is There an Agile Handover? An Empirical Study of Documentation and Project Handover Practices Across Agile Software Teams.** In 19th ICE & IEEE-ITMC International Conference, The Hague, the Netherlands.

8. Stettina, C. J., Zhou, Z., Bäck, T., & Katzy, B. (2013). **Academic education of software engineering practices: towards planning and improving capstone courses based upon intensive coaching and team routines.** In Software Engineering Education and Training (CSEE&T), 2013 IEEE 26th Conference on (pp. 169-178). IEEE.

9. Stettina, C. J., & Hörz, J. (2015). **Agile portfolio management: An empirical perspective on the practice in use.** International Journal of Project Management. 33(1), 140 - 152.

Further co-authored articles:

10. Medik, V. L., & Stettina, C. J. (2014). **Towards responsible workplace innovation: The rise of NWW in public knowledge organizations and their impact on governance**. In Engineering, Technology and Innovation (ICE), 2014 International ICE Conference on (pp. 1-9). IEEE.

11. Zijdemans, S. H., & Stettina, C. J. (2014). **Contracting in Agile Software Projects: State of Art and How to Understand It.** In Agile Processes in Software Engineering and Extreme Programming (pp. 78-93). Springer International Publishing.

12. Huayller, H., Stettina, C. J., von Ketelhodt, M., & Katzy, B. (2015). **The Kindergarten of Entrepreneurship: Developing Entrepreneurial Spirit and Skills through Community Building at Universities.** In 21st ICE & IEEE International Technology Management Conference, Belfast, Northern Ireland.

Part I

Introduction

1 Arranging Good Governance

The thesis investigates the worldwide intricate problem of arranging a good governance for pursuing innovation projects in public and private organizations. The emphasis on governance started some twenty-five years ago. In retrospect we may state: then, without any discussion, it was necessary. In the past two decades, many occurrences of failing governance reached the headlines of the daily papers. So, the question arose: is governance neglected?

The pluriformity of the organizations on which governance was imposed made a direct answer difficult, if not impossible. Therefore, we investigate the guiding question: is governance necessary *or* neglected? But in anticipation of the readers' thoughts I have titled the book *necessary and neglected*.

The course of the first Chapter is as follows. In Section 1.1 I lay out my motivation for the thesis. In Section 1.2 I formulate the problem statement and the research questions. In Section 1.3 I address the research objectives and the research methodology. In Section 1.4 I mention my point of departure with respect to my philosophical assumptions. In section 1.5 I discuss the scope of the thesis. In Section 1.6 I indicate the relevance and my contributions. Finally, in Section 1.7 I address the structure of the thesis.

1.1 Motivation

My interest in the topic of agile project management began when studying IT teams in the Norwegian banking sector. There I encountered software development teams being proud on their ways of working, driven by reflection and self-managing mechanisms. In contrast to

many traditional software development organizations where a project manager tells his[1] team what to do, these teams had team members with an eagerness to adapt and learn - to improve the quality of the project as well as their ways of working.

In our knowledge-based economy, there is an emphasis on innovation to sustain economic growth. Organizations creating innovative products or services are increasingly relying on project work executed by knowledge worker teams. However, the teams are not always at their best according to daily paper reports on failing projects (Matta & Ashkenas, 2003; Flyvbjerg & Budzier, 2011; Benschop/ANP, 2014).

The motivation to write this thesis is mainly driven by the question: How could it happen? Considering (a) our gradual transition towards a knowledge society and (b) the existing project management frameworks, for quite some time, roughly three options are possible in my opinion: (1) the knowledge workers did not do a good job, (2) the governance and management was insufficient, (3) the transition from traditional to agile is still in it is fledgling stages.

My first two clues are: (1) knowledge workers have properties different from those of manual workers towards the way of controlling of which many organizations today are shaped, and (2) existing governance models have difficulties to cope with the properties of knowledge work. Drucker (1999) calls the improvement of knowledge worker productivity a *"survival requirement"* for the developed countries. However, while the importance to establish project teams is widely understood today, the way to actually organize and execute innovation projects in and across knowledge worker teams is still troublesome for many organizations.

Due to the origins of project management in civil engineering and its roots in classic management, existing project governance frameworks often give the impression of full predictability of projects, their results and a similar applicability of the applied methods in all environments. However, the process to construct an oil platform is essentially different from that of (1) the development of Twitter, (2) the process to write a scientific article or (3) knowledge related work in general. Whilst most (project) management methods available today rely on a top-down instruction and command-and-control structures, innovation cannot be fully predicted and instructed in the same way as we used to predict the throughput of machines. Innovation and knowledge in general are created in a series of interactions across actors not bounded by hierarchies or other organizational structures. As such we cannot understand innovation and technology management using lenses of the industrial age.

An interesting case are the agile project management methods. They caused a silent revolution in the way projects are organized and executed in practice (Abrahamsson, Conboy & Wang, 2009). Compared to traditional plan-driven project management methods there are two differences: (1) they embrace project environments as uncertain rather than assuming their predictability and (2) they do not follow linear sequence of steps from project definition to delivery. They enable co-creation of the product with relevant stakeholders through an iterative delivery of intermediate project results. However, agile teams are entrepreneurial, they

[1] For brevity and readability, 'he' and 'his' are used whenever 'he or she' and 'his or her' are meant.

take over many traditional project management tasks such as team level resource allocation, estimation, planning, and presentation of intermediate results. This mixture challenges the way we understand, manage and govern project teams.

Instead, the currently available frameworks such as the guidelines provided by the Project Management Institute (PMI, 2008) or PRINCE2 (PRojects IN Controlled Environments, version 2) (Murray et al., 2009), provide mostly top-down, prescriptive approaches to understand multi-project environments. These frameworks enable understanding at higher levels of abstraction through looking at goals, potential deliverables and high-level steps to achieve them. However, they mostly treat team-level and inter-team-level project activities as blackboxes. It leads to a negligence of the actors and their interactions based on respective beliefs, desires and goals embedded in a context (cf. Cicmil, Williams, Thomas and Hodgson (2006)).

The literature points at the necessity to understand (1) innovation where it happens, namely on the level of individual actors and (2) a series of interactions they follow to pass and enrich an idea across a number of knowledge domains. Following that line of thought initial contributions in the field of agile project management promote an investigation of concrete as-is performative practices (Thummadi, Shiv & Lyytinen, 2011).

1.2 Problem Statement and Research Questions

While the interest in team-based project governance models becomes increasingly evident, our understanding of how to organize, coordinate, and govern multiple knowledge worker teams in practice is still very limited. With large networks of actors in teams and sub-teams a hierarchical view on organizations does not help understanding the interactions across the individuals and teams. Existing project management frameworks are being criticized for not providing a sufficient understanding of the practice in use (Cicmil et al., 2006).

In an attempt to contribute to the existing body of knowledge, my research objectives discuss the implications of knowledge workers on (1) multi-project management, (2) governance and (3) the role of management using agile teams as a case study. To practitioners, this thesis describes the characteristics of agile project management on team level and multi-team level, and multi-project level. To academia, it provides a view on project management as a (1) set of skills and organizational routines (Pentland & Feldman, 2007) performed by the teams in context rather than a number of high-level frameworks, and a (2) research agenda.

I thus pose the following problem statement.

PS: Is governance of innovation project management necessary or neglected?

To answer the problem statement three research questions (RQs) are formulated. These research questions guide the research.

RQ1: How can governing bodies in organizations of knowledge worker teams understand and steer multiple knowledge worker project teams in practice?

RQ2: What are the components of governance necessary to understand knowledge worker project organizations?

RQ3: How can governance address the dynamics of knowledge work across multiple knowledge worker teams?

1.3 Research Objectives and Research Methodology

The research objective is to understand the governance and management practices in a real-world context where events cannot be controlled. In order to understand and asses the problem statement and the proposed research questions, I have conducted eight studies. Each of them follows an own rigorous research methodology. I address the relationship of the studies towards achieving the overarching goal on governance.

There are many books about governance, most of which discuss a top-down approach. Other more contemporary authors argue that analogously to the time-and-motion studies by F. W. Taylor (1911) we need to go back to the *work floor*, to the actual knowledge worker to re-establish our understanding of their embedding in organizations (Latour, 2005).

For a proper execution of the research I follow the advice of contemporary authors in the fields of organization science (Feldman & Pentland, 2003), project management (Cicmil et al., 2006), policy making (Wilson, 2000) and agile software development (Thummadi et al., 2011) to recreate the understanding of organizations beginning with the individual.

The main methodological approach is: (1) literature review, (2) analyzing the findings, (3) executing eight case studies, (4) analyzing the results, and (5) formulating the conclusions. The analysis is mainly qualitative (cf. Robson (2002); Yin (2009)) to enable an in-depth understanding of the governance phenomenon in context. Further, following an interpretivist approach for combinations of qualitative and quantitative data (cf. Dixon-Woods, Agarwal, Jones, Young and Sutton (2005)) I applied a mixed-methods approach to enrich the qualitative view. The reason for this is twofold. On one hand I would like to enrich the qualitative data with comparable quantifiable results, and on the other hand I would like to improve my own knowledge of research methods.

For data collection I used qualitative process descriptions, ethnographical notes, semi-structured and informal interviews and linked them to quantitative questionnaires and artifacts developed across the cases.

The application of different rigorously followed research methods within the self-contained studies are expected to contribute to my methodology education. Indeed, methodology plays an important role within the PhD. In order to understand the knowledge worker context I use

Table 1.1: Research methods and data sets

Thesis Part	**Ch.**	**Research Method**	**Data Collection Method(s)**	**Data Set**	**PS**	**RQ1**	**RQ2**	**RQ3**
I	1	-	-		✓	✓	✓	✓
I	2	Literature review	-		✓	✓	✓	✓
II	3	Grounded Theory - like	Open Interviews	A		✓		
III	4	Mixed Method	Survey + Interviews	B		✓	✓	
III	5	Mixed Method	Survey + Interviews	B			✓	
III	6	Mixed Method	Observations	C				✓
IV	7	Grounded Theory-like	Semi-structured Interviews	E		✓	✓	
IV	8	Mixed Method	Observations, Interviews and Surveys	D			✓	✓
IV	9	Experiment	Observations, Survey, Interviews	F			✓	
IV	10	Experiment	Observations, Survey, Interviews	G			✓	✓
V	11	-	-		✓	✓	✓	✓
V	12	-	-		✓	✓	✓	✓

Data set:
A: Interviews with 24 board level executives in Swiss knowledge worker organizations
B: 78 valid survey responses from 79 individuals from 8 teams, qualitative responses/interviews with ScrumMasters of each team
C: On site observations and a video recorded intervention
D: Main-study: 10 interviews, 89 survey responses, Pre-study: 61 survey responses
E: 30 Interviews with project and portfolio management staff in 14 organizations
F: 8 teams, 14 weeks: Observations, informal interviews, artifacts, longitudinal surveys
G: 6 teams, 6 weeks: Observations, informal interviews, artifacts, longitudinal surveys

agile product development teams as a focus point. Those are enriched by cases on medical teams, education and software engineering. In Table 1.1 I provide an overview of the research methods and the data used. They are discussed later in detail.

1.4 Philosophical Positions

Methodology is only a part of the full research philosophy. Other parts are ontology, epistemology, and inquiry aim. I distinguish these philosophical positions: positivism, post positivism, and constructivism.

In this thesis I take a constructivist position. I aim to rebuild the organization starting with the individual knowledge worker, their work in teams and their embeddings in the scope of an organization. Table 1.2 presents a summary of the major world views, paradigms applied in research according to Lincoln, Lynham and Guba (2011).

1.5 Scope of the Thesis

My research focusses on governance of knowledge worker teams in multi-project organizations. Due to their increased application and the associated successful application of agile methods (cf. Dybå and Dingsøyr (2008a)) in knowledge worker teams developing new products and services. I will use organizations as case subjects. In summary, I will study knowledge worker teams in context. Table 1.3 provides an overview of topics. I divide the topics in topics inside and outside the scope of this dissertation.

1.6 Relevance and Contributions

This research has relevance to science and to practice. It is based on the use-inspired basic science of the Pasteur's Quadrant (Stokes, 1997).[2]

The dissertation aims to contribute to theory building in five ways. First, it builds theory based on empirical findings across three levels of analysis: the knowledge worker, the knowledge worker team and the multi-project organization. So, I rebuild the organization beginning with the individual knowledge worker, the knowledge worker project teams and their embedding in the scope of an organization. Second, it builds theory based on the analysis performed with respect to the existing literature. There I develop the concept of *Substantial*

[2] Scientific research methods can be classified by (1) whether they contribute to a fundamental understanding of scientific problems (e.g., such as the work of scientists like Niels Bohr), or (2) whether they contribute to societal challenges (e.g., such as the work of inventors like Thomas Edison). Louis Pasteur and his commitment to basic research and its practical applications are used to exemplify work bridging the gap between "basic" and "applied" research. The term Pasteur's Quadrant was introduced by Stokes (1997).

Table 1.2: Basic belief of inquire paradigms (adapted from Lincoln, Lynham and Guba (2011); assumed position marked gray)

	Positivism	**Postpositivism**	**Constructivism**
Ontology	Naïve realism - "real" reality but apprehensible	Critical realism-"real" reality but only imperfectly and probabilistically apprehensible	Relativism - local and specific structured constructed realities
Epistemology	Dualist/objectivist; findings true	Modified dualist/objectivist; critical tradition / community; findings probably true	Transactional / subjectivist / created findings / co-created multiple truths
Methodology	Experimental / manipulative; verification of hypotheses; chiefly quantitative methods	Modified experimental / manipulative; critical multiplism; falsification of hypotheses; may include qualitative methods	Hermeneutical / dialectical, often qualitative
Inquiry aim	theory verification, explanation: prediction and control		theory generation, understanding; reconstruction

Table 1.3: Scope of the thesis

Topic	In scope of the thesis	Not in scope of the thesis
Project management	• Project developing new products and services across several teams, especially software based products • Enterprise project management	• Large civil engineering and construction projects
Knowledge workers	• Creation and transfer of knowledge across teams	• Cognitive aspects, psychology
Governance	• Governance of projects	• Corporate governance, government

Management advocating a new role and new responsibilities of management, such as the forming of a coordinating body across knowledge worker teams. Third, it builds theory based on case studies of agile teams in product development, software engineering, medicine and academia. I contribute to understanding the role of routines and boundary objects as carriers of governance. Fourth, it builds theory based on the reestablished view on the organization. I develop a framework for governance in knowledge worker organizations. Fifth, I further contribute to theory building by developing a number of propositions to be evaluated in future research.

The dissertation also contributes to practice. Following the ideas of use-inspired basic research as pursued by Louis Pasteur (Stokes, 1997), I arrive at the practical implications. First, I provide a list of characteristics of agile knowledge worker teams and how to improve those using a reflective questionnaire. Second, I provide understanding and recommendations for knowledge transfer across project teams using routines and documentation as boundary objects. Third, I provide a model consisting of four components across three layers for participants to consider in a knowledge worker project organization as part of a reoccurring governance routine.

1.7 Structure of the Thesis

The thesis consists of twelve chapters divided into five main parts: (I) Introduction, (II-IV) Empirical Investigation, and (V) Conclusions. To improve readability the empirical part is further sub-divided into the three levels of analysis consisting of: (II) the knowledge worker, (III) the knowledge worker teams, and (IV) the multi-project organization. The empirical part

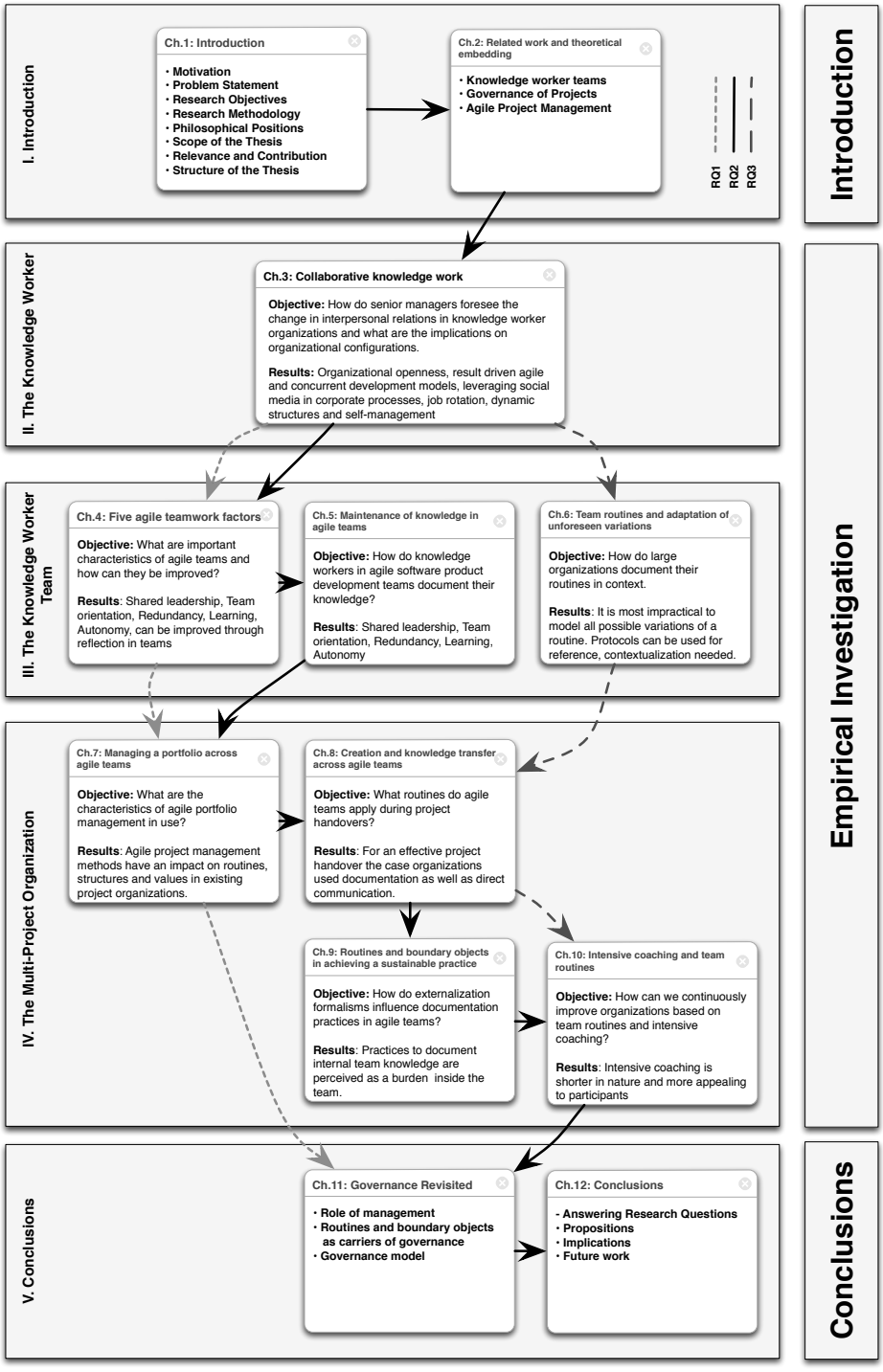


Figure 1.1: Thesis structure

consists of eight self-contained studies. Each of the studies follows a rigorous methodology and provides concrete research results answering RQ1, RQ2, and RQ3, and thus contributes to understanding the overall problem statement. Figure 1.1 depicts the structure of this thesis.

Moreover, the structure of the thesis is communicated by a chronological order of ideas. More precisely, the idea for this study emerged from (1) prior experiences with agile methods collected during my master's thesis and (2) the perceptions of top management presented in Part II. These two different ambitions led to a number of studies on knowledge worker teams presented in Part III and a number of subsequent studies in multi-team and multi-project environments presented in Part IV.

In Part I, Chapter 1, I motivate the study as well as its background. I formulate a problem statement and three research questions. Moreover, I provide the research objective, and describe the research methodology as well as my philosophical position. Thereafter, I formulate the scope, relevance, contributions and structure. In Part I, Chapter 2, I disseminate the related work and provide a review of the knowledge worker, the project management and the project governance literature.

In Part II, Chapter 3, I discuss the emergence of new organizational arrangements in knowledge worker organizations from the perspective of senior level management.

In Part III, I provide an in-depth empirical perspective on agile project teams developing new products and services in practice. Chapter 4 covers characteristics of agile knowledge worker teams. Chapter 5 discusses how knowledge worker teams develop new products and services and how they share their knowledge. Chapter 6 discusses how teams coordinate their work in respect to unforeseen adaptations.

Part IV provides an empirical evaluation of agile project management methods applied in multi-project environments. In Chapter 7 I discuss how knowledge worker teams, when embedded in organizations are challenging existing organizational structures. In Chapter 8 I discuss how project results are transferred across different teams. In Chapters 9 and 10 I present two experiments emphasizing the impact of routines and artifacts on sustainability of a practice.

In Part V, I review my results towards a governance model for innovation project organizations. Then I provide answers to RQ1, RQ2, and RQ3, and address the problem statement. Finally I draw conclusions and recommend future directions of research.

2 Related Work and Theoretical Embedding

This chapter describes my literature review and introduces three academic domains covered in this thesis. Obviously, the thesis is multidisciplinary as it covers multiple domains spanning across development of products and services, in particular software products (see Section 2.1), organizational studies, project governance (see Section 2.2), and agile project management (see Section 2.3).

2.1 Knowledge Worker Teams Creating New Products and Services

Already twenty years ago Drucker (1994) stated that our society increasingly evolved into a knowledge society. A knowledge society aims to make knowledge available to all its members in order to contribute to economic growth and improve the human condition. Put into wider context, a knowledge society differs from an information society as knowledge is the *"basic economic resource"* rather than capital or manual workforce (Drucker, 1993, 1994).

Changes to our society often bring a change to needs, and demands and to how we fulfill these demands in creating new knowledge, products and services. This has implications on how our organizations are structured and how we will create those new products and services. For example, the postwar manufacturing society was driven by Fordism (De Grazia, 2009). It fueled production of standardized products especially on assembly lines based on the principles of Scientific Management introduced by F. W. Taylor (1911).

Current organizations have been designed according to the need of the postwar manufacturing society (Piore & Sabel, 1984). As Drucker (1977, p.146) clearly explained, the role of management emerged in the 1920s, a time of the emergence of large-scale human organizations, such as General Motors, Standard Oil Trust, E. I. du Pont de Nemours & Company,

or the Japanese *zaibatsu* conglomerates. It was a time, as he argues, in which management was necessary to bring structure and control into the rapidly growing business enterprises. The new structure was largely built around manual labour and improvement of work conditions based on scientific management (F. W. Taylor, 1911).

A new knowledge society obviously challenges these structures that are based on control of the manual assembly line such as labour as envisioned by Drucker (1993) two decades ago. According to Drucker (1993) one of the major challenges of a knowledge society is to enable organizations to build capabilities enabling self-transformation through: (1) continuous improvement of all its activities, (2) creation of new applications based on earlier successes, and (3) establishing of continuous innovation as an organized process.

Below I focus on two specific issues, namely the properties of knowledge workers and knowledge creation as a governance routine.

2.1.1 Properties of Knowledge Workers

Individuals creating new products or services are generally considered as knowledge workers. They contribute to a knowledge-based economy through their application of theoretical and analytical knowledge (Drucker, 1994). The acquisition of knowledge involves complex cognitive processes and a knowledge worker needs to have a high-degree formal education in order to fuel the necessary analytical skills.

According to Drucker (1994) knowledge workers differ fundamentally in their approach to work, their mind-set, and their habit of continuous learning. There are fundamental differences between manual labor, assembly line jobs and knowledge creation. Weiss (1960) compares knowledge growth to the growth of an organism; consequently information (raw data) is then compared to food.

Due to the amount of formal education necessary to understand the complex concepts to be applied in analytics, many knowledge workers receive university level education. However, universities, while good at providing formal education, provide little training on how to work in teams.

Optimization of approaches to the work of knowledge workers has been mentioned as one of the biggest tasks for the 21st century (see e.g., Drucker (1999); Davenport (2013)). Drucker (1999) determines six major factors determining knowledge-worker productivity: (1) the knowledge of what the task is, (2) autonomy, as knowledge workers are responsible for their own productivity, (3) continuing innovation, (4) engage in continuous learning, (5) quantity is not a primary measure of productivity, as quality of a knowledge worker output is as least as important, and (6) acknowledging that the knowledge worker is an asset to the organization rather than a cost factor.

Research on knowledge worker productivity emphasizes the importance of autonomy. Autonomy is important for knowledge workers as it enables decision making on a local level,

since knowledge workers can make better decisions regarding the execution of a local task. As it is for the respective knowledge worker to decide what the actual task is (e.g., a nurse has to decide which action to take during a particular intervention), knowledge workers cannot be governed in the same fashion as manual workers were managed. As elaborated by Mintzberg (1979) in his ideas on the *Professional Organization*, governing bodies can enable knowledge workers by making sure they have the right skills, or routines.

While a few studies have examined the properties and effectiveness of knowledge worker teams most of the work concentrates on teams in separation. Here, a valuable contribution can be made to project management literature, particularly on (1) multi-project management as encouraged by Martinsuo and Lehtonen (2007) and (2) Project-Based Organizations as proposed by Hobday (2000).

2.1.2 Knowledge Creation as Part of a Governance Routine

In order to understand governance across knowledge worker project teams we need to understand (1) the nature of knowledge creation, (2) how new knowledge is (2a) created, (2b) shared, and (2c) used to make decisions to achieve specific goals.

Following the line of thought by Polanyi (1967) and other authors (e.g., Nonaka and Takeuchi (1995)), I would like to stress the two distinct features of knowledge: explicit and tacit (Nonaka & Takeuchi, 1995). The explicit component of knowledge is information, raw data. The tacit component is much more difficult to grasp and transfer due to its intangible nature. As Polanyi (1967, p.4) presents in the example of recognizing a human face *"we know more than we can tell"*. Analogously, there is the example of learning how to ride a bike. Although one could read many books on how to ride a bike, one will never be able to acquire the respective skills before actually practicing it.

While Taylorist workers could be steered on information using input and output rates (e.g., number of briks to be carried from A to Z), knowledge workers need to be steered on knowledge in context (e.g., how does the knowledge created contribute to solve a particular problem?).

Knowledge creation is different as it includes unpredictable and tacit components. The intangible, tacit aspect of knowledge and the growth of knowledge compared to that of an organism (Weiss, 1960) makes it more difficult to produce and control as compared to manual labor. In their seminal theory Nonaka and Takeuchi (1995) explain knowledge creation as embedded in a routine across four modes of conversion between tacit and explicit knowledge: socialization, externalization, combination, and internalization.[1] These are four distinct patterns of human action.

1. *Socialization:* Sharing experiences through direct interaction of individuals (e.g., in meetings, direct communication across team members and teams).

[1] A good account of the these modes can also be found in Choo (2006).

2. *Externalization:* Converting tacit knowledge into explicit knowledge (e.g., documenting).

3. *Combination:* Combining several sources of explicit information into new forms of explicit knowledge (e.g., writing a literature report, reading the application development handbook).

4. *Internalization:* Enabling embodying of explicit knowledge into tacit knowledge (e.g., reading this thesis, capturing known issues and manipulating them through an online system).

In order to enable a successful knowledge creation across a multitude of different functions and teams, several knowledge boundaries have to be considered to enable effective knowledge worker teams. Carlile (2002, 2004) describes three boundaries necessary to consider when managing knowledge: a syntactic, a semantic and a pragmatic boundary. The syntactic boundary addresses the problem of information transfer; this will enable a connection between sender and receiver. The semantic boundary addresses the problem of translating meaning across different functional domains with a potentially different understanding. The pragmatic boundary, also political boundary as called by Carlile (2004), addresses the challenge of potentially conflicting interests of actors when new knowledge is created (e.g., when knowledge created in one project team creates opportunities for an organization but makes adjustments to a second interrelated project necessary). According to Carlile (2004) adequate *boundary objects* are necessary to address their boundaries and enable transferring, translating, and transforming knowledge across involved actors.

Further, in order to understand how knowledge creation is used to create new products and services, we need to understand how organizations decide which knowledge is useful in achieving a specific goal (e.g., which ideas, potential project courses). Choo (2006) proposes the model of the Knowing Organization in which an organization can take action through a cycle of (1) sense making (e.g., what is the product or service to be developed and how does it relate to prior knowledge and the current project), (2) knowledge creation (e.g., what are potential solutions?) and (3) decision making (e.g., what is the right solution?). These are routines that each governing body (e.g., board member) has to execute in order to turn his existing knowledge into governance action in context (e.g., during a board meeting).

2.2 Project-based Organizations and their Governance

> *"A project is a temporary organization to which resources are assigned to undertake a unique, novel and transient endeavour managing the inherent uncertainty and need for integration in order to deliver beneficial objectives of change."* – (Turner & Müller, 2003)

In order to address the issue of increasing knowledge work for some time now researchers occupied with the development of modern organizations observe a trend towards the *projectification* of work (Lindkvist, 2004; Hobday, 2000). This is called the organization of knowledge workers in project teams.

Facing the need to address global markets, increased complexity of products and advances in technology, modern organizations need to hire increased numbers of knowledge workers while staying flexible and effective (see Hatch (2012)). Many modern organizations address this by increasing responsibility and commitment of employees (Hatch, 2012). Heydebrand (1989, p. 337) comments that modern organizations consists of *"specialists, professionals, and experts who work in an organic, decentralized structure of project teams, task forces and relatively autonomous groups."* He observes project-based structures as a focal point of many organizations today.

Whittington, Pettigrew, Peck, Fenton and Conyon (1999) discussed the use of Project-Based Organizations (PBO) as one the most significant changes in large European companies since the 1990s. According to Hobday (2000) Project-Based Organizations are those that use projects as the primary unit of organizing the main functions, the primary business mechanism. Hobday (2000) discusses Project-Based Organizations along a continuum of organizational forms between functional, matrix and rather autonomous project teams.

PBOs' configurations can be placed on a scale from purely functional, via a matrix form, to purely project based organizational forms (Hobday, 2000). Functional organizations have been found to create silos of knowledge (Prencipe & Tell, 2001). Matrix structures have been found inefficient in identifying existing knowledge (Van Den Bosch, Volberda & De Boer, 1999). In program management Lycett, Rassau and Danson (2004), for example, identified the following challenges: (1) aligning projects and the evolving business context, (2) ineffective cooperation and learning between project managers missing synergies across projects, and (3) bureaucratic project management with excessive control focus. Work on governance of knowledge work in Project-Based Organizations has resulted in the observation that different governance strategies are appropriate for different subunits in PBOs (Pemsel & Müller, 2012). PBOs have been found to promote loosely connected islands of knowledge (Lindkvist, 2004). Pemsel and Müller (2012) found that PBOs striving for excellence try to support a collaborative and inclusive culture.

Generally, two streams of literature can be identified in the field of PBOs. First, one that discusses the macro-level, and the increased use of projects in modern economies across various industries (Whittington et al., 1999). Second, one that discusses the micro-level of projectification (Lindkvist, 2004).

Following my research question in this thesis I address the micro-level of projectification. This is here expressed as how individual organizations change towards a project-based modus operandi. Therefore I discuss the following relevant topics. In Subsection 2.2.1 I discuss governance of projects, project governance and governmentality, in Subsection 2.2.2 I discuss

governance of projects and knowledge worker teams, and in Subsection 2.2.3 I discuss the origins of project governance and current challenges.

2.2.1 Governance of Projects and Project Governance

Governance determines rights, responsibilities and the power of actors in a project-based organization. It defines (1) who makes which decisions, (2) how actors make their voices heard and (3) whom they have to give account to. In that sense project governance defines *"the rules of the game"* of project work (Ahola et al., 2013). Too and Weaver (2013) call governance the *mirror of management* as its goal is to set the operating framework in which management can take its decisions.

Governance of projects is distinguished from corporate governance which relates to the structures, duties and obligations of an entire corporation (e.g., including rights and treatment of shareholders, integrity and ethical behavior, disclosure and transparency (OECD, 2004)). As elaborated by Müller, Pemsel and Shao (2014) project governance differs from governance of projects, as the latter addresses governance of multiple projects.

Further, governance of projects distinguishes from governmentality. While governance describes and defines the institutional configuration of roles and responsibilities, the concept of governmentality as proposed by Foucault, Burchell, Gordon and Miller (1991) addresses how the governance task is addressed by different approaches (mentalities), e.g., through soft modes (e.g., values, beliefs) or hard modes of governance (e.g., rules) (Clegg, Pitsis, Rura-Polley & Marosszeky, 2002).

In a recent review Ahola et al. (2013) identify the main streams of research in project governance literature and their origins. They recognize that two distinct streams of literature describe project governance as 'project governance external to any specific project' and 'project governance being internal to a specific project' (Ahola et al., 2013). The former implies that (1) governance is defined outside, and (2) in such way that standardized rules then unidirectionally are imposed on a project. The latter implies (1) a more equal role of projects and project-based firms in creating governance arrangements and (2) the necessity to tailor these arrangements and practices in each project context (Ahola et al., 2013). Similarly to this, Too and Weaver (2013) identify two schools of thought in governance. One school calls for different types of governance for each organizational sub-unit. The other school looks at governance as covering different aspects of a single process (e.g., finance, change people, and their relationships). Further, there is the distinction between (1) large and public sector projects, and (2) different project related levels within an organization, so called *enterprise project management* (Too & Weaver, 2013).

2.2.2 Governance of Projects and Knowledge Worker Teams

Project governance is a rather young discussion (Miller & Hobbs, 2005). While considered as one of the main prerequisites for knowledge worker organizations (Drucker, 1993), organizational learning is recognized as a not well addressed key issue in project-based organizations (see Turner and Keegan (1999)).

Classic project management literature (Kerzner, 2013) acknowledges the uncertain nature and dynamics of knowledge work projects such as R&D. Kerzner (2013), for example, mentions the importance of project management *"..Given that only a small percentage of R&D projects ever make it into commercialization where the R&D costs can be recovered.."* (p.47). However, most project management frameworks do not pay particular attention to the fundamental differences related to the nature of knowledge worker projects and the underlying routines. The prevailing questions is: Why do such projects fail and how to address this challenge.

Multi-project environments have to fight several effects such as collaboration, coordination, and communication (Sharp & Robinson, 2010). Next to traditionally organized hierarchical structures of organizations, projects are today increasingly organized in projectized organizations as programs (grouping related projects) and portfolios (grouping unrelated projects competing for resources) of projects.

According to project management literature program management currently suffers (a) extensive focus on control, (b) insufficient flexibility related to evolving business strategy, and (c) an ineffective cooperation among the projects (Lycett et al., 2004). This lack of understanding is to a large degree caused by the origins of project management in construction; a majority of theory and practice concentrates on single projects in isolation (Lycett et al., 2004) and a hierarchical understanding of organizations. While literature shows some advancement of program and portfolio management, providing a possible perspective, the challenge continues on how to organize and to manage multiple projects under an organizational umbrella.

2.2.3 Origins of Project Governance and Current Challenges

While general governance literature matured across the domains of economics, management and political science (see Levi-Faur (2012)), current project management models have been criticized for not appropriately addressing project reality as (re)shaped by participants (Cicmil et al., 2006). How to get projects organized across several project teams in so called multi-project environments consisting of 3, 50 or 1000 teams proves comparably more difficult as current and prominent examples of large projects show.

As Piore and Sabel (1984) draw out, much of our organizations is aligned towards mass production of goods and the assumptions of management staff based on successful experiences in the past. Classic management literature often assumes that environments can be controlled. The underlying reason for the assumption of classic management models lie in their origins in

the industrial age and the fact that machines are predictable, their output can be fully measured and their design optimized. While this might hold true for machines, it is more difficult for knowledge work. Innovation is difficult to predict.

While the workforce in many organizations today resembles rather what Mintzberg (1979, 1993) calls *Professional Bureaucracies*, thus organizations consisting mainly of highly educated professionals, knowledge workers who have a considerable degree of their own work, they are still governed like a *Machine Bureaucracy*[2].

Research regarding the micro-level of projectification and project governance produced valuable insights. However, little attention has been paid regarding the question on how to govern the dynamics of the knowledge work in project organization. Cicmil et al. (2006) discussed project management as a practice in use, they argued that current project management frameworks do not sufficiently contribute to understanding the projects embedded in context and how the participants continuously reshape the project and its context.

In order to understand governance we must understand its components. As project governance determines the rules, and the responsibilities of actors, we must be able to understand the modus operandi of the organization to be governed. Major models include those proposed by Waterman Jr, Peters and Phillips (1980), Kerzner (2013), and McLeod and MacDonell (2011). The 7s framework as proposed by Waterman Jr et al. (1980) discusses: (1) Structure, (2) Strategy, (3) Systems, (4) Style, (5) Staff, (6) Skills, (7) Superordinate Goals. In the project management literature Kerzner (2013, p.77) proposes: People, Work (Tasks), Tools, and Organization. According to McLeod and MacDonell (2011) there are four main factors influencing the outcomes of individual software product development projects: (1) People and Action, (2) Development Process, (3) Project Content, and (4) Institutional Context. Further, Too and Weaver (2013) distinguish: (1) Relationships, (2) Change, (3) Organizations' People, (4) Finance, (5) Viability and Sustainability.

While these models originated in different domains, they discuss the same components in slightly different configurations. The project governance literature does not discuss how to understand these components in dynamic settings. Further, project governance literature acknowledges the necessity to understand governance influenced by different organizational units (Ahola et al., 2013). However, current contributions focus on goals across the different participants and how those are aligned. A holistic view is missing.

[2] At this point one has to clarify that Project-Based Organizations, in particular those developing new products and services, need to be distinguished from traditional Professional Bureaucracies such as hospitals, law or accounting firms as generally referred to by Mintzberg (1993). As elaborated by von Nordenflycht (2010), organizations developing new products and services can be differentiated as *Professional Service Organizations*, and *Technology Developers* or *Neo-PSFs* in particular. While both rely on knowledge workers, organizations developing new products and services might require dozens of knowledge worker teams working in close collaboration to complete a particular assignment (e.g., the development of a modern mobile phone, a car or a spacecraft). While a doctor requires a highly skilled team to complete a complex medical operation, the development of a new product requires a constant coordination of knowledge and resources across different teams for a long period of time.

2.3 Agility and Agile Project Management

Agile project management methods and the practices applied in project contexts are generally rooted in the context of agile software development (Dybå & Dingsøyr, 2008a). Agile project management methods caused a silent revolution in the way projects are organized and executed (see, e.g., Dybå, Dingsøyr and Moe (2014); Dybå and Dingsøyr (2008a); Abrahamsson et al. (2009)). While originated in software projects, the methods are gaining increased attention in the general field of project management. In 2011, for example, the term *"agile project management"* for the first time surpassed *"agile software development"* on Google Trends. However, the current methods are bound to a sweet spot (Hoda, Kruchten, Noble & Marshall, 2010) of small, co-located software projects and individual teams.

Due to its origins across different scientific domains agility can be difficult to define. The roots of agility in organizations can be traced back across multiple domains including manufacturing and logistics (Booth & Harmer, 1994). Further domains discussing the concept are Business Agility (van Oosterhout, 2010), information systems literature (Conboy, 2009), and sports science (Sheppard & Young, 2006). For example, in Information Systems Development literature Conboy (2009) defines agility as:

> *"the continual readiness of an ISD method to rapidly or inherently create change, proactively or reactively embrace change, and learn from change while contributing to perceived customer value (economy, quality, and simplicity), through its collective components and relationships with its environment."*

The related concept of *Business Agility* is defined by van Oosterhout (2010) as:

> *"Business agility is the ability of an organization to swiftly change businesses and business processes beyond the normal level of flexibility to effectively manage highly uncertain and unexpected but potentially consequential internal and external events, based on the capabilities to sense, respond and learn."*

In the sports science community Sheppard and Young (2006) define agility as:

> *"a rapid whole body movement with change of velocity or direction in response to a stimulus"*

Fueled by the difficulties to define the concept, a recent analysis by Laanti, Similä and Abrahamsson (2013) suggests to look at agility as a set of concrete practices. Also other studies have investigated the usage and perceptions of practices perceived as agile within software development teams (Williams, 2012).

A definition on the organizational level is also challenged. While the application of the methods in individual projects has been widely covered in the literature their application in multi-project environments and the general field of project management outside the software development domain is widely uninvestigated.

Based on my prior research and experiences with the existing literature I will use the following definition:

> *Definition 2.1: "Agile knowledge worker project organizations are those that learn fast and are effective in delivering value."*

2.3.1 Agile Project Management

Agility as a concept to execute and organize software development projects emerged in the 1990s based on ideas found in new product development (Takeuchi & Nonaka, 1986). Agile project management methods such as Scrum are design-oriented (Nerur & Balijepally, 2007). They enable frequent feedback loops based upon recurring project cycles enabling demonstration of intermediate results in so called *iterations*. Compared to traditional plan-driven project management methods they embrace project environments as uncertain and enable an iterative delivery of intermediate project results rather than assuming their predictability and a linear sequence of steps from project definition to delivery (Nerur & Balijepally, 2007).

Dybå et al. (2014) propose the following four principles of agile project management.

1. *Minimum viable product*: Specification of not more than what is absolutely essential and critical to the overall success of the project.

2. *Autonomous teams*: Teams are responsible for managing their work.

3. *Redundancy*: Team members should be skilled in more than one function in order to provide a knowledge overlap and backup.

4. *Feedback and learning*: Learning is an integral aspect of the project, adaptations are preferred.

Agile methods, including the most prominent framework *Scrum* (Schwaber & Beedle, 2001), address this uncertainty through implementing the project in a sequence of iterations (Larman, 2004). According to Larman (2004) an iteration can be viewed as a mini-project, as each iteration contains analysis, design, implementation and testing. Iterations allow for a stepwise delivery of partial results. Each of such releases aims to deliver value to clients through framing such delivery as a *"stable, integrated and tested partially complete system"* (Larman, 2004). This enables a feedback loop with the client(s) and a (theoretically) intermediate

creation of value through the usability of intermediate results (e.g., a webshop with a trivial ordering system allowing initial sales).

So, agile project management differs substantially from the traditional plan-driven project management. Traditional methods predict project deliverables up-front and derive a plan (cost and schedule) based on the assumed features of these deliverables. In project reality, however, the envisioned deliverables often change in scope or are more complex than originally predicted. This often leads to a large scope and delayed schedule. To emphasize the difference of these two project management approaches I illustrate how these two relate to each other in Figure 2.1.

In traditional methods requirements, as depicted by the left triangle in Figure 2.1, the concrete deliverables are fixed at the beginning of a project (e.g., by means of a contract). Based on those requirements costs and schedule are estimated. In agile methods, depicted by the right triangle in Figure 2.1, it is acknowledged that the requirements and the concrete deliverables are likely to change in the course of the project. Such changes in scope can be caused by external influences such as market changes, new technologies or possibilities which otherwise could not have been foreseen at the beginning of the project. In agile project management, the budget and schedule are fixed at the beginning of a project based on the initial scope. It is then communicated that due to the nature of knowledge creation, the exact outcomes cannot be fully predicted. Instead, it is stressed that the customer is an integral part of the project and always in charge of the project priorities through frequent (*iterative*) delivery of the product or service to-be.

Following the illustration traditional project management methods such as PRINCE2 (PRojects IN Controlled Environments, version 2) (Murray et al., 2009) are categorized as *plan-driven* or predictive (Nerur & Balijepally, 2007). Agile methods such as Scrum (Schwaber & Beedle, 2001) are considered to be *value-driven* or *adaptive* (Nerur & Balijepally, 2007).

The development of new products and services cannot be fully planned up front, as acknowledged across the fields of new product development (Takeuchi & Nonaka, 1986), computer science (Fægri, 2010), entrepreneurship and new business development (Sykes & Dunham, 1995). Creating new products and services is a complex problem (DeGrace & Stahl, 1990). According to some authors (e.g., Cynefin framework by Snowden (2005)), a good strategy to tackle complex problems is to do things and test them in practice.

According to the literature (Fægri, 2010) it is easier to foresee intermediate goals and the derived tasks while the longer-term goals remain more difficult to predict.

Agile software development (Dybå et al., 2014; Nerur & Balijepally, 2007) acknowledges that the final software product cannot be predicted all up front. The team moves through unexplored waters (Ramesh, Cao & Baskerville, 2007) and the customer cannot state all his wishes and only in course of the project learns about technical possibilities and what he wants. Further market changes make adjustments to the product necessary. New business development literature (Sykes & Dunham, 1995) recognizes that the strict adherence to a plan can lead to

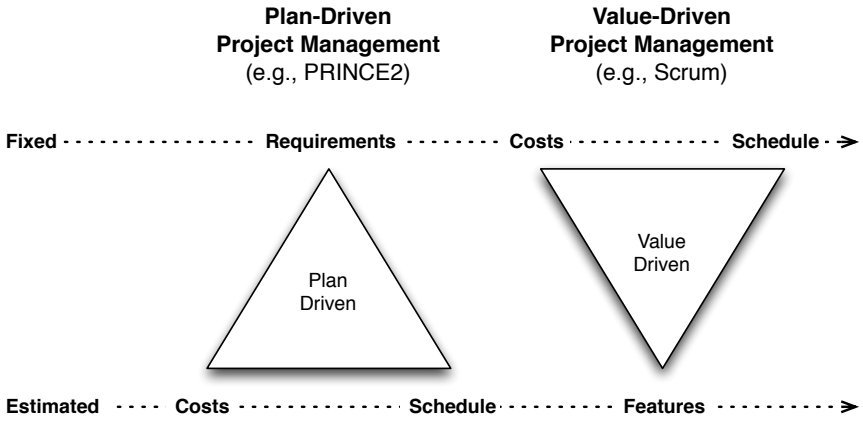


Figure 2.1: Plan-driven vs. value-driven project management (from Leffingwell (2010))

the failure of a business venture and that different more adaptable planning approaches are better suited for the task.

Initial challenges have been reported, regarding the application of agile methods in project-based organizations. The emphasis was on (a) those related to the alignment of business needs and strategy (Hodgkins and Hohmann, 2007; Kalliney, 2009), (b) establishing agile IT project portfolios with prioritization, resource allocation and governance (Rautiainen et al., 2011; Thomas and Baker, 2008) and synchronizing development dependencies (Hodgkins and Hohmann, 2007; Kalliney, 2009). Kalliney (2009) discusses issues concerning the alignment with business strategies and company vision, managing cross-team risks and synchronizing development dependencies as well as handling the knowledge and skill silos of the company. Hodgkins and Hohmann (2007) showed to be aficionados of an agile program management office. They found that Scrum backlogs were insufficient in addressing business needs and introduced roadmaps as a filter to aid backlog prioritization and to communicate strategic intent and business opportunities between the product managers and the technical team. These findings indicate a need for further and more integrated research on agility in project-based organizations.

2.3.2 Governance in Individual Agile Project Teams

Rather than trying to predict the outcome of an entire project up front and following a plan-driven Tayloristic approach, agile methods (Dybå & Dingsøyr, 2008a) advocate a continous reevaluation of the assumptions and vision on the to be delivered product or service. This is implemented through frequent customer feedback anchored in iterative routines. Instead of trying to define everything up front, presenting the deliverable only at the end of the project

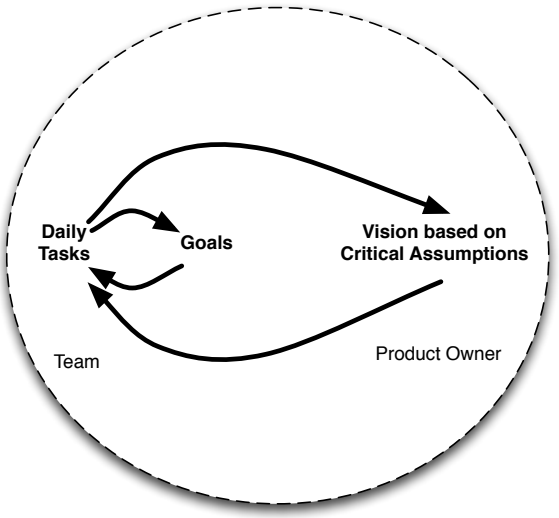


Figure 2.2: A two-level planning cycle as the main governance mechanism in Scrum teams

and hoping that he or she will like it, agile methods such as Scrum (Schwaber & Beedle, 2001) (1) prescribe an iterative delivery where the customer is involved in frequent reviews, (2) observe, and (3) co-create the project outcome.

As illustrated in Figure 2.2 Scrum introduces an iterative delivery of results supported by a two-level planning mechanism: (1) Project/Product-level planning (to capture the full scope of the project and the evolving assumptions), and (2) an Iteration-level planning (more predictable activities within the scope of a few days to the maximum of a month). First, a Project/Product-level planning is conducted with stakeholder representatives setting the initial scope of the project and trying to capture as many tangible requirements as possible. Second, at the beginning of each iteration, the project team selects the items realistically to be delivered within an iteration and created an iteration planning where the items are divided into daily tasks. At the end of each iteration (1) the results are presented to stakeholder representatives, (2) reviewed, and (3) the project-level backlog is (re)prioritized.

Following an emerging metaphor of design, the traditional management cycle of analysis, diagnosis, goal setting, planning and execution is embedded as an iterative routine in each iteration instead of being executed only once throughout the project (Nerur & Balijepally, 2007). The management cycle is repeatedly executed by the team of well educated knowledge workers instead of an individual manager who predicts and controls the team. To maintain a continuous translation across these two levels agile software development methods introduce close feedback loops based on recurring team routines such as review cycles reviewing the product.

In order to maintain this routine Scrum (Schwaber & Beedle, 2001) divide project responsibility across (1) the roles of the team, (2) the product owner and (3) a Scrum master in order to maintain these routines. Project-level planning is facilitated through a Product-backlog by the *product owner* (Schwaber & Beedle, 2001) while the Iteration-level planning is maintained by the project team in a Sprint-backlog. In Scrum the product owner, is a person with genuine interest in the success of the project and authority to make decisions on priorities, while the team is rather autonomous and self-managing supported by a coach/servant team lead (Larman, 2004). In Scrum this routine of frequent review of intermediate results and direct customer feedback is represented by the role of the product owner (Schwaber & Beedle, 2001).

Scrum assumes a predefined project environment including allocated budget, set vision, and available project team. The framework does not address certain aspects of project management such as finance or staffing (IPMA, 2006).

Agile methods have been criticized for not being new (Ågerfalk, Fitzgerald & In, 2006). The concept of overlapping development iterations was, for example, described as "the rugby approach" by Takeuchi and Nonaka (1986) in Japanese product development companies in 1986. However, Scrum for the first time made the ideas accessible to a larger audience through a relatively general and easy applicable framework for software development teams, providing a description of concrete practices as well as the organizational configuration of roles and their responsibilities.

To illustrate the governance mechanism, Figure 2.2 presents the Scum cycle (Schwaber & Beedle, 2001) in accordance with Critical Assumption Planning (Sykes & Dunham, 1995). The longer-term goals outside the scope of an iteration remain more vague and fluid, and are likely to change. Although one could try to predict these higher-level goals and plan the concrete underlying tasks I argue that it is more efficient to update the critical assumptions and the vision rather than moving the focus towards a plan and away from the original assumptions from which the plan was derived.

Through these interleaved feedback loops agile methods limit the uncertainty in knowledge work projects as the product owner through his frequent involvement is in control of the product and development process. Research on agile routines indicates that agile projects are executed in a more structured way compared to traditionally executed projects (Thummadi et al., 2011). In their study on post-adoption usage, Senapathi and Srinivasan (2014) report increased productivity, predictability, and quality of software development.

To summarize, plan-driven methods fix the deliverables in a contract at the beginning of a project and derive the costs and schedule accordingly. Value-driven methods fix the costs and schedule of a project, while providing an estimate on the deliverables, and emphasizing that the project client is always in control of the development priorities and direction of those.

2.4 Research Gap

Following the literature review presented in this chapter I recognize the following gap in knowledge. On one hand there is a lack of understanding of how to govern the dynamics of knowledge work, on the other hand there are techniques that can govern the dynamics across an organization of multiple knowledge worker teams with potentially different objectives and concerns.

Part II

The Knowledge Worker

3

A Rising Number of Knowledge Workers

This Chapter investigates RQ1: *How can governing bodies in organizations of knowledge worker teams understand and steer multiple knowledge worker project teams in practice?* The attempt to provide a potential answer will be supplemented by the answers from Chapters 6-12. In order to make RQ1 suitable for our field research we formulated two subquestions RQ1a and RQ1b. RQ1a: *How do senior managers foresee the change in interpersonal relations in knowledge worker organizations?* RQ1b: *How do senior managers aim to address these changes?* As we see it now, information and communication technology, ICT, has introduced new styles of working. The virtuality of the styles has impact on how employees within an organization perform their task and how they interact with colleagues, clients and supervisors, but so far the impact is not yet well understood. 24 board level executives from 18 Swiss knowledge firms have been interviewed about virtual work arrangements and the change they bring in networking dynamics within their organization. After a careful analysis and discussion we have identified emerging issues and formulated five possible recommendations suggested by our respondents.

This chapter has been pre-published with minor adaptations as the following publication[1]:

Katzy, B. R., Stettina, C. J., Groenewegen, L. P., & de Groot, M. J. (2011, June). **Managing weak ties in collaborative work**. In Concurrent Enterprising (ICE), 2011 17th International Conference on (pp. 1-9). IEEE.

[1]The author would like to thank IEEE and his co-authors for permission to reuse relevant parts of the article in this thesis.

3.1 Collaborative Knowledge Work

A core theme of the our research is collaborative work and its virtual organization. In this chapter we report on a field work study with 24 board level executives from 18 Swiss knowledge firms. In the summer of 2010 we presented a one-page scenario paper (Barzilai-Nahon & Mason, 2010) to the executives (similar to B. Katzy, Zhang and Löh (2005)) and asked the interviewees for their feedback. In order to get unfiltered views, we did not ask structured questions but left the initiative to the interviewee. This resulted in interviews that on average lasted 90 minutes and led to a total of 650 pages of transcripts, discussing to what degree collaborative work has emerged from insider discussions at academic conferences to board room attention.

One recurring theme addressed by the interviewees is the *emergence of new organizational arrangements and their impact on the way individuals perform their day-to-day job*. The driver is a rapidly increasing number of relationships to colleagues, co-workers, clients and supervisors. This chapter adopts Granovetter's (Granovetter, 1983, 2003) perspective on networks as configuration of weak, strong and absent ties as its analytical theoretical lens. The aim of this chapter is an analysis of how these senior managers foresee the change in interpersonal relationships and how the balance will be divided between strongly tied groups and groups based more on the essence of weak ties (see Section 3.2). We review how they perceive the support of information technology for virtual work arrangements. With the fresh memory of assumptions that programs like the European FP 6 Collaborative Work Environment (CWE) program and its projects (e.g., CoVES, COSPACES, or ECOSPACE) made, we aim at contributing to understanding adoption of flexible collaboration environments as well as review requirements for their design and engineering (see Section 3.3).

A second recurring theme addressed by the interviewees is the *perceived higher dynamics or frequency of change in interpersonal relationships and their ties in organizations*. We therefore are particularly interested in the interrelation dynamics between people and how groups based on weak and strong ties shape organizational structures and processes as the interviewees in the initial round of the interviews addressed them. Senior executives, whose responsibility is to create effective work arrangements and organizations, have expressed their interest in the changing nature or organizational arrangements. Based on reviewing how they perceive the support of information technology for virtual work arrangements, we point out how this relates to existing literature (see Section 3.5).

3.2 Increasing Relationships in Knowledge Work

Researchers have repeatedly described pictures of future collaborative organizations (Lipnack & Stamps, 1997; Igbaria, 1999; Igbaria & Tan, 1998; B. Katzy et al., 2005; Schaffers, Brodt, Pallot & Prinz, 2006; Prinz, Jeners, Ruland & Villa, 2009; Sari, Loeh & Katzy, 2010) especially

in response to the adoption of information and communication technology and virtual environments in which time, location and connected devices no longer matter because all are available anytime anywhere. This emergent innovation age affects not only the private space but also businesses. Early signs of this evolution were work-from-home approaches that increasingly develop to more encompassing virtual work styles and work-anytime-anywhere strategies. Suppliers of enabling technology (Microsoft, 2005; SPS, 2010) incorporate these trends in their product and technology roadmaps for virtual workplaces, New Ways of Working, or Work 2.0. Still, there remains doubt on how new ways of work actually affect work effectiveness in practice. With the increasing adoption the need emerges to analyse the implications on practice.

Granovetter (1983) defines three types of such ties that connect actors, or a group of actors in what he calls a clique. Strong ties are bonds that represent contact-intensive and tightly knit relationships that a person has with close friends, family and colleagues. These are opposed to weak ties, which refer to acquaintances with less social involvement, more superficial and on a smaller, less intimate basis. Links in social software platforms fall in this category. Absent ties are between people with no relationship to each other. The argument of Granovetter (1983) is that the main benefit for weak ties lies in the linkage of information between multiple networks. They are ties that can establish a relationship between two strongly tied networks, for example, two departments within an organization, or two organizations via a boundary spanning weak link as summarized in Figure 3.1. An example would be a person within the first network who is acquainted with a person of the other network. This effect is similar to what is frequently referred to as network-broker concept (Fernandez & Gould, 1994; Mowshowitz, 2002; B. R. Katzy & Crowston, 2008) meaning that a specific weak tie forms a bridging factor between two or more socially separated networks.

Weak ties have been found important in transmission of knowledge between coherent groups (Granovetter, 1983, 2003; Maric, 2014), however, research indicates that strong tie networks are considered favorable for solving of complex problems (M. T. Hansen, 1999; M. Hansen, 2013).

Each tie group, be it strong or weak, comes with certain set of components, both static and dynamic, defining the strength of a tie. As proposed in Granovetter's (Granovetter, 1983, 2003) theory there are four components defining the strength of a tie in offline situations: the amount of time spent together, emotional intensity, intimacy, and reciprocal services. Research on the impact of new media on (existing) tie strengths indicates (Haythornthwaite, 2001) that positive effects may be seen for strong ties when the new medium provides further means and opportunities for contact and complementing existing communications methods, or for weak ties when the medium increases connectivity among otherwise unconnected individuals. Where ties are strong, maintained via multiple media, the impact will most probably be small; where ties are weak and maintained via one medium, the impact may be disintegrative for existing connections (Haythornthwaite, 2001).

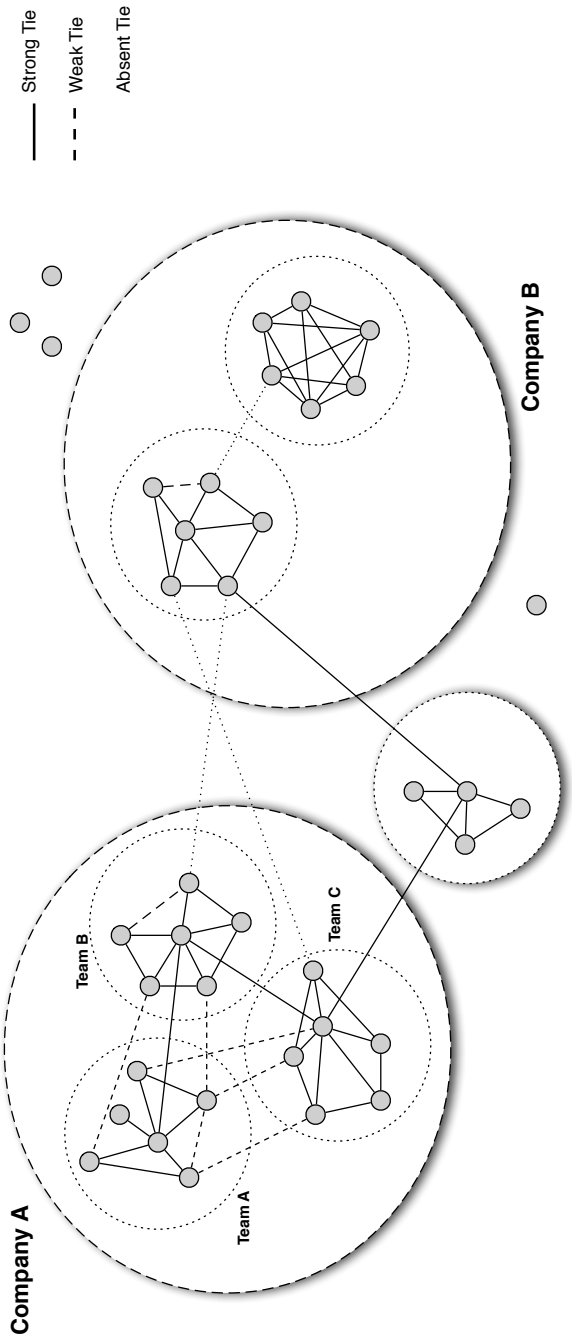


Figure 3.1: Two companies and a freelance team connected via strong and weak ties

Based on the current state of art we thus pose the two following subquestions:

- *RQ1a: How do senior managers foresee the change in interpersonal relations in knowledge worker organizations?*
- *RQ1b: How do senior managers aim to address these changes?*

3.3 Methodology: an Exploratory Study

As we want to understand how senior managers perceive change in interpersonal relationships and ties in organizations in an exploratory manner, we have chosen an interpretive approach to understand social life and the meaning people attach to it (Schurink & Schurink, 1998). Using qualitative interview data, gathered in summer 2010, we used grounded theory (Strauss & Corbin, 1990) to generate our insights into the phenomenon of network dynamics in virtual work environments.

The recorded interview sessions were transcribed, if necessary translated and imported into ATLAS.ti for qualitative data analysis. The analysis of the primary data material occurred in three coding phases: open, axial, and selective (Ghauri & Grønhaug, 2005). Via open coding of the research data we have formulated a series of categories from both literature and from the interview respondents themselves that describe the impact on strong, weak and absent ties as a result of moving towards a virtual work arrangement. The open coded interviews were ranked via axial coding and, where necessary, reduced coding categories to become mutually exclusive and commonly exhaustive. Finally, via selective coding we assessed the relevance of the coding categories and we summarized the common ideology based on the conclusions from all categories used in the coded primary material.

3.4 Characteristics: New Ways of Work in Knowledge Work

The interview coding process led to the definition of 16 unique codes. After a careful analysis and discussion we have chosen four emerging issues and five possible recommendations to be discussed in this chapter. In the following section we will discuss the four emerging issues as addressed by the interviewees.

3.4.1 A movement towards weak tie relationships

The respondents perceive that the relevance of weak ties for knowledge work has increased and has reached relevance that in Granovetter's theory are attributed to strong ties only. Weak tie relationships are considered dependent on the medium of relationships. Platforms such as Facebook and LinkedIn allow these types of relationships to stay longer connected to each

other. *"But what we see is that next generation of people coming in is obviously very addicted to the use of social networks and whatever Facebook and LinkedIn and other sites that have become part of the personal lives and also seem them as part of their professional lives."*, as one of the interviewees remarks.

Through social software and the Internet, knowledge resources are globally accessible and it seems more important to find the right sources quickly than to develop one's own solutions. The respondents perceive that within their organization interpersonal relations shift, making the physical relationships one that maintains with their direct colleagues and supervisors more superficial. One manager expresses his perceptions on the changing relationships, Digital Natives, or the Millennials, yes I think that is their daily bread, how they work together, how they communicate with each other, how they interact. It is some kind of a clash of generations. This has consequences for the design of work environments in that only openness for inter-organizational relationships and exchange will allow for high knowledge work productivity.

3.4.2 Impact of weak tie increase on the infrastructure

In the perception of executives, knowledge workers are aware of the relevance of weak ties for their own professional development while corporate organizational structures are still mostly based on strong ties. This results in perceived loyalty problems for the employer. The respondents perceive that the young workforce just entering organizations has a different work style than the current operative workforce. They are considered to have a more nomadic work style, using different technological tools for communications and collaborations like instant messaging and Facebook. *"They do not use the intranet, if you really want to reach someone, then it must be the Communicator, or at best via Facebook over the private page. We really noticed, if we want to reach them than then we really have to do it via Communicator, SMS or somehow. We need to use their channels."*, states the head of vocational training of a major telecommunications provider.

They are spending time on social media websites to follow and update on private and corporate proceedings. According to the interviewees, this leads to a decrease in loyalty towards the employer, resulting in employees that tend to be more loyal towards their network of peers than towards their network of authority. This forms a dilemma for organizations because employees tend to shun corporate policies for sharing corporate information online with their peers and, in addition, they can easier leave a job or certain position if it suits them. *"I think that is coming. You need a chief social media officer, someone who takes care of these things."*, a HR manager comments.

Moreover, the increase in weak ties affects organizational processes and structures, as they currently tend to be based on strong tie linkage of departments to supervisors to board level executives. The respondents feel that hierarchies as they currently exist in business, no longer work for the next generation of workers who prefer weakly tied alternatives. *"They*

are less hierarchically organized, I believe many companies [...] they are more flat, a bit like Google, more dynamical building workgroups.", says a development manager. A similar topic of discussion among the respondents is that of organizational processes. Similarly like organizational structures, organizational processes tend to be based more on the essence of strong ties rather than weak ties. Meaning that such processes are mostly consisting of a static fixed set of steps/activities in order to get from input to output.

3.4.3 Weak ties require different working models

Work processes based on weak ties require different working methods as traditional process management is largely based on strong ties. Concurrent and agile engineering models (Dybå & Dingsøyr, 2008a) could serve as an example for such working methods. What emerged in the engineering domain is perceived to be applicable for knowledge work at large, in different industries as well as in different functional departments. Moving from concurrent engineering into concurrent enterprising, future research needs to generalize theory on knowledge work.

The respondents believe that offering virtual tools and incentives help them being more attractive as an employer towards knowledge workers, whether it's (1) the allowance of social media websites, (2) nomadic or flexible work styles or (3) other forms of virtual collaborative incentives. According to the respondent they believe that knowledge workers want to be provided with such tools that help them facilitate weak tie investments. One manager states, "*Well, we believe we have learned from those and we also believe that there are people like young engineers and like they talk about who entered a company and that is the normal way of organizing their social life, so you cannot just shut that down for them. We would not be an attractive employer if we do that.*" Young knowledge workers tend to appreciate such technological tools as virtual work incentives in such a manner, that they are willing to pay a premium in the form of a job function and salary to get into the environment that does offer them such benefits. The respondents state, for example, that the next generation workforce tend to use their own type of privately downloaded tools or online search engines to get information or perform steps in a process cycle rather than the ones paid and maintained for internally. *"If you compare it to a period of 20 years ago without doubt, without doubt, they are using different kind of IT tools. They are used to look for information on the Internet.*", one manager responds.

3.4.4 Changing structural dynamics of weak ties

Weak ties introduce dynamics that strong ties do not have. For example they are active for short periods of time only. The regular habit of switching (Mowshowitz, 2002) and favoring of temporary relationships is based on explicit rather than implicit agreements. Knowledge work environments, thus, not only are perceived to increase efficiency of pre-structured processes, but also need to focus in particular on supporting the new organizational dynamics, unusual in

their virtual as well as in their flexible character. While Granovetter's approach prepares for the structural dynamics, it does not model dynamics explicitly.

Overall the respondents perceive that frequency will increase through rapid collaboration between individuals when operating on a virtual collaborative basis. *"Collaborating and online collaboration will have a very important role to people that are in fixed locations and try to work more and more from home or wherever they. We will spend less time in our offices and a large number of cases can collaborate in this context [...] and they will have to develop skill sets and how to keep in touch with their colleagues and enter the projects that they are working on."*, as one participant responds. Within virtual work arrangements the means to collaborate across time and space has allowed to quickly connect and disconnect between partners because transaction costs will be lower (Neus, 2001).

Now we can answer RQ1a: *"How do senior managers foresee the change in interpersonal relations in knowledge worker organizations?"*. From our analysis above, we may conclude, that senior managers perceive (1) a move towards weak tie relationships, (2) shift away from hierarchies towards more dynamic work groups, and (3) they perceive the necessity to adapt ways of working.

3.5 Discussion and Recommendations

In order to manage the increase of weak tie relations and their structural dynamics the respondents mention a variety of options. In this section we summarize the emergent options as applied by our respondents into five concepts in alignment with existing theories.

3.5.1 Organizational Openness

'Open' organizations tend to aim at sharing knowledge both internally and externally thus brokering knowledge among individuals and inter-company networks. The respondents perceive that the increase in weak ties will lead organizations to facilitate knowledge exchange among stakeholders. We have seen that knowledge workers tend to do something they 'like' and something that motivates them. One respondent initiative lets employees submit ideas via an idea management which is well received by employees, *"The idea management we have works in this way, you basically enter your idea and everybody sees that idea and can actually act on it. They are like oh thats good, I like that. And I will help if you try to do that and there is a board looking at those ideas regularly which goes like oh this is a small idea, this is a major idea."* Thus engaging knowledge work via employee-based approaches with an open style of communication letting, them develop ideas on their own in entrepreneurial spirit, is a possible opportunity.

3.5.2 Result Driven Agile and Concurrent Development Models

Result-driven management and goal-orientation focus on the end deliverable rather than the process. Tiwana and Keil (2004) found synergies between result driven management and high levels of principal knowledge, stating that only such control can be enforced across corporate boundaries as opposed to process control, which cannot. Thus, in their theory only a result driven approach seems feasible when working via virtual work styles which crosses corporate boundaries. *"..if you want to work modern or mobile, then it has to, it needs a different leadership style. It needs a culture of trust and you must lead goal-oriented. Say ... ultimately the results are important, and not on the input or control, or how it was done, but the result must be there. And where the outcome is developed and how, I as a superior should not care."*, notes a HR manager.

Current work processes are still largely based on hierarchies and traditional strong tie command-and-control structures. With the shift toward organizations with dominant weak ties, more flexible work models with an emphasis on people are necessary. Agile development models, for example, aim to replace command-and-control management with collaborative self-managing teams (Moe, Dingsøyr & Dybå, 2010; Stettina & Heijstek, 2011a). Virtual work styles come with the opportunity to access information anytime anywhere, driving concurrent engineering methods by enabling access through computer aided engineering platforms.

3.5.3 Leveraging Social Media in Corporate Processes

Social media offer platforms where individuals form and maintain weakly tied networks with peers sharing information both professional and private in digital form. Using these technologies for business purposes seems farfetched, alternatively they can bring organizations a way to manage their growing weak tie networks and help leveraging collective intelligence. *"I mean 15 years ago we have to force all the partners to learn how to email, how to use Outlook how to use the calendar in order to have a system in place which is quite effective. [...] I think with the new system in turn that we have in place, with the new wave of communication, well have to make sure that everybody knows of this, an application which we have deemed as important to be efficient."*, remarks one manager. The widespread proliferation of web tools such as wikis and blogs helped to popularize the idea of collective intelligence (Lévy & Bonomo, 1999; Woolley, Chabris, Pentland, Hashmi & Malone, 2010).

3.5.4 Job Rotation

For leveraging internal knowledge, one of the companies applies job rotation. *"Someone here is a boss for a day [...]. That person is responsible [...]. It rotates; it is someone else each day. [...] Everybody is the boss once. It has proved its value very well."*, as one R&D manager states. According to some researchers job rotation can lead to organizational benefits

such as accumulating more individual human capital and leading to more knowledge aware organizations where workers are more generalists then specialist (Ortega, 2001; Eriksson & Ortega, 2006; Fægri, Dybå & Dingsøyr, 2010). Appearing as "multiskilling" in socio-technical literature (Emery, Thorsrud, Engelstad, Gulowsen & Qale, 1976), job rotation can further contribute to improve knowledge redundancy by integrating knowledge from different domains (Fægri et al., 2010). This can help organizations to find replacements when a certain individual leaves the organization and help to establish a more dynamic workforce.

3.5.5 Dynamic Structures and Self-Management

The respondents perceive that hierarchies, as they currently exist in business organizations, will lose their importance with the next generation of workers. To facilitate organizational structures based on weak ties, which are more dynamic and are more likely exposed to change, literature suggests dynamic network structures. In such organizations individuals act as nodes in a network, working together for a common purpose (Lipnack & Stamps, 1997).

Teamwork research suggests that leadership should be transferred accordingly to the key knowledge, skills and abilities necessary at a moment in time (Pearce, 2004) and shared leadership could function as one possible management style. Self-organizing project teams (Takeuchi & Nonaka, 1986; Stettina & Heijstek, 2011a) have been found successful while studying product development projects, and have been found to have high productivity and increased speed in problem solving in practice (Tata & Prasad, 2004; Fægri et al., 2010).

We can now answer RQ1b: *"How do senior managers aim to address these changes?"*. From our analysis above we may conclude, that our interview participants discuss the following options: (1) Organizational Openness, (2) Result Driven Agile and Concurrent Development Models, (3) Leveraging Social Media in Corporate Processes, (4) Job Rotation, and (5) Dynamic Structures and Self-Management.

3.6 Chapter Conclusions

As a partial answer to RQ1: "*How can governing bodies in organizations of knowledge worker teams understand and steer multiple knowledge worker project teams in practice?*", we would like to put forward the recommendations provided by our participants: (1) Organizational Openness, (2) Result Driven Agile and Concurrent Development Models, (3) Leveraging Social Media in Corporate Processes, (4) Job Rotation, and (5) Dynamic Structures and Self-Management.

In this manner we presented the results of a naturalistic study conducted with 24 board level executives from 18 Swiss knowledge firms. We found that the executives perceive a growth of interpersonal ties, and that they perceive implications on organizational structures and working models. To provide a theoretical basis we link their perceptions to the existing

theory of strong and weak ties as proposed by Granovetter (1983). As those findings alone provide little advice to practice, we furthermore provide recommendations as applied by the participants in alignment to current literature.

Executives perceive that the relevance of weak ties for knowledge work has increased and bypasses the relevance of strong ties. As weak tie relationships become more numerous and more influential, knowledge workers are aware of the relevance of weak ties for their own professional development. Harmonization of work and personal life makes quality of work more important for the preferences of the individual. Thus, personal values become equally important as those of the company. Many observe that knowledge workers give priority to their network relationship over loyalty to the employer. The war for talent in a knowledge society is fought in terms of attractiveness, which in turn determines the success of an organization.

Work processes based on weak ties require new adaptive working models such as lightweight processes in concurrent engineering or agile software development. What has emerged in the engineering domain is perceived to be applicable for knowledge work at large, in different industries as well as in different functional departments. There is a lack of organizational theory in organizing weak ties and future research needs to generate theory for knowledge work and in particular for specifying the interrelation dynamics located in weak ties, thereby doing justice to both its virtual and its flexible character.

Part III

The Knowledge Worker Team

4

Knowledge Workers and Five Dimensions of Agile Teamwork

This Chapter investigates the research questions RQ1: *How can governing bodies in organizations of knowledge worker teams understand and steer multiple knowledge worker project teams in practice?*and RQ2: *What are the components of governance necessary to understand knowledge worker project organizations?* In order to make RQ2 suitable for our fieldwork we formulated the following subquestion: RQ2a: *To what extent can we use the findings by Moe et al. (Moe, Dingsøyr & Røyrvik, 2009) to measure self-reflection in agile teams?* Our provisional answer will be supplemented later by answers from Chapters 5, 8, 10, 11, and 12. To address the research questions I discuss the characteristics exhibited by agile teams and propose a tool to foster self-reflection in agile project management teams. We contribute a quantitative questionnaire organized along five dimensions of agile teamwork. To test this survey tool and its alignment with existing studies, we have executed an empirical validation of the tool with 79 individuals and 8 international Scrum teams. We find that inter-team agreement on the factors is high and that the survey tool is found very useful.

This chapter is based on the following publication[1]:

Stettina, C. J., & Heijstek, W. (2011). **Five agile factors: Helping self-management to self-reflect.** In Systems, Software and Service Process Improvement (pp. 84-96). Springer Berlin Heidelberg. [1]

[1]The author would like to thank Springer and his co-author for permission to reuse relevant parts of the article in

4.1 Replacing Command and Control

With the introduction of agile methods such as Scrum (DeGrace & Stahl, 1990) and Extreme Programming (XP) (Beck & Andres, 2004), the emphasis on people and their integration into the organizational process of software development has become increasingly important. Scrum, as an adaptive and empirical process, for example, aims to replace *command-and-control* management with *collaborative self-managing teams* (Moe, Dingsøyr & Røyrvik, 2009).

Takeuchi and Nonaka (Takeuchi & Nonaka, 1986) found self-organizing project teams successful while studying product development projects in large Japanese companies. Since then, they have been identified with high productivity and increased speed in problem solving (Guzzo & Dickson, 1996; Tata & Prasad, 2004). While innovative projects often feature self-managing teams, self-management is often considered difficult to implement. Human and social factors are important in agile projects (compare (Beck et al., 2001)). Frameworks such as Scrum explicitly pay emphasis to social and cultural aspects of software development which was not the case in previous frameworks. The related changes in company culture and the awareness necessary identified as difficult to adapt in practice (Dybå & Dingsøyr, 2008b; Moe, Dingsøyr & Røyrvik, 2009; Moe, Dingsøyr & Dybå, 2009)

While more embedded within the process surrounding software development than the pure function of writing code, agile teams are exposed to organizational barriers to a greater extent. In traditional and plan-driven *command-and-control* environments there exists a clear separation of roles, driven by self-managing professionals. In collaborative self-managing teams instead it is more important that team members understand individual barriers as well as organizational level barriers (Moe, Dingsøyr & Dybå, 2009). We aim to improve understanding of these barriers in this contribution.

4.2 Objectives

Implementation of agile and self-organizing teams can be aided by increased development team self-awareness. In order to protect themselves from management, agile teams have been observed to give the impression that the team is better than they were (Moe, Dingsøyr & Dybå, 2009). This *impression management* (Moe, Dingsøyr & Dybå, 2009) has been mentioned as a reason for failure to learn and change operating modes inside agile teams, preventing key issues of the process to be addressed.

In this chapter we propose a tool for self evaluation and reflection of agile software development teams. This tool is developed from a questionnaire. We aim to improve self-reflection in agile teams. To this end, we developed an instrument to provide a comparable and practical measure for team members and feedback for self-reflection based upon the study

this thesis.

design and findings of Moe et al. (Moe, Dingsøyr & Røyrvik, 2009). We pose the following additional research question:

> *RQ2a: To what extent can we use the findings by Moe et al. (Moe, Dingsøyr & Røyrvik, 2009; Moe, Dingsøyr & Dybå, 2009; Dybå & Dingsøyr, 2008b; Moe et al., 2010) to measure self-reflection in agile teams?*

Our objective is to promote the discussion inside agile teams through the adoption of an impersonal survey tool in order to understand adequately the mechanisms of effective teamwork and organizational requirements.

4.3 Related Work

In contemporary psychology, the *Five Factor Model* (FFM) also known as the *Big Five Personality Traits* is a model describing human personality through lexical analysis. The five factors were discovered and defined by factor-analyzing hundreds of measures of known personality traits (Digman, 1990).

Dybå and Dingsøyr conducted a structured literature review on empirical studies to address the scientific level of evidence behind agile software development methodologies identifying 36 out of 1996 studies matching their criteria (Dybå & Dingsøyr, 2008b). To examine teamwork in agile software development teams the group developed five dimensions of agile teamwork (Moe, Dingsøyr & Røyrvik, 2009) building up on work of Salas et al. (2005). They have placed their dimensions based upon a set of open-ended interview questions within an action research program with companies applying Scrum. Moreover, they evaluated their qualitative design conducting interviews with all team members in three longitudinal projects (Moe, Dingsøyr & Røyrvik, 2009). In the scope of the three years lasting program they found the absence of redundancy and the conflict between team level and individual level autonomy as one of the biggest barriers in implementing self-managing agile teams (Moe, Dingsøyr & Dybå, 2009).

This instrument as originally developed by Moe, Dingsøyr and Røyrvik (2009) consists of the following five dimensions: *shared leadership*, *team orientation*, *redundancy*, *learning*, and *autonomy* as outlined in table 4.1. They have developed set of open-ended interview questions for each of the dimensions to be conducted with all respective members of a Scrum team. They build their five dimensions of agile teamwork on theoretical and empirical ground (Dybå & Dingsøyr, 2008b; Moe & Dingsøyr, 2008; Salas et al., 2005; Moe, Dingsøyr & Dybå, 2009). Their qualitative questionnaire forms the basis of our quantitative research design.

4.4 Method

With the goal to promote understanding and self-reflection on organizational and team level barriers (Moe, Dingsøyr & Dybå, 2009) from a development unit's perspective, we provide an instrument to be applicable from within the team. To simplify the collection process, we have thus developed an anonymous questionnaire to promote more objective answers.

To reduce bias we encouraged the team members to provide their honest opinions by emphasizing the anonymous treatment of data. No results other than the processed outcome for the whole team would be distributed or given to their superiors. While we provided personalized links for each team member to ensure the consistency of input, no personal details were stored or used within the examination. Furthermore, some of the questions would only strengthen the agile factor when disagreed upon. This prevents a high ranking when answering all questions positively.

To increase transparency of the data, we documented the level of agreement and, the variance of answers given by the team members. This should help pointing at inconsistency within the team.

4.4.1 Questionnaire Design

To enable data collection via online surveys we adapted the qualitative questions by Moe et al. (Moe, Dingsøyr & Røyrvik, 2009) into a quantitative design. The question sentences (table 4.1) have been held as close as possible to the original design. A screen shot of the online questionnaire page can be found in figure 4.1.

The questionnaire has been changed by adding "I feel" at the beginning of each sentence. This has been done to enable team members to identify themselves in a better with the research while keeping a comparable measure to the original findings. Then, for each of the questions the participants were given a standard Likert scale to express their perceptions. To prevent inconsistency among the rating items we used a standard Likert scale consisting of 5 items: *Strongly Agree* = 5, *Agree* = 4, *Neutral* = 3, *Disagree* = 2, *Strongly Disagree* = 1.

4.4.2 Team Agreement

Variance (σ^2) is a measure of how far each value in a set of responses is from the mean. Variance is a useful measure for the level of agreement within a team, based on our survey, because variance is proportional to the scatter of the response metrics and independent of the number of responses.

The variance is defined as

$$\sigma^2 = \frac{\sum(X - \mu)^2}{N} \tag{4.1}$$

Table 4.1: Five dimensions of agile teamwork and related personal questions for the agile team radar as inspired by Moe et al. (2009)

Shared Leadership

Creation and maintenance of the team's shared mental model and transfer of leadership according to key knowledge, skills and abilities, shared decision authority

- I feel everyone is involved in the decision-making process
- I feel team members make important decisions without consulting other team members
- I feel the team vision is well defined and presented
- I feel the team is designed (and redesigned) according to its purpose

Team Orientation

Promotion of team cohesion counteracts social loafing and increases individual responsibility, team goals are given priority over individual goals

- I feel the team takes into account alternative suggestions in team discussions
- I feel the team values alternative suggestions
- I feel team members relate to the tasks of individuals
- I regularly comment on a co-worker's work

Redundancy

Cross-functionality avoids bottlenecks and enables possibility to shift workloads and mutual assistance

- I feel it is easy to complete someone else's task
- I feel I get help if I get stuck
- I help others when they have problems
- I feel it is easy to substitute a person if someone leaves the team

Learning

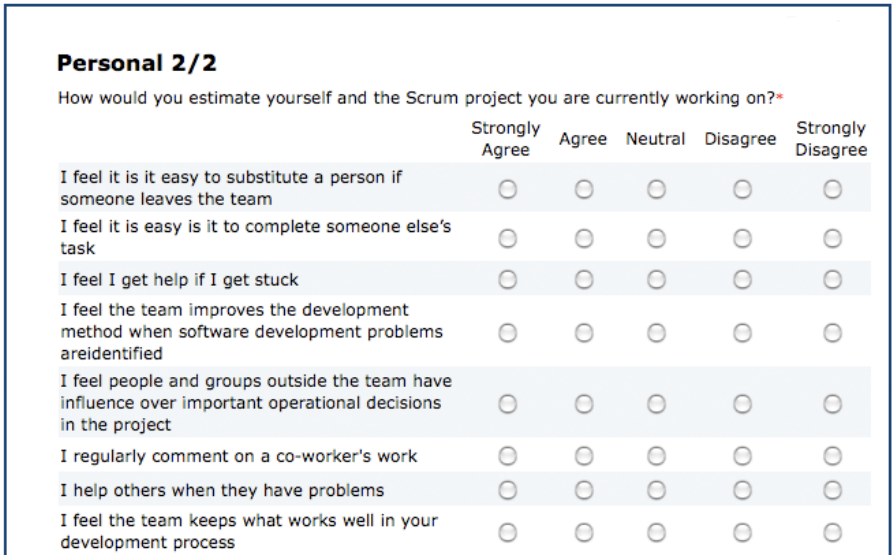
Interdisciplinary knowledge acquisition to promote self-optimization in a wider environment

- I feel the team keeps what works well in the development process
- I feel the team improves the development method when software development problems are identified
- I feel the team gives feedback on all aspects of each others work

Autonomy

External influences on the activities of the team, a precondition for self-management. Although sometimes beneficial, such influences can discourage group thinking.

- I feel the team loses resources to other projects
- I feel people and groups outside the team have influence over important operational decisions in the project
- I feel decisions made by the team are respected by people and groups outside the team



Personal 2/2

How would you estimate yourself and the Scrum project you are currently working on?*

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
I feel it is it easy to substitute a person if someone leaves the team	○	○	○	○	○
I feel it is easy is it to complete someone else's task	○	○	○	○	○
I feel I get help if I get stuck	○	○	○	○	○
I feel the team improves the development method when software development problems areidentified	○	○	○	○	○
I feel people and groups outside the team have influence over important operational decisions in the project	○	○	○	○	○
I regularly comment on a co-worker's work	○	○	○	○	○
I help others when they have problems	○	○	○	○	○
I feel the team keeps what works well in your development process	○	○	○	○	○

Figure 4.1: Online Questionnaire

where μ is the mean.

A lower variance therefore corresponds with a greater level of agreement within a team. The maximum variance in a team is the variance of the maximum and minimum values that can be given in response to an answer. The minimum variance is 0, denoting complete agreement. On a Likert scale from 1 – 5, the maximum variance is

$$\sigma^2 = \frac{\sum(X-\mu)^2}{N} = \frac{\sum X^2}{N} - \mu^2 \tag{4.2}$$

$$\text{MAX}(\sigma^2\{1,5\}) = \frac{1^2+5^2}{2} - \left(\frac{1+5}{2}\right)^2 = 4 \tag{4.3}$$

4.5 Results

To test our questionnaire and to inquire its matching to existing research, the questions have been presented to a group of international project teams, practitioners, and experts applying the Scrum methodology. The instrument has been provided to the participants as a set of online survey questions in random order. The participants had to be actively involved in a Scrum development team. All questions had to be answered in order to count the respective data set as valid.

To look for international Scrum teams interested in the study, one of the authors searched Scrum/Agile oriented groups within business related networks. After identifying related

individuals from different online community platforms, user groups as well as originating from direct and indirect contacts, the author sent invitations for participation to 150 Scrum related professionals. Those who were interested in participation received an anonymized link allowing identification of teams within the online survey system. In addition each ScrumMaster of a potentially interested Scrum team received a set of open-ended questions regarding the project environment.

After data collection, the given answers were accumulated into global and team views as shown in table 4.2 and figure 4.2. The total number of valid data sets collected contains 79 individuals and 8 teams from 13 countries. The teams are called T1, T2, ... T0. Most of the participants belong to the group of software developers (47%) and ScrumMasters (18%). Other groups, however, emerged within the data collection phase. Their data has been taken into evaluation as long as the individuals were committed to a *Pig* role within the Scrum project: Product Owner (8%), Quality Assurance (6%), Agile Coach (6%), Consultant (9%), Interaction Designer (1%), CTO (5%). The gross amount of relevant working experience among the participants is situated around a work record of 1-5 (38%) and 6-10 (29%) years.

After primary analysis, the author decided on 8 teams to be taken into team analysis. The teams had to consist of at least four members with, depending on the team size, at least two-thirds of the team having answered the survey in order to represent a consistent group image. The remaining survey answers were only analyzed globally.

4.5.1 Team View

Table 4.2 contains each team's self-assessment scores based on the Likert scale data from the questionnaire as mean values for each team.

The minima reveal a consistency towards the dimension of autonomy and there is a noticeable tendency towards learning among the maxima. Autonomy consistently earns the smallest score for all teams, while learning is the highest perceived characteristic for half of the eight teams and changes between redundancy and team orientation for the other half. The results show a similar trend in distribution as those presented in the original findings by Moe, Dingsøyr and Røyrvik (2009).

Team agreement, expressed by variance (σ^2) is mentioned in table 4.2 below the aggregated team level measurements. We observe a pretty high (0.06-0.33) level of agreement within the teams as represented by a fairly low variance. Also, the agreement on the five factors is pretty high (0.18-0.20). Teams agree least on redundancy and shared leadership and most on team orientation and autonomy.

The consistent low rating on low autonomy and high agreement is a pointer to organizational level barriers and can be tracked back to our first two questions for the factor *autonomy* (table 4.1). The least agreement on redundancy can be a pointer to contended ideas regarding specialization in agile teams. Many participants reacted skeptical towards the implementation

Table 4.2: Descriptive variables, radar results (x) (**min** & **max**) and agreement (σ^2)

		T1	T2	T3	T4	T5	T6	T7	T8	AVG. AGR.
country		UK	US	UK	NO	NL	SE	IN	NZ	
team size (pers.)		4	9	5	12	6	4	8	6	
collected answers		4	6	5	6	5	3	8	4	
avg. exp. (yrs.)		7.75	13.7	6.6	12.7	2.6	10	7	3.5	
shared leadership	x	**4.13**	**3.83**	**3.90**	**3.83**	**3.10**	**3.17**	**3.59**	**3.69**	
	σ^2	*(.05)*	*(.08)*	*(.29)*	*(.47)*	*(.06)*	*(.22)*	*(.08)*	*(.36)*	*(.20)*
team orientation	x	**4.56**	**4.21**	**4.15**	**3.88**	**3.30**	**3.83**	**3.69**	**3.88**	
	σ^2	*(.14)*	*(.15)*	*(.27)*	*(.34)*	*(.01)*	*(.06)*	*(.09)*	*(.39)*	*(.18)*
redundancy	x	**4.38**	**3.67**	**3.85**	**4.10**	**3.30**	**3.67**	**3.94**	**3.25**	
	σ^2	*(.08)*	*(.22)*	*(.14)*	*(.16)*	*(.32)*	*(.18)*	*(.28)*	*(.22)*	*(.20)*
learning	x	**4.58**	**4.22**	**4.20**	**3.50**	**3.33**	**3.56**	**3.58**	**3.75**	
	σ^2	*(.02)*	*(.25)*	*(.03)*	*(.32)*	*(.36)*	*(.25)*	*(.13)*	*(.19)*	*(.19)*
autonomy	x	**3.50**	**3.61**	**3.27**	**3.17**	**3.07**	**2.78**	**3.13**	**2.92**	
	σ^2	*(.03)*	*(.16)*	*(.24)*	*(.33)*	*(.32)*	*(.18)*	*(.11)*	*(.08)*	*(.18)*
AVERAGE AGREEMENT	σ^2	*(.06)*	*(.17)*	*(.19)*	*(.33)*	*(.15)*	*(.18)*	*(.08)*	*(.25)*	*(.18)*

of cross-functionality. Although being aware of the "quagmire" effect of specialization (Moe, Dingsøyr & Dybå, 2009), many could not think of how to overcome the idea as a waste of resources.

Members of T1 (UK) and T7 (India) agree most. T1 (UK) provides a back-end software for a major Massive Multiplayer Online (MMO) game publisher. T7 (India) worked on a e-commerce solution. Both were collocated development teams with similar roles and good team consistency. Members of T4 (Norway) and T8 (New Zealand) agree least. T4 (Norway) is employed by a company providing smart card based public key solutions for security transactions, consisting of developers from two separate locations running "several parallel projects". Team T8 (New Zealand) consists of a business analyst, a quality assurance specialist and two developers working for a state insurance agency. Both are rather diversified teams with different roles. T4 and T8 have a notably increased variance for shared leadership and team orientation, while T1 and T7 agree on those. Although the level of agreement does not reflect on agile values, it indeed seems to correlate with the consistency of the teams.

4.5.2 Global View

To have a more detailed view on the data we have compiled a global team radar consisting of the answers of all 79 participants, as depicted in figure 4.2. In consistence with the findings of

Moe et al. (Moe, Dingsøyr & Dybå, 2009) we can find the organizational and individual level autonomy as well as redundancy as the factors with the lowest global values.

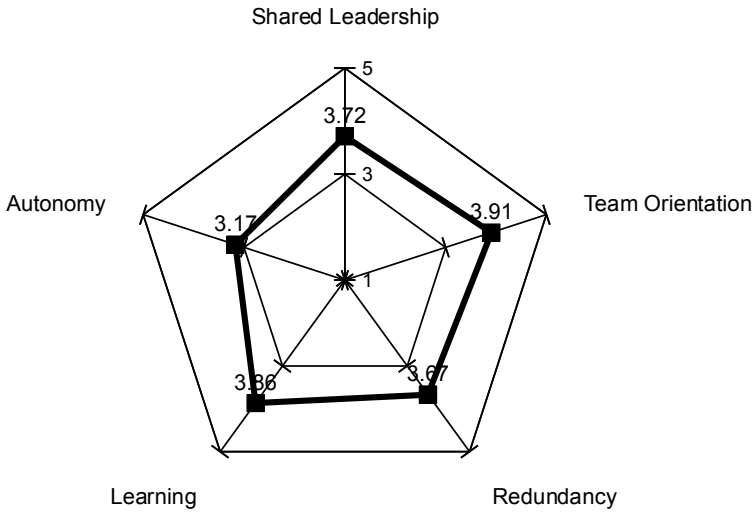


Figure 4.2: Global Team Radar

4.5.3 Validity Considerations

Due to the low amount of data sets containing 79 individuals, conclusions should be drawn carefully. Here, we see stressed particular attention to the quality of collected data. Throughout the whole process of data collection we encouraged participants in giving realistic answers and emphasized the anonymous treatment of data to establish a reasonable level of trust. Only complete data sets and teams with a minimum amount of participants were taken into evaluation.

Quantitative data collection typically grounds evidence on big data sets. As we base our evidence on small team data sets instead, we have improved the transparency of data by adding the variance of given answers among the team members.

The distribution of given answers reveals an expected bias of participants towards positively perceived answers. In psychology this effect is being referred to as Socially Desirable Responding (RDS) (cf. Paulhus (2002)). This effect can be lowered by anonymous self-administration, meaning that when the subjects' personal details are not required, the person does not feel directly and personally involved in the answers he or she is going to give. The second provision we applied to reduce social desirability, is the self-administration of the survey through a computer. The self-administration of the survey through a computer neutralizes here social

desirability through the impersonality of the machine.

4.5.4 Discussion

The findings of this study are twofold: (1) the questionnaire and team radar are perceived a useful tool to support self-reflection, and (2) while steam orientation is perceived as strong, the organizational embedment disturbs the effectiveness of teams through limited autonomy.

The noticeable tendencies reveal a global minimum for measured autonomy and maxima for learning and team orientation. The data suggests that Scrum teams seem not to be well prepared, to cope with the cultural environment existing in their companies in order to maintain a level of autonomy required to apply the methodology. It seems that the internal factors are indeed supporting Scrum within a team, by the willingness to share leadership and good team orientation - values which arguably might be perceived as being passive and existing within small development teams with divided roles anyway. Redundancy, scoring the second lowest value here, resulting in a lack of cross-functionality could in fact create a breeding ground for interpretations that some teams are not actively prepared for a faithful implementation of Scrum. This strengthens the need of further involvement of developers in discussions regarding implementation of agile processes.

The application of our questionnaire was met with interest. This is also reflected by the relatively high response rate: 79 respondents out of circa 150 inquired professionals. During data collection we received questions and suggestions from participants, especially from those with most experience in application of agile methods. However, it was not always easy to collect consistent data from a whole team, and thus out of 79 participants just 8 teams could be taken into team analysis. This might be partially caused by the invitation offered through superiors. In two cases there was direct interest of clients or *Product Owners* with an offshore development team. In this case the contractors were assumed to be interested in learning about the consistency of the hired development team. This attitude led to poor commitment to the survey. Data collection should be motivated by the desire to learn and improve inside the team and should not be used by means of organizational control. Commitment thus is to be expected when executed on team initiative.

We can now answer RQ2a: *"To what extent can we use the findings by Moe et al. (Dybå & Dingsøyr, 2008b; Moe, Dingsøyr & Røyrvik, 2009; Moe, Dingsøyr & Dybå, 2009; Moe et al., 2010) to measure self-reflection in agile teams?"*. From our analysis above we may conclude, that the findings by Moe et al. (Dybå & Dingsøyr, 2008b) can be be applied as a survey based tool to improve self-reflection in agile teams.

4.6 Recommendations

After data collection we have been repeatedly asked by the teams and *ScrumMasters* for recommendations with respect to the findings as during the design of the study we did not think of recommendations. As the five factors alone provide a comparable measure but little practical advice to the audience we would like follow-up on this. In the following section we thus provide advice for each of the five factors from current literature.

4.6.1 Shared Leadership

Literature argues that leadership should be transferred accordingly to the key knowledge, skills and abilities necessary for a particular issue at a moment in time (Pearce, 2004; Hewitt & Walz, 2005). A team leader's task as argued by Salas et al. (Salas et al., 2005) therefore should be the creation and maintenance of the team's shared metal model while the teams collaborative process. Moe et al. (Moe, Dingsøyr & Dybå, 2009) give the example of a "chief architect" on one hand and of a newly hired developer on the other: while the chief architect took over most of the decisions in one company, leading to frustration of team members, a newly hired software developer had to fight for attention in another company. Team members should share decision authority to promote commitment (Takeuchi & Nonaka, 1986). Communication plays an important role here and the common goal should be known and respected within the team and organization (Moe, Dingsøyr & Dybå, 2009).

4.6.2 Team Orientation

This dimension can be directly found in the framework of Salas et al. showing improved individual effort and performance. Lack of team orientation leads to demotivation, social loafing, diffusion of responsibility, and sucker effects, thus lowering the cohesion of the team (L. Thompson, 2002). Moe et al. (Moe, Dingsøyr & Dybå, 2009) have found out that team members gave a too high priority to individual goals rather than team goals. Shared team orientation promotes cohesion of the group and counteracts social loafing as team members perceive that the task and the team itself is important (Karau & Williams, 1993). Organizations (Walton & Hackman, 1986) with greater influence of task skills as well as rewarding systems for team performance increase team cohesion and team orientation (G. Shea & Guzzo, 1987). Job rotation and a culture of trust in collaboration can help to improve this and cross training can be valuable by increase the team's flexibility (Moe, Dingsøyr & Dybå, 2009).

4.6.3 Redundancy

The concept of redundancy is equivalent to the characteristics of Backup Behavior described by Salas et al. (Salas et al., 2005). Cross-functionality allows members to substitute each other in case of demand creating involvement and innovation of team members due to broader expertise. It is reported as crucial for self-managing teams (Morgan, 2007) and appears as "multiskilling" in socio-technical literature (Emery et al., 1976). Lack of Redundancy means specialization of team members, dependency of task accomplishment on availability of certain team members leading to bottlenecks when these are unavailable. It also leads to a general lack of diversified views enhancing the product due to concentration of knowledge.

To improve redundancy literature generally recommends (1) to collocate the team in the same room (Allen & Henn, 2006). Moe et al. (Moe, Dingsøyr & Dybå, 2009) recommend (2) to appreciate generalists inside the team and company culture and (3) to select them during team building and recruitment. (4) Job rotation can further contribute to improve knowledge redundancy by integrating knowledge from different domains (Fægri et al., 2010).

4.6.4 Learning

Learning describes a team's ability in identifying weak points and improving the development process. It is one of the ideas of Scrum originating from the new product development literature (Takeuchi & Nonaka, 1986) known as *multi-learning*. (1) Multilevel and multifunctional learning allow team members to acquire broad knowledge outside their direct product scope, allowing the team to respond quickly and to solve problems fast (Takeuchi & Nonaka, 1986). (2) Job rotation can help to integrate knowledge from different domains and appreciation of organizational concerns (Fægri et al., 2010), but must be legitimized by the organization. (3) Efforts to collect data and to improve should be motivated by the desire to learn and improve inside the team and should not be used to push organizational control.

4.6.5 Autonomy

Team autonomy is necessary. So, the Scrum team perceives its total responsibility over the product without external influence on the team's work plan inside a sprint. It is described as the influence of management and other individuals outside the team. Lack of team autonomy is believed to lead to excessive over-time, high defect rates and personnel burnout. In Scrum, it damages the concept of self-organization (Boehm & Turner, 2003), thus disturbing the team cohesion. Autonomous and self-organizing teams are recognized as a premise, but also as one of the biggest challenges of agile methods (Dybå & Dingsøyr, 2008b).

Autonomy of a team is affected by individual as well as organizational level, (1) self-management thus must be fostered on both levels (Moe, Dingsøyr & Dybå, 2009). (2) Assigning people on more than one project at a time leads to competing for team members and

unequal distribution of resources, thus should be avoided (Moe, Dingsøyr & Dybå, 2009). (3) Collocation of the the team in the same room further helps (Moe, Dingsøyr & Dybå, 2009) .

4.7 Chapter Conclusions

As a partial answer to RQ1: "*How can governing bodies in organizations of knowledge worker teams understand and steer multiple knowledge worker project teams in practice?*", in this Chapter we discussed five dimensions agile teamwork and empirically validate a tool to improve self-reflection of agile software development teams.

As a partial answer to RQ2: "*What are the components of governance necessary to understand knowledge worker project organizations?*", in this Chapter, we measured the cornerstones of agile teamwork in the following five dimensions: (1) shared leadership, (2) team orientation, (3) redundancy, (4) leadership, (5) autonomy. We found that the organizational and individual levels of autonomy and redundancy are the dimensions with the lowest scores as given by the users of this tool. This finding is consistent with the original findings of (Moe, Dingsøyr & Røyrvik, 2009; Moe, Dingsøyr & Dybå, 2009). We introduced a measure for agreement regarding the dimension measurements and found that it was high in our empirical study. This indicates both that (1) team members have similar notions of each dimension and (2) how it applies to their particular team situation. In addition, the teams found the survey tool in general, a useful method to reflect.

Further improvements of the survey tool included a web-enabled version to improve the usability and accessibility. This would enabled easy updates of recommendations to the latest research finding. Data collection via customized online tool would furthermore allow collection of additional teams contributing to the framework's improvement. To improve the quality of data collection, psychometric scale questions (E. R. Thompson & Phua, 2005) could be incorporated into future versions of the survey tool to be able to measure the degree of accuracy or truthfulness the participant tends to give to the answers.

5

Maintenance of Knowledge in Agile Teams

This Chapter also investigates RQ2: *What are the components of governance necessary to understand knowledge worker project organizations?* The attempt to provide a potential answer will be supplemented by the answers from Chapters 8, 10, 11, and 12.

When compared to traditional development methods, agile development practices are associated with more direct communication and less documentation. However, few empirical studies exist that investigate the role of documentation in agile development teams. We thus employed a questionnaire to measure the perceptions of a group of agile practitioners with regard to the documentation in their projects. We obtained responses from 79 agile software development professionals and 8 teams in 13 different countries. Our findings include that (1) over half of developers in our data set find documentation important or even very important but (2) that too little documentation is available in their projects.

This chapter is based on the following publication[1]:

Stettina, C. J., & Heijstek, W. (2011). **Necessary and neglected? An empirical study of internal documentation in agile software development teams.** In Proceedings of the 29th ACM international conference on Design of communication (pp. 159-166). ACM. [1]

[1]The author would like to thank ACM and his co-author for permission to reuse relevant parts of the article in this thesis.

5.1 Internal Documentation

Scrum (Schwaber & Beedle, 2001) is an often applied software development methodology (Abrahamsson, Warsta, Siponen & Ronkainen, 2003; Dybå & Dingsøyr, 2008a). In the spirit of lean manufacturing, Scrum emphasizes a focus on working software and direct communication rather than written documentation (Abrahamsson et al., 2003; Dybå & Dingsøyr, 2008a; Clear, 2003; Rubin & Rubin, 2011). Based upon the mental attitude of value-oriented utilization of resources, lean manufacturing considers the expenditure of resources for any goal other than the creation of value for the end customer as wasteful. In agile development, these attitudes can be found in the agile manifesto (Highsmith & Fowler, 2001) which considers heavyweight processes and comprehensive internal documentation of no direct use to the end customer.

Internal documentation in software engineering may include requirement specifications, design specifications and technical documentation of program code. While such might be of no direct use to the end-user of consumer products, technical documentation in software projects such as embedded comments within the source code, descriptive and/or explanatory notes of APIs, interfaces and algorithms are thought to be beneficial to project initiators as well as to the engineers who are to maintain or expand the software in the future.

Originating from practice (Schwaber & Beedle, 2001) and reflected in theory (Nerur & Balijepally, 2007), the advantages and limitations of Scrum as an agile development method and agile teamwork in particular (compare Chapter 4) have been widely discussed in literature (e.g. (Abrahamsson et al., 2003; Dybå & Dingsøyr, 2008a)). However, although companies using agile methods have been found to be more customer-centric and flexible than document-driven ones (Sillitti, Ceschi, Russo & Succi, 2005), little has been reported on use of documentation within agile teams. While the strictly defined Scrum method with its subsequent phases aids in reducing uncertainty, there is surprisingly little to no anchorage of documentation within the process. In this contribution we therefore aim to improve the understanding of internal documentation in agile software development teams in general, and how agile practitioners perceive this documentation in particular.

5.2 Related Work

The term Scrum depicts cross-functional team characteristics and overlapping phases similarly to those in rugby. It is a lightweight, agile development method, aiming to strip the process of software development to its bare minimum and putting emphasis on direct communication and self-managing teams (Dybå & Dingsøyr, 2008a; Stettina & Heijstek, 2011a). As an adaptive rather than predictive model (Rubin & Rubin, 2011) it does not depend on heavy documentation written upfront. Relying on constant collaboration among the team members and stakeholders, however, literature points at the dangers of general loss of undocumented

knowledge (e.g. (Abrahamsson et al., 2003; Rubin & Rubin, 2011)) when members leave the team or when the project is delivered.

In distributed software development projects, which are increasingly common, software documentation is thought to play a more central role due to the absence of face-to-face communication. The common "transfer by development stage" (Mockus & Weiss, 2001) approach to global software development (GSD) where design and implementation activities take place at different geographical locations, complicates matters further. In this situation, offshore developers are often not able to directly contact a member of the architecture team due to geographical separation. Synchronous communication is often difficult due to time zone differences. In such a scenario documentation plays a more central role in intra-project knowledge sharing processes. In addition, complete and detailed documentation is essential to software maintenance as this typically involves different engineers from the ones who developed the system.

Studies reporting on the use of software documentation during software development are few and deliver mixed results. In studies researching developers' preferences regarding documentation, Lethbridge et al (Forward & Lethbridge, 2002; Lethbridge, Singer & Forward, 2003) find a preference for simple and powerful documentation and conclude that documentation is an important tool for communication, even if it is not up to date. In their literature review on empirical studies addressing the scientific level of evidence behind agile software development methodologies, Dybå and Dingsøyr (2008a) address "documentation" in just one of the identified contributions (Sillitti et al., 2005). In one of the few contributions to understanding of documentation in agile projects, Clear (2003) points at the behavior of students and observes documentation being seen as something external. Instead of being produced in-line with the system as natural part of the development process, documentation was often hurriedly pieced together at the end of a project.

5.3 Objectives

There is a commonly assumed antipathy of agile practitioners towards internal documentation. This assumption stems not in the last place from the agile manifesto, which states that *"[Software documentation is] important, but we need to understand that customers don't care about documents, UML diagrams or legacy integration."* (Highsmith & Fowler, 2001). With the increasing number of integration projects and changing team setups, however, the transfer of development knowledge becomes increasingly important. In this chapter we are interested in understanding the role of documentation in agile development projects, and how the amount and importance of documentation is perceived from the perspective of an agile team. We therefore pose the two following research questions:

- *RQ2b: How do team members in agile software development projects document their*

work?

- *RQ2c: How do they perceive the amount and importance of their internal documentation?*

5.4 Method

In order to be able to attain a level of external validity, data from a wide range of agile practitioners is needed. In order to uniformly obtain a sizable sample and to promote objective answers we developed an anonymous questionnaire. Surveys are easy to deploy and when self-administered through a computer, minimize the effect of socially desirable responding (Paulhus, 2002) due to the impersonality of the computer. To reduce bias we encouraged the respondents to provide their honest opinions by emphasizing the anonymous treatment of data. While we provided personalized links for each team member to ensure the consistency of input, no personal details were stored or used within the examination. An example of the format of the questionnaire is displayed in Fig. 5.1.

Additionally, Miles and Huberman (Miles & Huberman, 1994) propose linking quantitative and qualitative data to provide a deeper picture for the research. We thus asked each ScrumMaster (i.e. (agile) project leader) of an interested Scrum team a set of open-ended interview questions regarding the project environment at the end of the data collection. These qualitative questions (compare Tab.5.1) aimed to learn about the background of the project and details not covered by the quantitatively collected data.

5.4.1 Questionnaire Design

In line with our objective, we want to understand how agile teams produce their documentation and how satisfied they are with it. To reach this two-fold goal we divided the questionnaire into two parts. Part one was geared towards collecting the perceptions of team members regarding their documentation. Part two was designed to inquire about the software tools applied to manage that documentation. The questionnaire was part of a bigger study on agile teams and was developed with software developers including input from practitioners and fellow researchers working in the field of agile software development. To minimize possible flaws in the questionnaire design and to ensure appropriate scales applied, we reviewed the questions with colleagues from the faculty of psychology at the NTNU Trondheim which was the original site of this study.

A: Perceptions Regarding Documentation

The data collected from the teams was coded according to a five-point Likert scale. The questions have been administered in the following order:

- *How much time do you spend on writing documentation daily?*
- *How do you feel about documentation at work?*
- *How effective do you consider finding internal documentation?*
- *How important do you consider documentation for your project?*

Next to the questions regarding documentation, the following question was included to probe the usage of physical artifacts within the agile environment:

- *How important do you consider physical artifacts like story cards or "the wall" for your project?*

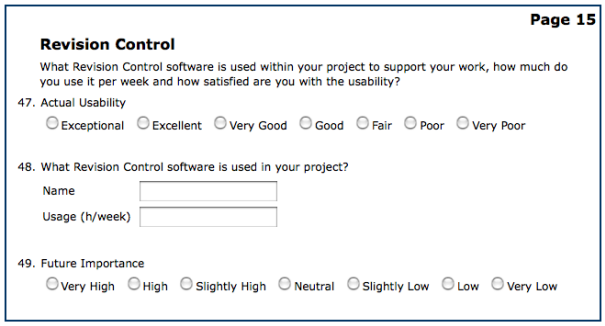
B: Supportive Software

In addition to developers' perceptions of documentation we also needed to inquire about the software tools supporting the development work. To cover a wide range of documentation tools within the survey we agreed upon including 6 categories of tools, namely: (1) *issue tracking*, (2) *revision control*, (3) *electronic discussion*, (4) *Scrum support*, (5) *document management* and (6) *calendar & scheduling*. As we needed to measure the usage of software packages applied, we first asked the participants if a certain tool category is being used within their project. We then asked for the perceived usability from the perspective of the developer. To optimally collect an objective measure of software usability, we applied a method adopted from the European Software Institute's (ESI) 1995 Software Excellence Survey (Dutta & van Wassenhove, 1996) as discussed by Dybå and Moe (Dybå & Moe, 1999). We assessed the usability by placing two opposing subjective rating scales accompanying each question: the *actual usability* and the *believed future importance* as depicted in Fig. 5.1. The gap between believed importance and usability can be regarded as a measure for disaffection with a given solution. This gap is visualized in Fig. 5.7b.

The perception of each team member was gathered for the respective categories as depicted in Fig. 5.1. To prevent inconsistency among the rating items we used two Likert scales consisting of 7 items: *Very High* = 7, *High* = 6, *Slightly High* = 5, *Neutral* = 4, *Slightly Low* = 3, *Low* = 2 and *Very Low* = 1.

5.4.2 Team Agreement

Variance (σ^2) is a measure of how far each value in a set of responses is from the mean. We calculated the variance of the collected values on a team basis to obtain insight in inter and intra-team agreement. A lower variance corresponds with a greater level of agreement within a team. The maximum variance in a team is the variance of the maximum and minimum values


Page 15

Revision Control

What Revision Control software is used within your project to support your work, how much do you use it per week and how satisfied are you with the usability?

47. Actual Usability

Exceptional Excellent Very Good Good Fair Poor Very Poor

48. What Revision Control software is used in your project?

Name

Usage (h/week)

49. Future Importance

Very High High Slightly High Neutral Slightly Low Low Very Low

Figure 5.1: Online Questionnaire: Technology

Table 5.1: ScrumMaster Interview Questions

• Please describe your project shortly. What is the product? What is its size (e.g. man months, lines of code)?
• Please describe your team shortly.
• What is the team's spatial distribution? Is it physically separated?
• What changes did you had to adapt to while switching to Scrum?
• Which aspects of communication do you consider the most salient for the project?
• Does the communication differ to the projects you had before Scrum? How?
• How do you report to your manager? Do you have separate reports to Project Owner and to "Chickens"?
• How do you document the work you do? What is the structure of the project documentation?
• How do you prioritize documentation tasks within the project?
• Are you happy with the current process to handle communication in general (Documentation, Requirements, Meetings etc.)?
• Are you using additional tools to those shared by the rest of the team to improve your productivity?

that can be given in response to an answer. The minimum variance is 0, denoting complete agreement. On a Likert scale ranging from -2 – 2, the maximum variance is

$$\begin{aligned} \sigma^2 &= \frac{\sum(X-\mu)^2}{N} = \frac{\sum X^2}{N} - \mu^2 \\ \text{MAX}(\sigma^2\{1,5\}) &= \frac{1^2+5^2}{2} - \left(\frac{1+5}{2}\right)^2 = 4. \end{aligned}$$

5.4.3 Data Collection

To collect data, the questionnaire has been presented as a set of online survey questions to a group of international project teams, practitioners and experts applying the Scrum methodology. The participants had to be actively involved in a Scrum development team. Only completely filled questionnaire were considered for analysis. To look for international Scrum teams interested in the study Scrum and agile oriented groups within business related networks were targeted. After identifying related individuals from different online community platforms, user

groups as well as agile practitioners originating from direct and indirect contacts, invitations to participate were sent to 150 potential participants. Those who expressed their interest in participation, received an anonymized link allowing identification of teams within the online survey system. In addition, each ScrumMaster then received the set of open-ended questions regarding the project environment (Tab. 5.1).

5.5 Results

After data collection all obtained answers were collected into a *global sample*. In addition to this sample, where at least four agile practitioners of a single team provided a valid survey response, data was also divided into *team samples*. These team samples provide a second perspective on the collected data set by enabling analysis on a team level. The total number of valid data responses comprises 79 software engineers from eight teams located in 13 different countries. Most of the engineers are software developers (47%) and ScrumMasters (18%). Other groups, however, emerged within the data collection phase. These groups are: *Product Owner* (8%), *Quality Assurance* (6%), *Agile Coach* (6%), *Consultant* (9%), *Interaction Designer* (1%) and *CTO* (5%). Their data has been taken into evaluation as long as the individuals were committed to a *"Pig role"* within the Scrum project, thus, being directly involved in the production in a formal and frequent way while referring to internal documentation. The gross amount of relevant working experience among the participants is situated around a work record of 1–5 (38%) and 6–10 (29%) years. Eight teams were analysed in detail. The teams consisted of at least four members and at least two-thirds of all teams have answered the survey. We therefore feel that a consistent representation of a group image was attained for all eight teams. We first present our results from the global sample and then switch to the team perspective.

5.5.1 Global Sample

A: Documentation

In Fig. 5.2, an overview can be found of the time the participants spend on writing documentation in their project daily. We find that the majority of participants spend less than 15 minutes on writing documentation daily. In line with this finding, we observe that almost half of the participating software experts perceive the existing documentation in their project as *too little* (Fig. 5.3).

Fig. 5.4 presents an overview of developers' perceived effectiveness of finding existing documentation in their project. The diagram depicts a balanced distribution. This balanced

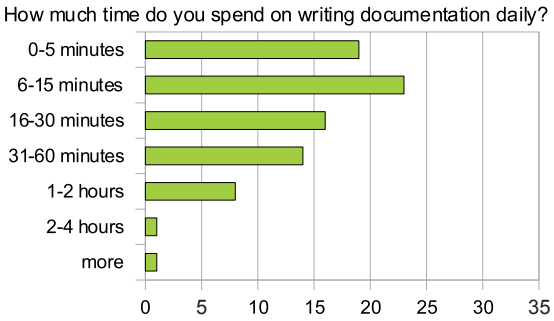


Figure 5.2: Time spent on writing documentation

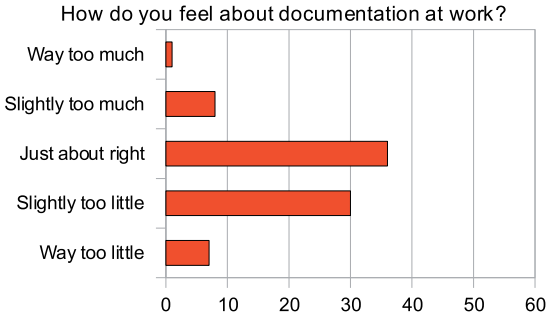


Figure 5.3: Perceived amount of documentation

distribution implies that a substantial group of participants does not find availability in line with agile practices in which documentation must support the development process.

Fig. 5.5 depicts an overview of participants' responses with regards to their perception of the importance of documentation. Here, we observe that majority of the participants define documentation as being *important* or *very important*. This observation is remarkable as we would have expected that agile practitioners would rate documentation as not so important. According to Sharp et al., while relying heavily on direct, face-to-face communication, agile teams use simple physical artifacts to support interaction (Sharp, Robinson & Petre, 2009). This is confirmed by our findings as the majority of participants think physical artifacts are important in their project (Fig. 5.6).

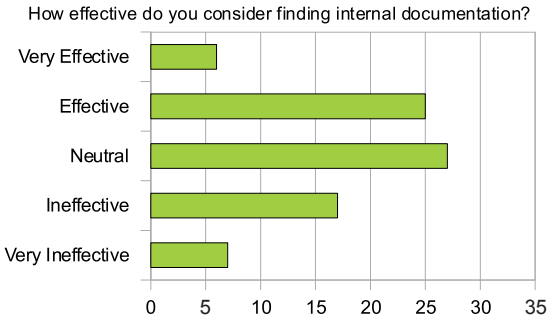


Figure 5.4: Perceived effectiveness in finding documentation

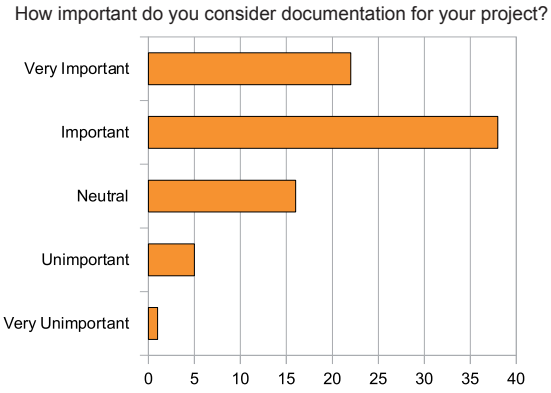


Figure 5.5: Perceived importance of documentation

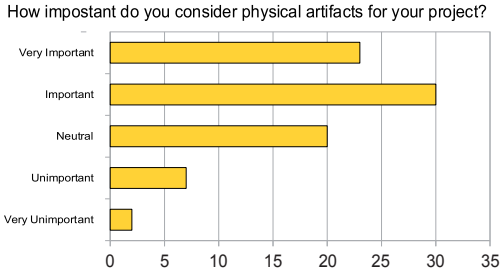
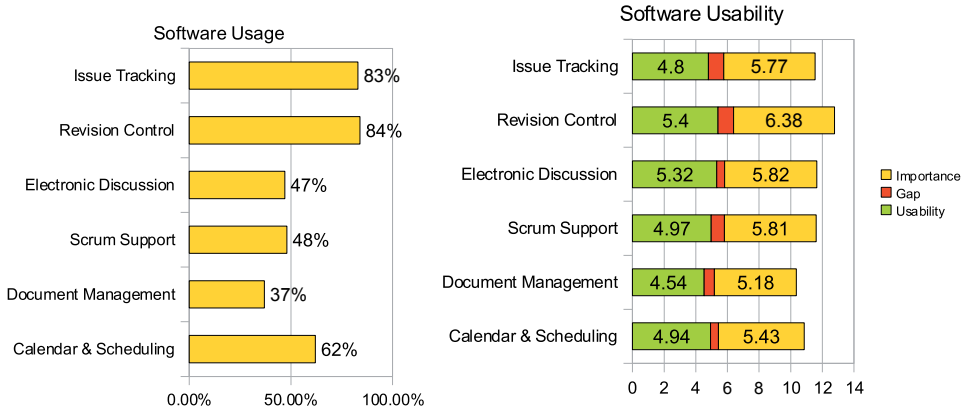


Figure 5.6: Perceived importance of physical artifacts

B: Supportive Software

In Fig. 10.1, we present an overview of the software categories applied by the agile practitioners in their projects. We observe that issue tracking and revision control are used by most participants. The common use of this technology might well be explained by the notion that, next to their employees, source code and knowledge are among the most valuable assets of a software company. Calendar and scheduling tools are used by 62% of the participants. Software supporting electronic discussions and Scrum are similarly distributed with 47% and 48%. Document management is the least applied tool category. Interestingly, document management is reported to be used least by the respondents. In the survey, *document management* was explained to entail the storage of requirements, design papers, used interfaces, created functions and sub-functions.



(a) Usage (b) Usability

Figure 5.7: Software usage, perceived usability and believed importance

While looking at the deviation of usability and importance we observe that revision control, issue tracking and Scrum support have the biggest gap. This means that the participants perceive the highest disaffection for these three categories. The use of Scrum support tools seems surprising. Although one might think that the need for applications supporting a rather direct and physical development method would be rare, there indeed seems to be a demand for supplementary software tools. The perceived usability of solutions for electronic discussion shows the smallest gap among all categories. The most applied software packages for the respective categories were: (1) issue tracking: *JIRA*, (2) revision control: *Subversion*, (3) electronic discussion: *Skype*, (4) Scrum support: *JIRA and Microsoft Team Foundation Server (Scrum plug-in)*, (5) document management: *Wikis and Google Docs* and (6) calendar &

scheduling: *Microsoft Outlook*.

C: Correlation Analysis

Due to non-normality, we resort to non-parametric tests. Using a Mann-Whitney test on the global sample of 79 participants, we find, not surprisingly, that those who identify as software engineers are significantly younger and have significantly less experience than people in managerial roles such a project leader, director or chief technology officer. The latter group notes that they spends significantly more time on conversations at work and writing documentation. This contrasts with the claim by software engineers that they find that too little documentation is available. However, when we compare software engineers to those in senior technical roles such as architects, we find that this latter group also spends significantly more time writing documentation. So in the teams in our data set, top down dissemination of documentation seems prevalent from the perspective of who writes the documentation.

Interestingly, we find no difference between the answers given by the ScrumMasters and those that identify themselves as 'normal' project leaders. When compared to 'traditional' technical leads, ScrumMasters note that they write significantly less documentation but that they spend significantly more time in conversations. Again, this does not seem to correspond with agile developer's requirements regarding the availability of documentation in their projects.

We find that the more experienced agile team members are, the more efficient they find documentation in their project ($\tau = -0.294$ at $p = 0.018$). A possible explanation could be that developers only learn to understand the importance of documentation as they progress in their careers.

We find that those that spend much time creating documentation are significantly more positive about the amount of documentation available in their jobs ($\tau = 0.555$ at $p = 0$). This could simply mean that these people justify their labor or that this group is simply more aware of the documentation that is available. It might, however, also indicate that this group is given the time to write documentation and are therefore happier. We are unsure whether teams in which these engineers work were more efficient as a result of this, though.

5.5.2 Team Sample

Data sets consisting of sufficient interrelated team members have been aggregated into team samples and are presented in Tab. 5.2. Team T1 (UK) provided a back-end software for a major Massive Multiplayer Online (MMO) game publisher. Team T2 (US) developed a collaborative solution to support building design and construction-related tasks for a major American software company. Team T3 (UK) developed software for digital media and mobile platforms concentrating on software development, interaction and visual design. Team T4

(Norway) was employed by a company providing smart card based public key solutions for security transactions, consisting of developers from 2 separate locations running several parallel projects. T5 (The Netherlands) was in charge of corporate websites and web shops. Team T6 (Sweden) developed and maintained a Swedish news guide and community website. Team T7 (India) worked on an e-commerce solution. Team T8 (New Zealand) consisted of a business analyst, a quality assurance specialist and two developers working for a state insurance agency.

Between the teams, the team members agree most on the fact that too little documentation is available. This consistency between dissimilar teams strengthens our earlier finding that agile practitioners find that too little documentation is available.

A: Documentation

As presented in Tab. 5.2, the perceptions on the amount of documentation, effectiveness in finding internal documentation and importance of artifacts are accumulated from the data. According to the respective Likert scale, the values are distributed on a scale from "−2" to "+2". A "0" thus corresponds to *"just about right"*, a "+1" to *"slightly too much"* and a "+2" to *"way too much"* while a "−1" and a "−2" correspond to *"slightly too little"* and *"way too little"*, respectively. The value of "1" therefore means that the team perceives documentation as *"slightly too much"*, while the value of "−2" would represent a team perception of documentation as *"way too little"*. Consistently with the global sample Tab. 5.2 reveals that more than half of the researched teams reported perceived a deficit in the amount of documentation. Note that none of the teams perceive their documentation as *"too much"*.

B: Supportive Software

We observe that Wiki solutions are prominent within the researched teams: five of the eight groups specifying to use a document management solution name a Wiki system as the tool used. The two Wiki solutions in brackets, mentioned in row "documentation tool" in Tab. 5.2, mark the two cases where the team ScrumMaster mentioned the particular application used by the team in the open questions. Team members of those same teams, however, did not specify the use of these specific tools. This might imply that these Wiki solutions were introduced but not adapted in practice by the team: *"[W]e have a wiki that we are supposed to use,"* comments the ScrumMaster of Team T6.

After the Wiki, the most commonly used artifact seems to be the "user stories". A user story is the agile equivalent of a use case. The distinction between the Wiki and user stories seems to be fluid, as one ScrumMaster (Team 1) notes, *"Work is documented on the company wiki where required. If it is a small addition then it will be part of a task, but if it is a larger*

addition then it may have it's own user story.". Software tracker JIRA[2] was mentioned several times, *"We use JIRA with the Greenhopper plug-in. This provides virtual equivalents to for a wall, User Story cards, burndown,"* says a ScrumMaster (Team 2). Also Team seven was positive about JIRA, *"Jira is our "source of truth". Confluence [A Wiki] is the repository for collaborative documentation. We also required sensible comments in commits. We ran Clover for test coverage, Crucible for collaborative code reviews, CruiseControl for Continuous Integration. Most of the project documentation was divided between the product backlog (in Jira) and shared documents in Confluence,"* noted its ScrumMaster.

Team three makes use of a combination of Google Docs and a whiteboard, *"We have a sprint backlog for each team as a Google docs, editable by the whole team. Our release plan is on a whiteboard, and we have per-client product backlogs for individual products - again, in Google Spreadsheets. [. . .] Beyond that the main documentation we produce is user experience design docs. I think we're a bit heavyweight here right now but we're working for some clients who demand this stuff for their internal sign-off procedures,"* according to the ScrumMaster.

Team five works with a combination of an advanced change and configuration management system and post-it notes: *"We have a Story Board with progress using post-its next to the team. I add the burndown-chart there every morning. The team keeps the StoryBoard up to date. For estimating we use Planning Poker (with cards). [...] As the ScrumMaster I don't have a lot of documentation to do. The Product Owner will provide the documentation for the team, creates User Stories for them in Microsoft Team Foundation Server (TFS) and decides the prioritization,"* says the ScrumMaster.

5.6 Discussion

In this section we discuss our findings in line with our research questions. There seems a natural antipathy of software engineers towards activities not directly linked to the creative process of writing code, many engineers indeed prefer writing code instead of drafting their ideas in text (Clear, 2003). Agile software development methods concentrate on direct communication emphasizing that comprehensive internal documentation is of no direct use to the end customer (Highsmith & Fowler, 2001). In line with existing concerns (Abrahamsson et al., 2003; Rubin & Rubin, 2011), however, the findings of this study point out that direct communication alone seems insufficient.

5.6.1 Documentation amount and importance

Agile practitioners in our multinational sample perceive documentation as important for their projects and find there is too little internal documentation is available in their projects. Doc-

[2] http://www.atlassian.com/software/jira/

umentation seems neglected by original Scrum literature. It is mentioned but not anchored in the original process (Schwaber & Beedle, 2001) which relies heavily on verbal communication. But verbal communication is susceptible to lapses of memory and after some time it get progressively harder to recall design rationale. This problem is compounded by team turnaround. As a member of T6 comments, *"Code comments are used in an effective manner. During project development any needed documentation is generally available. However, finding documentation for older projects is not always easy, and sometimes this documentation is missing."*

The correlation analysis confirms a predominant top-down dissemination in agile settings where practitioners in senior technical roles spend significantly more time on writing documentation. As mentioned in the interviews, product owners usually write requirements and user stories. From there on it is up to the team, it takes care of the implementation and controls what is documented.

Some agile scholars propose documentation to be *"light but sufficient"* (Cockburn & Highsmith, 2001). Sufficient, however, it doesn't seem to be, as none of the researched teams noted that they perceive documentation as such. Writing less than 15 minutes (Fig. 5.2) of documentation daily seems not enough for the produced software and is surely not enough to sustain a co-developed artifact, produced in-line with emerging code. Surprisingly, the vast majority of respondents perceives documentation as important or very important.

We can now answer the two subquestions. Regarding RQ2a: *"How do team members in agile software development projects document their work?"*) we may conclude that agile teams apply a multitude of online and offline artifacts to document their work. Regarding RQ2a: *"How do they perceive the amount and importance of their internal documentation?"*) we may conclude that team members perceive the amount of internal documentation as too little. Further, they perceive documentation as missing for older projects.

5.6.2 Software, more than a backchannel

The teams in our study predominantly adopt collaboration tools to document and share agile artifacts such as user stories or sprint backlogs. An interesting finding is the perceived importance and application of software that directly aim to support Scrum. This is surprising to that extend that one could expect the sufficiency of direct communication and physical artifacts. Convenient handling of agile artifacts and distributed settings seems to be one reasons here. *"We have good experience using physical artifacts for local projects, but most of our projects are multi location and require an electronic solution."*, says the ScrumMaster of Team T4.

The perceived usability of solutions for electronic discussion showed the smallest gap among all categories, meaning that the participants find the current solutions very usable. According to comments team members, the solutions surpass the expectations of a pure *"backchannel"* (Team T3). The growing instant messaging culture seems to make a contribution

Table 5.2: Descriptive variables, team results (x) and agreement (σ^2)

		T1	**T2**	**T3**	**T4**	**T5**	**T6**	**T7**	**T8**	**AVG. AGR.**
country		UK	US	UK	NO	NL	SE	IN	NZ	
team size (pers.)		4	9	5	12	6	4	8	6	
collected answers		4	6	5	6	5	3	8	4	
avg. exp. (yrs.)		7.75	13.7	6.6	12.7	2.6	10	7	3.5	
spacial distribution		co-loc.	co-loc.	co-loc.	distrib.	co-loc.	co-loc.	distrib.	co-loc.	
documentation tool		(Wiki)	Confl. Wiki	Google Docs	-	-	(Wiki)	(Confl. Wiki)	-	
perceived doc. amount	x	**-0.25**	**-0.50**	**-0.40**	**-1.30**	**-1.00**	**-0.75**	**-0.13**	**0**	
	σ^2	*(0.19)*	*(0.25)*	*(1.44)*	*(0.89)*	*(0.40)*	*(0.67)*	*(0.61)*	*(0)*	*(.56)*
perceived eff.. finding doc	x	**0.65**	**0.76**	**0.44**	**0.60**	**0.52**	**0.50**	**0.75**	**0.45**	
	σ^2	*(0.69)*	*(0.47)*	*(0.16)*	*(1.33)*	*(1.44)*	*(0.89)*	*(0.69)*	*(0.69)*	*(.80)*
perceived importance artif.	x	**1.00**	**0.70**	**0.76**	**0.57**	**0.72**	**0.75**	**0.7**	**0.85**	
	σ^2	*(0)*	*(2.25)*	*(0.16)*	*(0.47)*	*(1.04)*	*(0.67)*	*(0.50)*	*(0.69)*	*(.72)*
AVERAGE AGREEMENT	σ^2	*(.29)*	*(.99)*	*(.59)*	*(.90)*	*(.96)*	*(.74)*	*(.60)*	*(.46)*	*(.69)*

here and Skype has been the most named tool in this category. A quote from the ScrumMaster of Team T7: *"Communication with the team and our client worked very well when we decided to move away from "voice" conversations (accents, network latencies, time wasted setting up conference phones) to text chats. Even though they can take substantially longer, logs are permanent and we found it easier to share documentation, make decisions and stick to them."*.

5.6.3 Validity Considerations

Due to the low amount of data sets containing 79 individuals conclusions were drawn carefully. As we base our evidence on small team data sets, we have improved the transparency of data by adding the variance of given answers among the team members. Throughout the whole process of data collection, we encouraged participants to give realistic answers and emphasized the anonymous treatment of data to establish a reasonable level of trust and to reduce bias. No results other than the processed outcome for the whole team would be distributed or given to their superiors. While we provided personalized links for each team member to ensure the consistency of input, no personal details were stored or used within the examination. We address the bias of participants towards positively perceived answers (socially desirable responding (SDR) (Paulhus, 2002)) by anonymous self-administration of questions through a

computer, meaning that when the subjects' personal details are not required the person does not feel directly and personally involved in the answers he or she is going to give. This effect is further neutralized through the impersonality of the machine. To validate the perceptions collected via quantitative questionnaire we collected open-ended interview responses from ScrumMasters as proposed by Miles and Huberman(Miles & Huberman, 1994). We must be careful to make conclusions regarding the reported time spent documenting as we are unsure what the quality of the resulting documentation was. Furter conclusions can only be drawn when we review the documentation itself.

5.7 Future Work

The results of this study point to a number of directions to pursue future research. First, our study in a reality-check manner collects how agile practitioners perceive internal documentation in their projects, further in-depth studies thus are necessary to address the quality of created documentation along with the process and team routines. Second, it has been argued that the lack of architectural focus leads to suboptimal design-decisions (McBreen, 2002) in agile methods. UML has been found useful during the design phase but is considered as heavyweight by many practitioners. Further research on incorporation of lightweight modeling methods such as the ICONIX process (Rosenberg & Scott, 1999) or the Active Documentation Software Design (ADSD) (Rubin & Rubin, 2011) is recommended. Physical artifacts are widely accepted in the agile community, perhaps there are also possibilities for a more physical extension of UML. Third, previous studies (Aguiar & David, 2005) found that Wiki systems have been successfully applied to improve internal documentation efficiency, and have been found widely applied in this study. Further research on the adoption of lightweight documentation systems such as Google Docs or a Wiki is necessary.

5.8 Chapter Conclusions

As a partial answer to RQ2: "*What are the components of governance necessary to understand knowledge worker project organizations?*", in this Chapter we discuss knowledge maintenance and transfer as an integral aspect of agile teams.

In this chapter we present the results of an empirical study on documentation in agile Scrum software development teams. Executed in a multidimensional manner we analyze the collected data sets consisting of quantitative team and global samples as well as qualitative interview questions. Our findings stem from a representative data set of agile practitioners and international teams and warrant further study.

One of our main findings is that documentation alone is insufficient. While agile methods recommend to make "lean" documentation, suggesting that documentation should only include

information that is used, we found that agile software development practitioners perceive their internal documentation as important but that they feel that too little documentation is available. Analogously to the observations of Clear (Clear, 2003), we found that documentation is rather seen as a burden than a co-created (core) artifact and found support to the perceptions in literature that without ensuring a proper documentation process agile methods can cause major knowledge loss during or after system development (Abrahamsson et al., 2003; Rubin & Rubin, 2011).

We found that agile teams adopt collaboration tools to share and work on agile artifacts and that Scrum dedicated software is perceived as important and helpful to support the method. We found that instant messaging is perceived as a helpful "backchannel" and supports documenting decisions. We may conclude that integration of software tools into the process is crucial. Lightweight solutions such as Wikis or Google Docs are prominent, their adoption however needs further research (compare Tab. 5.2).

When interviewing practitioners the authors often found a passion for agile methods. Discussing their tools and routines it was the first time developers would address a software development process as something passionate. This, however, seems not yet true for agile documentation, and future research needs to address an appropriate incorporation within the process to make knowledge transfer truly agile *and* sufficient.

6

Team Routines and the Adaptation of Unforeseen Variations

This Chapter investigates RQ3 ("*How can governance address the dynamics of knowledge work across multiple knowledge worker teams?*"). The attempt to provide a potential answer will be supplemented by the answers from Chapters 7-12. According to Mintzberg (1979, 1993), professional organizations can be governed by controlling the work processes and routines applied. The work of highly agile medical teams, just like project management and project work in general, is highly dependent on experience and processual knowledge. Processual knowledge, tacit knowledge, opposite to explicit knowledge (information) can only be acquired on the process. In this Chapter I present a short case study of medical teams, (1) I present their necessity for local autonomy to ensure right decisions are made in context, (2) I provide a view on how large organizations especially in medical context document their routines and processes, and (3) I contribute to the understanding of enterprise process models.

This chapter is based on the following publication[1]:

Stettina, C. J., Groenewegen, L. P., & Katzy, B. R. (2012). **Towards flexibility and dynamic coordination in computer-interpretable enterprise process models**. In Enterprise Interoperability V (pp. 105-115). Springer London. [1]

[1]The author would like to thank Springer and his co-authors for permission to reuse relevant parts of the article in this thesis.

6.1 Autonomy, Adaptation, and Documentation of Routines

For decades knowledge organizations seek for efficient ways to capture and transfer knowledge and experience stored in their collective memory and organizational routines (Pentland & Feldman, 2008). To achieve this goal numerous notations have been implemented in workflow, management information and decision support systems. In industry, for example, the dominating flowchart and workflow driven notations such as Event-driven Process Chains (EPCs), UML Activity Diagrams and the Business Process Modeling Notation (BPMN) are used to analyze and improve ways of working (Reijers & Mendling, 2011). In medicine the standardization of care processes as promoted by implementation of clinical pathways and their executable computerized counterparts, computer-interpretable guidelines have been widely discussed in literature (Sonnenberg & Hagerty, 2006; Isern & Moreno, 2008). Most prominent examples of such representation languages developed for medical purpose are: Asbru (Shahar, Miksch & Johnson, 1998), GLARE (Giordano, Terenziani, Bottrighi, Montani & Donzella, 2006), GLIF3 (Boxwala et al., 2004), and SAGE (Tu, Campbell & Musen, 2004).

Studies have shown that process models and guidelines implemented in decision-support systems (DSS) (Peleg & Tu, 2009) have a better impact on the behavior of professionals than the traditionally narrative guidelines (S. Shea, DuMouchel & Bahamonde, 1996). However, one major and still unresolved concern is how to address the large number of deviations in live routines (Pentland & Feldman, 2008). While dynamic consistency is still problematic in most notations like UML 2.0 (Küster & Engels, 2004), modelers using the current notations, described as the "task-based paradigm" (Fox & Das, 2000) or "Task-Network Models (TNMs)" (Peleg et al., 2003) have to predict and incorporate all possible execution paths. Literature argues that current decision-support systems do not work well when the encoded medical knowledge is incomplete and that it is unrealistic to predict all possible exceptions and errors (Grando, Peleg & Glasspool, 2010). The challenge is to strike a balance between flexibility and the need for structure and control in process models.

In this chapter we contribute to the understanding of collaboration with adaptation to unforeseen variations in enterprise process models translating qualitative process data into a computer-interpretable guideline model (CIG) (Grando et al., 2010). To overcome the current implementation barriers the coordination modeling language Paradigm (Andova, Groenewegen & de Vink, 2011), as a possible approach, addresses coordination of collaborating components in terms of dynamic constraints, which can be easily translated into executable models sharable among different enterprises. The Paradigm component McPal (Andova, Groenewegen, Stafleu & de Vink, 2009) allows the addition of new behavior, and, subsequently, gradually adapts the system dynamics without quiescence. For enterprises it is important to point out the roles of different actors in the respective departments and organizations involved. Taking the example of a medical procedure we want to focus on the interoperability through modeling of dynamic coordination aiming at adaptation to variations on-the-fly.

6.2 Paradigm

The name Paradigm is an abbreviation of *PARallelism, its Analysis, Design and Implementation by a General Method* (Andova et al., 2011). The language has a strongly visual representation, analogous to other models such as those of UML. However, Paradigm is underpinned by precise mathematical constructs, constituting the formal definitions of its notions and their dependencies. On the basis thereof dynamic consistency between participants in collaboration can be understood and analyzed. As such, Paradigm consists of five basic notions to address coordination of collaborating components: state transition diagrams, phases, (connecting) traps, roles and consistency rules. In this section, we shall briefly explain Paradigm notations though a realistic medical example. We will now first introduce the medical example and then proceed explaining the relevant Paradigm notations.

In the course of the EDAFMIS project we have used a recorded bronchoscopy intervention from the pulmonology department of a major Dutch university hospital and structured the process steps as narrative fragments according to Pentland and Feldman (2007). Flexible diagnostic bronchoscopy is a non-invasive medical procedure to examine the inside of human airways. As opposed to open surgery, non-invasive interventions allow a practitioner to examine a patient's airways via a bronchoscope without damaging biological tissue. Recent developments in image-guided intervention techniques allow a deeper examination of even smaller individual bronchi, better spacial orientation, and enable integration of process support into the intervention. Table 6.1 presents the collected process steps structured as narrative fragments (see Pentland and Feldman (2007)) and Figure 6.1a gives a simple activity diagram thereof in UML-style. At the figures left side it is indicated which actor is performing a particular step: the Doctor, Nurse or both.

To turn the example into a Paradigm model, the first thing to do is: grouping the steps per actor into a so-called STD (state transition diagram) where states (rounded rectangles) correspond to the original steps, thereby keeping the sequential order per actor, but possibly adding some states in view of cooperation. Figure 6.1b gives the two STDs: for this explanation, we have put the two STDs inside a similar but less simple activity diagram as before, here with Doctor in the left swimlane and with Nurse in the right swimlane of the activity diagram. Note, we keep the specific actor step at the same level as in the figures part (a).

Thus step Brief, in part (a) performed by both, reappears as state Briefing in STD Doctor as well as as state InBriefing in STD Nurse. In view of synchronizing the sojourns in Briefing by Doctor and in InBriefing by Nurse, the two actors do have to take transition start together: each from an extra state ToStart; and, as they do not have to arrive simultaneously in ToStart, each can arrive there independently (but usually in time) from another extra state Elsewhere: their starting state.

A. (Doctor) Brief the nurse about the procedure
[GOAL: Establish common understanding of the team]
B. (Nurse) Gather necessary equipment
C. (Nurse) Position the patient, the bronchoscopist and equipment
D. (Nurse) Check if patient is under anesthesia
E. (Doctor) Insert Bronchoscope nasally or orally. Nasally 3 positions: 1) Supine position. 2) Patient standing, 3) Patient sitting
F. (Doctor) Insert the bronchoscope gently
G. (Both) Check the patients face repeatedly until retraction
[GOAL: Ensure patient state as he cannot speak]
H. (Doctor) At the main carina: rotate the bronchoscope to 90 degrees to the right
I. (Nurse) Apply topical anesthesia to the right main bronchus, repeat for the left main bronchus
J. (Doctor) Inspect right bronchial system
K. (Doctor) Inspect left bronchial system
L. (Doctor) Slowly retract bronchoscope
M. (Doctor) Carefully inspect the proximal part of the trachea
N. (Doctor) After fully retraction hand over the bronchoscope to the nurse
O. (Nurse) Clean and store the bronchoscope
P. (Nurse) Monitor patient
Q. (Doctor) Write bronchoscopy report

Table 6.1: Narrative Fragments: Flexible Bronchoscopy

Assuming Doctor takes the lead, he explicitly announces the end of the briefing, by entering BriefReady, thereby actually releasing Nurse from being briefed. This means, Nurse is launched into subsequently proceeding via states Gathering (equipment), PositioningPatient, CheckAnest (checking anesthesia) to Checking (the patient's condition while Doctor takes over). Arriving in Checking then means, Nurse launches Doctor into proceeding to RotateReady. Arriving in RotateReady by Doctor launches Nurse into proceeding to Checking1. And so on until, when both are in Ready, they leave together by going to Out. Figure 6.1b contains annotations about the launchings.

The above subsets of states into which Doctor as well as Nurse can launch each other to be in, are precisely the constraints on the various STDs, our Paradigm model has to impose for a certain while only, the so-called phases. Figure 6.2a visualizes them as STD fragments. For Doctor we have the phases: Before, Prologue, ToLungs, Lung1, Lung2AndBack, Epilogue and After. Similarly, for Nurse we have the phases: Before, Intro, GettingReady,

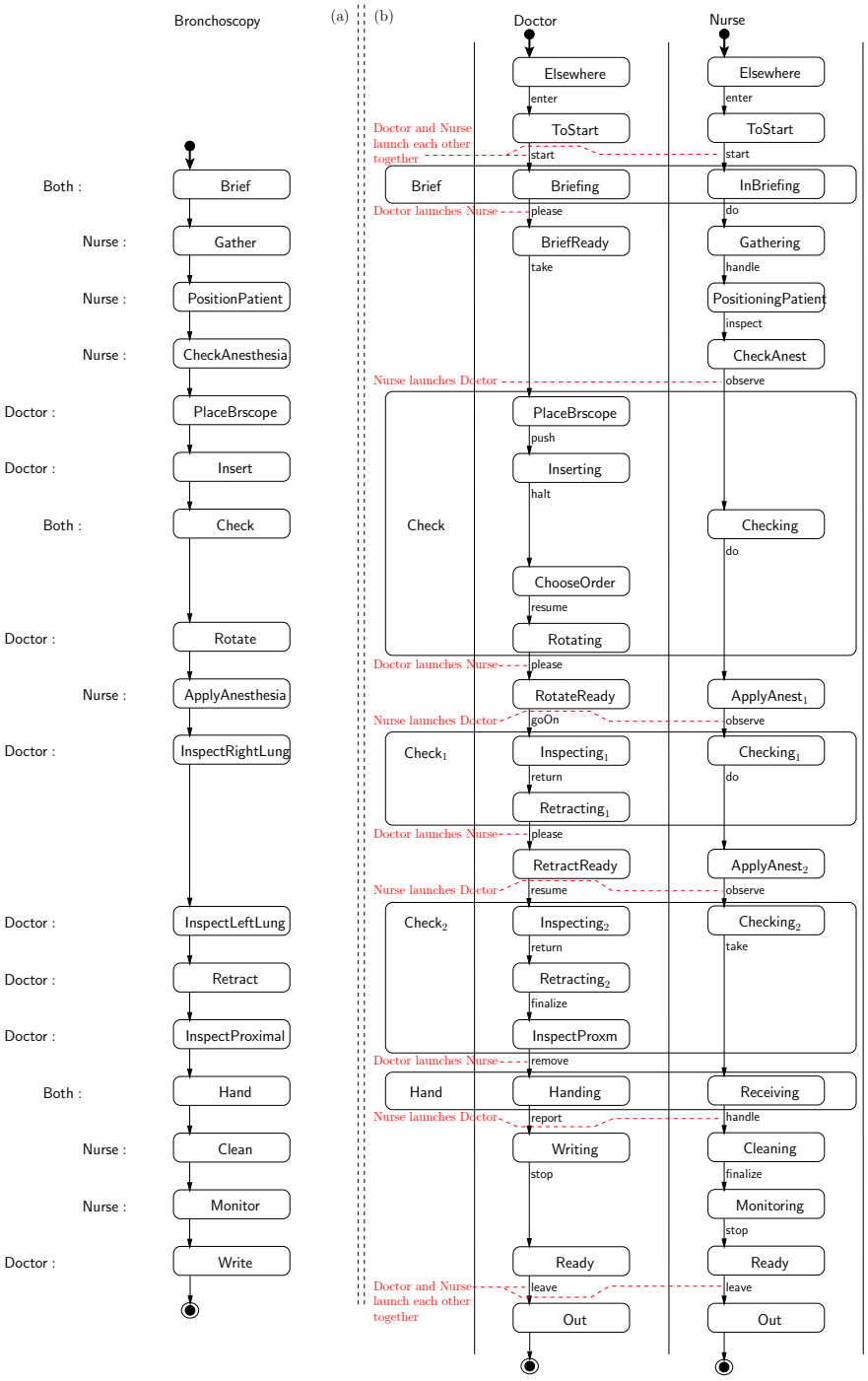


Figure 6.1: (a) Overview STD Bronchoscopy; (b) STDs Doctor and Nurse, in 2 swimlanes.

Figure 6.2: (a) Phases, traps and (b) (synchronized) roles of Doctor and Nurse.

Lung1, Lung2AndBack, Finishing and After. Based on phases of an STD and traps of these phases, Paradigm provides a so-called role of the (underlying) STD; in this example role Doctor(BronchoTeam) and role Nurse(BronchoTeam), see Figure 6.2b. A role of an underlying STD is another STD, where states are phases of the underlying STD. A role specifies in which order(s) a constraint imposed can change, a so-called phase transfer. Such a phase transfer should be allowed only after some form of sufficient progress within the phase currently imposed has occurred. Paradigm specifies such a progress condition through a so-called trap of a phase: a further constraint within a phase, not imposed, but committed to by just entering the trap: once entered a trap cannot be left as long as that same phase remains imposed. Traps are visualized as rectangles around the subset of states in a phase constituting the trap. So, once a trap of a phase has been entered, the entering may serve as condition for a transfer from that phase to a next phase. Therefore, traps label transitions in a role STD, cf. Figure 6.2b); if such phase transfer is to a different next phase, the trap moreover has to be connecting, i.e. each state within the trap also belongs to the next phase, but within that next phase the former connecting trap can be left (is not necessarily a trap any longer). Contrarily, non-connecting traps can serve as necessary condition for a phase transfer in a different role.

Through synchronization, singular transitions from different roles are being coupled into one protocol step. It is through a consistency rule, Paradigm specifies a protocol step. In the example as presented here, we restrict the (Paradigm) consistency rules to those having the following format: (*i*) each rule starts with an $*$; (*ii*) the right-hand side of the $*$ lists one or more role steps being synchronized, separated by a comma; (*iii*) all role steps in such a list come from different roles. (Technically this means, in our example all protocols are so-called choreographies, see (Andova et al., 2011); so, here neither orchestrations occur nor any conductor driving one or more protocol steps; also, here no variable updates will be related to protocol steps in the form of a so-called change clause, see (Groenewegen, Kampenhout & Vink, 2005).)

Below is the complete set of consistency rules for the above example, specifying how Doctor and Nurse can coordinate their activities while making progress in the order as required. As we shall see, the order is precisely the one suggested in Figure 6.2 via the two different swimlanes. For this example choreography there are 10 consistency rules.

$$* \ \mathsf{Doctor}(\mathsf{BronchoTeam}) : \mathsf{Before} \overset{present}{\rightarrow} \mathsf{Prologue}, \quad \mathsf{Nurse}(\mathsf{BronchoTeam}) : \mathsf{Before} \overset{present}{\rightarrow} \mathsf{Intro} \tag{6.1}$$

$$* \ \mathsf{Doctor}(\mathsf{BronchoTeam}) : \mathsf{Prologue} \overset{informed}{\rightarrow} \mathsf{Prologue}, \quad \mathsf{Nurse}(\mathsf{BronchoTeam}) : \mathsf{Intro} \overset{listening}{\rightarrow} \mathsf{GettingReady} \tag{6.2}$$

$$* \ \mathsf{Doctor}(\mathsf{BronchoTeam}) : \mathsf{Prologue} \overset{informed}{\rightarrow} \mathsf{ToLungs}, \quad \mathsf{Nurse}(\mathsf{BronchoTeam}) : \mathsf{GettingReady} \overset{prepared}{\rightarrow} \mathsf{GettingReady} \tag{6.3}$$

$$* \ \mathsf{Doctor}(\mathsf{BronchoTeam}) : \mathsf{ToLungs} \overset{atLung_1}{\rightarrow} \mathsf{ToLungs}, \quad \mathsf{Nurse}(\mathsf{BronchoTeam}) : \mathsf{GettingReady} \overset{prepared}{\rightarrow} \mathsf{Lung}_1 \tag{6.4}$$

$$* \ \mathsf{Doctor}(\mathsf{BronchoTeam}) : \mathsf{ToLungs} \overset{atLung_1}{\rightarrow} \mathsf{Lung}_1, \quad \mathsf{Nurse}(\mathsf{BronchoTeam}) : \mathsf{Lung}_1 \overset{doLung_1}{\rightarrow} \mathsf{Lung}_1 \tag{6.5}$$

$$* \ \mathsf{Doctor}(\mathsf{BronchoTeam}) : \mathsf{Lung}_1 \overset{atLung_2}{\rightarrow} \mathsf{Lung}_1, \quad \mathsf{Nurse}(\mathsf{BronchoTeam}) : \mathsf{Lung}_1 \overset{doLung_1}{\rightarrow} \mathsf{Lung}_2\mathsf{AndBack} \tag{6.6}$$

$$* \ \mathsf{Doctor}(\mathsf{BronchoTeam}) : \mathsf{Lung}_1 \overset{atLung_2}{\rightarrow} \mathsf{Lung}_2\mathsf{AndBack}, \quad \mathsf{Nurse}(\mathsf{BronchoTeam}) : \mathsf{Lung}_2\mathsf{AndBack} \overset{doLung_2}{\rightarrow} \mathsf{Lung}_2\mathsf{AndBack} \tag{6.7}$$

$$* \ \mathsf{Doctor}(\mathsf{BronchoTeam}) : \mathsf{Lung}_2\mathsf{AndBack} \overset{retracted}{\rightarrow} \mathsf{Lung}_2\mathsf{AndBack}, \quad \mathsf{Nurse}(\mathsf{BronchoTeam}) : \mathsf{Lung}_2\mathsf{AndBack} \overset{doLung_2}{\rightarrow} \mathsf{Finishing} \tag{6.8}$$

$$* \ \mathsf{Doctor}(\mathsf{BronchoTeam}) : \mathsf{Lung}_2\mathsf{AndBack} \overset{retracted}{\rightarrow} \mathsf{Epilogue}, \quad \mathsf{Nurse}(\mathsf{BronchoTeam}) : \mathsf{Finishing} \overset{thankYou}{\rightarrow} \mathsf{Finishing} \tag{6.9}$$

$$* \ \mathsf{Doctor}(\mathsf{BronchoTeam}) : \mathsf{Epilogue} \overset{toExit}{\rightarrow} \mathsf{After}, \quad \mathsf{Nurse}(\mathsf{BronchoTeam}) : \mathsf{Finishing} \overset{toExit}{\rightarrow} \mathsf{After} \tag{6.10}$$

Please note how the rules really specify the various mutual launchings of Nurse and of Doctor. Rule 1 says: Doctor and Nurse both do a simultaneous phase transfer from Before to Briefing or to InBriefing respectively, but only after they both have been trapped in present. Rule 2 specifies how Doctor launches Nurse into GettingReady, but Doctor does not change its own current phase. Contrarily, rule 9 specifies how Nurse launches Doctor. Rule 10 then specifies how Doctor and Nurse both do a simultaneuous phase transfer to After, coming from Epilogue or Finishing respectively. The additional six rules inbetween 2 and 9 specify the other launchings.

In the remainder of this section we sketch, how Paradigm's special pattern-like component McPal (Andova et al., 2009) can be put on for unforeseen flexibility. McPal has the following characteristic features: (1) McPal owns a formal specification of the complete Paradigm model,

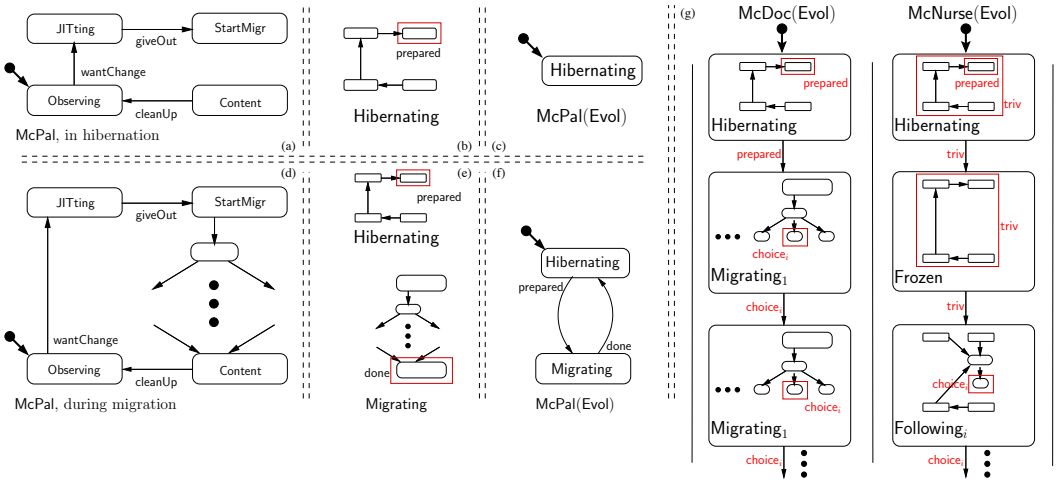


Figure 6.3: McPal: (a, d) STD, (b, e) phases, traps, (c, f) role Evol; (g) cooperating McPals.

McPal included; as long as it is the as-is model, McPal is in hibernation: not influencing the as-is dynamics at all. (2) McPal can change the specification, e.g. by adding new, specific dynamics for a to-be situation as well as for migrations to it, while keeping all dynamics constrained to the original as-is specification; (3) on the basis of the change, McPal awakes from hibernation and coordinates traversing one of these migration trajectories; (4) McPal returns to hibernation after successful migration, possibly shrinking the specification by removing parts no longer relevant, thus keeping the to-be model only.

Figure 6.3a–f visualize the general idea of McPal; a,d give its self-adapting STD: a-STD when in hibernation; d-STD when coordinating an example migration. When McPal's d-STD is current, the state McPal is currently in, belongs to a path from StartMigr to Content: taking step giveOut extends the a-STD to d-STD (for this example) and, taking step cleanUp shrinks the d-STD back to a-STD. Upon arrival in StartMigr phase Hibernating still is the current constraint (b,c), but as trap prepared is being entered, the transfer from Hibernating to Migrating is enabled (e, f) and carried out later. Technically, all this can be specified via well-chosen consistency rules and other Paradigm model fragments. Thus, one might recognize McPal's above mentioned characteristics (1) and (2). Moreover, when phase Migrating is the current constraint and assuming Migrating has been well-defined, the migration is indeed coordinated as it should (3). Eventually, McPal arrives in state Content, thereby entering trap done; hence the phase transfer from Migrating back to Hibernating gets enabled and is carried out later. So step cleanUp will be taken, shrinking McPal's STD back to a-STD (4).

In the existing McPal papers, one McPal not only coordinates this self-adaptation but also a suitable migration of the other model components. In medical cases such as the above

brochoscopy, one would like to have more McPal components: at least one for each person involved having responsibility to change the ongoing medical procedure, if needed. Figure 6.3g visualizes a possible step towards a far more mature solution for coordination needed between different McPal components: here McDoc and McNurse for the above Doctor and Nurse respectively. Similar to Figure 6.2b, an activity diagram is given for first synchronized transfers in McDoc's and McNurse's Evol roles. Concsistency rules are omitted.

In this case McDoc takes the initiative (e.g. for changing the patient's position on-the-fly of the ongoing procedure). So, McDoc(Evol) tranfers from Hibernating to Migrating_1, thereby freezing McNurse(Evol) into Frozen, thus preventing it to take a similar initiative. McDoc then decides about the details of the repositioning by choosing to enter trap choice_i. As a result, McNurse then transfers from Frozen into Following_i whereupon it will enter a similar trap choice_i. From then on, both can guide their respective Doctor and Nurse component into well-coordinated repositioning of the patient, eventually resuming their ongoing procedure.

6.3 Discussion

Organizational routines occupy *"the crucial nexus between structure and action, between the organization as an object and organizing as a process"* (Pentland & Rueter, 1994). Literature argues that organizational routines are not simply followed or reproduced as people have a choice between following a routine or whether to amend it (Feldman & Pentland, 2003; Becker, 2004). Although Figure 6.1 might imply a sequential process course, the true routine in practice might vary with every execution. Adaptation to the current situation, as well as to the current knowledge available is the normal procedure and modeling of all detailed "live" steps of a routine applied is most impractical (if not impossible) to model.

In medical intervention rooms almost everything is an exception. This requires model notations for routines in their "live" environment. As many systems today, medical equipment is affected by dynamic changes in its operational environment. Such systems cannot be simply shutdown to be changed, updated or upgraded and restarted again. This is particularly important for a live saving environment in which adaptation has to be done smoothly, quickly and without quiescence to support ongoing collaboration to meet clinical effectiveness.

We agree with Mulyar, Pesic, Van Der Aalst and Peleg (2008) that imperative decision support systems and their notations limit a practitioners flexibility while putting pressure on the completeness of a model. We argue that the definition of "what" (e.g., tasks and phases) is enriching, while the "how" should be left to the operational staff (e.g., medical practitioners). In case of the Paradigm representation as presented we define a best-practice recommendation divided into phases, with specific rules connected to each phase. Operational staff then should launch the execution of a phase. In case the doctor, by any reason, decides to inspect the left bronchial system before the right, the system should act smoothly to the changing procedure. The sequential process description as in Figure 6.1 can be used for orientation, however,

detailed execution needs to be modeled in an evolutionary manner.

Future work is to present all relevant technical details, also with respect to more aspects than positioning. This is not only interesting in view of mastering flexibility, but also for revealing how larger numbers of McPal-like components could be consistently coupled. The McPal mechanism enables fetching and execution of process patterns from a repository; organization and classification of such a repository is another topic to be investigated.

6.4 Chapter Summary

As a partial answer to RQ3 ("*How can governance address the dynamics of knowledge work across multiple knowledge worker teams?*"), in this Chapter we employ the example of a highly agile medical procedure to contribute to understanding of collaboration and adaptation of unforeseen variations in enterprise process models.

According to Mintzberg (1979, 1993), professional organizations can be governed by controlling the work processes and routines applied. The challenge is to strike the appropriate balance between flexibility and the need for control structures in process models - Adding flexibility while keeping traceability and enabling process improvement on the operational level. Taking the notation of the Paradigm coordination modeling language we provide a guided example of flexible bronchoscopy, a non-invasive medical intervention to construct a computer-interpretable guideline model out of qualitative process data. As current literature suggests, it is most impractical to model all possible variations of a process and we argue for an evolutionary process execution using the McPal mechanism.

In more detail, we achieved the following results: (1) Narrative fragments can be used as a transitional step to specify Paradigm models. In view of flexibility to be added on-the-fly of the computer guided interpretation of the model, we add a number of McPal-like components. Via these we are able to add, in a piece-wise as well as in a consistent manner, specific local variations of dynamics to the various participants or to their roles. Also, dynamically consistent coordination for such variations can be included. (2) An interesting new feature here is, the achievement of cooperation between different McPal-like components; all earlier Paradigm models had at most one McPal. So the McPal technique is getting more distributed, actually well in line with flexible cooperation as occurring in knowledge organizations.

Part IV

The Multi-Project Organization

7

Managing a Project Portfolio across Agile Teams

In the this chapter we discuss the importance of autonomy and self-management for agile methods. As agile teams take over many traditional project management tasks such as planning and internal resource allocation, implications of the methods applied in existing organizational contexts are expected. Our research provides an empirical view on agile project management methods applied in multi-project environments. Following the team perspective we discuss: (1) the implications of agile project management methods on multi-project management and its characteristics, and (2) the importance of routines in agile projects and why they are an important tool of governance.

The research questions investigated are: RQ1 (" *How can governing bodies in organizations of knowledge worker teams understand and steer multiple knowledge worker project teams in practice?*") and RQ2 ("*What are the components of governance necessary to understand knowledge worker project organizations?*"). Further, in order to make RQ1 suitable for our field research, we formulated the following subquestion RQ1c: *"What are the characteristics of agile portfolio management in use?"*

This chapter is based on the following publication[1]:

Stettina, C. J., & Hörz, J. (2015). **Agile portfolio management: An empirical perspective on the practice in use**. International Journal of Project Management. 33(1), 140 - 152 [1]

[1]The author would like to thank Elsevier and his co-author for permission to reuse relevant parts of the article in

7.1 Linking Strategy to Projects

Agile project management methods caused a silent revolution in the way projects are organized and executed (Abrahamsson et al., 2009; Dybå & Dingsøyr, 2008b). While originating in software projects, the methods are gaining increased attention in the general field of project management. In 2011, for example, the term *"agile project management"* for the first time surpassed *"agile software development"* on Google Trends. However, the current methods are bound to a "sweet spot" (Hoda et al., 2010) of small, co-located software projects and individual teams.

In order to break out of this comfort zone and implement the advantages of agile project management in broader organizational contexts, research calls for a view on agility outside of individual projects and teams (Kettunen & Laanti, 2008). One possible perspective, especially prominent in project-based organizations, is that of project portfolio management (PPM). PPM links organizational strategy to the distribution of resources across projects in the portfolio (Martinsuo & Lehtonen, 2007; Cooper, Edgett & Kleinschmidt, 1999). As such portfolios provide an opportunity to make organizations more agile outside of individual projects.

While portfolio management is well established in traditional project management literature, the iterative nature of agile methods introduces new challenges to the current management practice. Agile methods show substantially different patterns of action to traditional projects (Thummadi et al., 2011; Nerur & Balijepally, 2007). They are largely based on recurring activities, so called organizational routines (Pentland & Feldman, 2007), such as iterative delivery of intermediate results or daily standup team coordination meetings (Schwaber & Beedle, 2001; Williams, 2012). Agile software development is fast and flexible due to frequent feedback loops, iterative reviews and close customer contact. Without this direct interaction agile methods loose much of their effectiveness (Hoda et al., 2010; Stettina & Heijstek, 2011a). This is especially challenging for larger organizations with well established routines and structures.

Leffingwell (2007, 2010), Krebs (2008), and Vähäniitty (2012) propose frameworks for agile portfolio management and point out initial benefits and challenges, however, there is a lack of empirical evaluation. While most contributions originate in consulting literature only a few limited single-case studies exist on program management (Kettunen & Laanti, 2008; Laanti, 2008; Laanti, Salo & Abrahamsson, 2011), and a few conference publications exist on the application of agile methods within project portfolios, all in individual organizations (Rautiainen, von Schantz & Vahaniitty, 2011; Kalliney, 2009). In order to close this research gap we take the perspective of the concrete practices applied across three stakeholder teams: senior management, portfolio management and project management. We interviewed project and portfolio management staff in 14 organizations in the Netherlands, Germany and

this thesis.

Sweden on their experiences in using agile methods in the context of IT project portfolios. The 30 interviews resulted in a total of roughly 1600 minutes of recorded material.

In this chapter we report on this study for the first time presenting an insight into the portfolio management practice in multiple organizations applying agile methods. To academics this chapter provides an overview of the portfolio practice domains affected by agile methods, thus enabling an appropriate investigation on the necessary micro-activities to establish agile portfolio management capabilities (Salvato, 2009). To project management professionals it provides an understanding of the potential characteristics of agile portfolios and the implications to be expected when applying agile project management methods in portfolios of projects.

7.2 Related Work

While project portfolio management originates in project management literature (Martinsuo & Lehtonen, 2007), agile project management practices as we know them today originate in the domain of software development (Dybå & Dingsøyr, 2008b). Further, the roots of agility in organizations can be traced back across multiple domains including manufacturing and logistics (Booth & Harmer, 1994). Due to different interpretations across domains the concept can be difficult to define (Laanti et al., 2013). Widely, agile organizations are regarded those that learn fast and are effective (Conboy, 2009; Booth & Harmer, 1994). Agility as a concept to execute and organize software development projects emerged in the 1990s based on ideas found in new product development (Takeuchi & Nonaka, 1986). Agile project management methods such as Scrum (Schwaber & Beedle, 2001; Dybå & Dingsøyr, 2008b) are design oriented and enable frequent feedback loops based upon recurring project cycles (e.g., demonstration of intermediate results). Compared to traditional plan-driven project management methods they embrace project environments as uncertain and enable an iterative delivery of intermediate project results rather than assuming their predictability and a linear sequence of steps from project definition to delivery (Nerur & Balijepally, 2007).

In project management literature the goals of project portfolio management are established as (see Martinsuo and Lehtonen (2007)) as follows: (1) maximization of the portfolio's financial values, (2) linkage of the firm's strategy to the portfolio, (3) and balancing the project within the portfolio with respect to the organization's capacities. There is a number of contributions describing how such a process is implemented in traditional project management practice, most prominently the work by Cooper et al. (1999) and the guidelines provided by the Project Management Institute (PMI, 2008). Although literature generally distinguishes portfolio management from program management by the fact that the projects are content-wise independent, there is an overlap to program management literature. Ferns (1991) distinguishes three types of programs: strategic (group of projects to implement a strategic reorganization, e.g., change of an organizations mission), business-cycle (group of projects linked to a time-

related business cycle such as an annual plan, this configuration is generally understood as portfolio management) and single-objective (a macroproject, so large in size that it is divided into sub-projects, and managed as a group of smaller sub-projects).

While the standard PPM models mentioned above have their specialities the main concept remains the same, they describe mostly linear process steps to *identify, prioritize, allocate, balance* and *review* the projects within a portfolio. In that sense the iterative nature of agile methods with frequent reevaluation of project results might affect current portfolio management practice. Lycett et al. (2004) point at the contextually of multi-project environments. They outline the fact that current frameworks assume an equally effective application of prescriptive and highly structured approaches in all contexts. Recent contributions argue that the complex societal setting of project work is not sufficiently reflected in the available frameworks, neglecting their embedment in context and the relevance of actors and their interactions continuously (re)shaping the project environment (Cicmil et al., 2006). To improve this understanding literature proposes to conduct concrete empirical analyses of project management methods enacted in practice (Cicmil et al., 2006; Pentland & Feldman, 2007; Wenger, 1998).

Agile practices are an integral part of agile methods such as Scrum. In Scrum many project management tasks are taken over by project teams. The practices are concrete team routines to a large extent based upon recurring micro-activities such as daily team coordination meetings, bi-weekly planning and review meetings with stakeholders, or post-mortem reviews (Williams, 2012). As such they make the software project management more explicit by describing team level routines and shedding light on parts of the process not considered earlier. However, these recurring activities make agile methodologies substantially different from traditional methods (see event sequencing study of Thummadi et al. (2011)). It is especially troublesome for large organizations which have to deal with co-existing sequential project management approaches and legacy systems. Here, the perspective of organizational routines (Pentland & Feldman, 2007) can be helpful in uncovering the underlying activities and their implications on existing practice.

Framework descriptions of agile methods applied in portfolio management are provided by Leffingwell (2007, 2010), Krebs (2008), and Vähäniitty (2012). Leffingwell (2007, 2010) describes in his books and his framework description of the Scaled Agile Framework (SAFe)[1] several practices to implement agile methods at enterprise scale. He divides his framework into the levels: portfolio, program and team. On portfolio level the portfolio management team maintains the portfolio vision, allocates resources to value streams through investment themes and defines and prioritizes a portfolio backlog, the highest-level mechanism and artifact holding business and technology development initiatives. On program level a product manager or comparable "chief content authority" Leffingwell (2010) continuously interacts with the portfolio management team and participates in decision-making on priorities of the program backlog. On team level about 5 to 10 agile teams are responsible for implementing executing

[1] http://www.scaledagileframework.com/

the projects following agile project management practices, such as those provided by Scrum. He further defines four core values of the frameworks, viz: (1) alignment (of strategy from portfolio backlog down to the respective team backlogs), (2) code quality (ensured by number of practices), (3) transparency (to build trust end enable better decision making), and (4) program execution (successful execution of the entire program).

In his book, Krebs (2008) proposes a dynamically managed portfolio based upon agile principles with flexible financial models. He divides portfolio management into project, resource (e.g., personnel) and asset (e.g., systems, applications, materialized projects) portfolio management while suggesting to use a dashboard to assess the situation as a whole and adopting progress, quality and team morale as key metrics for the individual projects. Krebs (2008) discusses challenges across these three portfolio domains as: (1) Project portfolio: too many active projects and incorrect mix of of projects, (2) Resource portfolio: lack of vision, too many projects while not enough (right) resources, and lack of feedback, (3) Asset portfolio: legacy systems as roadblocks and underestimation of total cost of ownership. According to him, implementation of a project management office (PMO) and transparency of resources are key to agile project management.

The dissertation by Vähäniitty (2012) discusses agile product and portfolio management in the context of small software organizations. He proposes a framework for connecting business and development decision making through three key processes (development portfolio management, product roadmapping, release planning (Vähäniitty (2012), p.113) across three groups of actors (top management, strategic release management, and software development management, (see Vähäniitty (2012), p.80). According to him the key steps in establishing agile portfolio management are: (1) establishing public prioritized list of all ongoing activities, (2) making sure incentive system do not encourage local optimization, and (3) appointing a steering group to meet and regularly decide on priorities and resourcing.

While the most elaborated views of agile portfolio management are discussed in the references mentioned above, empirical evaluation of agile methods in portfolios and the enactment of the proposed frameworks is scattered. Initial challenges have been reported, especially related to the alignment of business needs and strategy (Kalliney, 2009; Hodgkins & Hohmann, 2007), establishing agile IT project portfolios with prioritization, resource allocation and governance (Rautiainen et al., 2011; Thomas & Baker, 2008) and synchronizing development dependencies (Kalliney, 2009; Hodgkins & Hohmann, 2007). Kalliney (2009) discusses issues concerning the alignment with business strategies and company vision, managing cross-team risks and synchronizing development dependencies as well as handling the knowledge and skill silos of the company. Hodgkins and Hohmann (2007) report their key challenges in the adoption of an agile program management office. They found that Scrum backlogs were insufficient in addressing business needs and introduced roadmaps as a filter to aide backlog prioritization and to communicate strategic intent and business opportunities between the product managers and the technical team. These findings indicate a need for

Table 7.1: Common characteristics of agile portfolio management across literature

	Leffingwell (2007, 2010)	Krebs (2008)	Vähäniitty (2012)	Cases
	Senior Management			
Commitment	● Executive sponsor		● Product portfolio management ● Roadmapping	● Commitment to strategically managed portfolios
	Portfolio Management			
Transparency	● Transparency to build trust and improve decision making ● Alignment of strategy from portfolio backlog down to team backlogs	● Transparency of resources via ROI ● Project Management Office is crucial ● Funnel kept prioritized ● Incremental Return on Investment ● Risk-reward diagrams	● Public prioritized list of all ongoing activities ● Traceability of iteration level work items to high level product and business goals ● Development portfolio management	● Transparency on resources and decision making ● One portfolio for the entire organization ● Strategic backlogs
Collaboration	● "Chief content authority" (e.g. product manager) participates in decision making	● Iterative portfolio balancing	● Release planning	● Frequent portfolio reviews
	Project Management			
Team orientation	● 5-15 agile teams	● No project switching		● Dedicated project teams

further and more integrated research on agility in portfolio management.

While the interest on agile project management grows, there is little empirical evidence on the methods enacted in portfolio management and how the proposed frameworks relate to the characteristics across the domains of PPM practice. Based in the present state of the art we thus formulate the following research question:

- *RQ1c: What are the characteristics of agile portfolio management in use?*

7.3 Method

In this chapter we aim to contribute to the understanding of management practices in a real world context where events cannot be controlled. Thus we chose for a case study research approach as commonly proposed in the literature (cf. Yin (2009)). Qualitative studies allow to research complex problems while developing rich and informative conclusions while engaging practitioners in a constructive dialogue to create a shared understanding (Cicmil et al., 2006).

As current literature on portfolios in agile software development focuses on single cases we chose to conduct a multiple-case study (Yin, 2009).

7.3.1 Case Selection

In our case study research, the unit of analysis (i.e., the single individual case), is an organization with IT project portfolios applying agile methods to manage and develop the endorsed projects. To select our case organizations we followed a replication logic strategy. This strategy, as recommended by the case study design (Yin, 2009) recommends selecting similar cases and dissimilar cases to provide similar and dissimilar results for predictable reasons. Accordingly we have chosen organizations with little experience and recent adoption of agile methods, as well as organizations with up to 10 years of experience with agile software development, as one could hypothesize a better integration of the process. Agile practices have been found especially adopted in small organizations (Hoda et al., 2010). Project portfolios are however more to be found in bigger organizations. We thus set the scope of the study to large (more than 250 employees, at least 3 software development teams) organizations developing software projects using agile methods. To ensure variability across the cases we selected organizations from different industrial domains with a variety of organizational structures.

We created an initial list of 25 organizations from references and Internet search according to the pre-defined selection criteria: large organization, active software development, agile methods adopted, presence of a project portfolio. Within the 25 contacted organizations 14 were chosen according to availability of interview partners (see Table 7.2). As some organizations use different terminology for portfolio and/or program management we use the definition by Cooper et al. (1999), following what Ferns (1991) defines as a group of projects linked to a time-related business cycle such as an annual plan. We have selected the organizations accordingly. The collected data from 14 organizations represent a rich set of fields from insurance, government to media.

7.3.2 Data Collection: Semi-Structured Active Interviews

According to the qualitative design of our study the primary source of our data are semi-structured interviews. Those allowed us to collect rich data while keeping the flexibility necessary for an explorative study. According to Yin (2009) researchers should formulate a research question including potentially important variables, however, should avoid linking variables and theories as much as possible. We know the importance of the software development and the portfolio management process. Based on those we created protocols for semi-structured interviews. The interviews covered the three domains: (1) portfolio management, (2) software development and (3) project handover. Example questions were: *Could you please write down a step-by-step description of your portfolio process, as detailed as you remember? What is your process of prioritizing, allocating, monitoring and reviewing of projects? Which specific*

Table 7.2: Case organizations and descriptive variables

Organization	Industry	Organizational Structure	Project Management Method	Software Development Method	Commitment to Strategic Portfolio	One Portfolio	Other Initiatives in Portfolio	Frequent Portfolio Reviews	Dedicated Teams	Agile Experience (years)	Interviewees and their roles
A	Insurance	Functional	PRINCE2	Scrum/Kanban	❍	❍	❍	❍	❍	10	4 (PRM, PM,C, SM)
B	Investment	Functional	PRINCE2	Scrum	●	●	❍	●	●	1	5 (DPD, PRM, MBU, 2xSM)
C	Telecom	Functional	(Custom)	Scrum	●	●	●	●	●	5	1 (HSD)
D	Auction	Matrix	(Custom)	Scrum	❍	❍	❍	●	❍	0.3	1 (HPPM)
E	Finance	Functional	(Custom)	Scrum	❍	❍	❍	❍	●	0.3	1 (C)
F	Government	Functional	PRINCE2	RUP/Scrum	❍	❍	❍	❍	❍	2	3 (HSD,PM,SE)
G	Finance	Project	PRINCE2	RUP/Scrum	❍	❍	❍	●	●	2.5	1 (CAI)
H	Government	Functional	(Custom)	ScrumBut	❍	●	●	❍	❍	0.1	1 (PM)
J	Government	Functional	(Custom)	Scrum/Kanban	❍	●	●	❍	●	3.5	1 (HSD)
K	Insurance	Functional	PRINCE2	Scrum	❍	❍	●	●	●	3	3 (HPPM, DM, SS)
L	IT service	Functional	(Custom)	ScrumBut	❍	❍	❍	❍	●	4	2 (SRM, PM)
M	IT service	Matrix	PRINCE2	Iterative RUP	❍	❍	●	❍	❍	1	4 (MBU,SM, 2xSE)
N	Social media	Project	(Custom)	ScrumBut	❍	●	●	●	●	3	1 (HPPM)
O	Media	Matrix	PRINCE2	Scrum	●	●	●	●	●	1	2 (HPPM, DIP)

Roles of the interviewees: Head of Project and Portfolio Management (HPPM), Director Product Development (DPD), Head of IT / Systems Development (HSD), Manager Business Unit (MBU), Director Innovation Planning (DIP), Senior Manager (SRM), Coordinator Agile Implementation (CAI), Program Manager (PRM), Project Manager (PM), Project Management Officer (PMO), Delivery Manager (DM), Coach (C), ScrumMaster/Team Lead (SM), System Specialist (SS), Software Engineer (SE).

agile practices were applied (e.g. iterations, standup meetings, pair programming)? Are you satisfied with your current process to manage the IT portfolio, what are your challenges? The interviews took place between May and July 2012, were conducted face-to-face and voice recorded with the consent from the participants. The interview guide has been adopted in course of the study to reflect on comments of participants.

In course of the interviews we asked the participants to write down their activities step by step on a piece of paper as narratives (Pentland & Feldman, 2007), in their own words and as detailed as possible. This allowed us to capture their practice in natural language as well as visually and discuss it in course of the interview. The interviewer was present during the entire interview, would ask questions and discuss the steps with the interviewee. This more active form of interviewing (Holstein & Gubrium, 1997) allowed the interviewers and interviewees to establish a deeper, commonly created understanding of the practice. When available we also asked participants to provide documented process descriptions.

7.3.3 Data Analysis

To analyze the data according to our study design, we first created a full description of each case, then transcribed and coded the interviews and the collected process steps. This was performed by both authors in an interleaved and iterative way. First, the interviews have been transcribed and analyzed using open, axial and selective coding. By open coding we broke down, compared and categorized the transcripts line by line assigning a code and a short summary to each. Coding was performed by both authors on consensus. An example of such a code is: *c_proc_commitment_seniormgmt: "Getting senior management committed"*. Second, the analysis of the process descriptions occurred in two ways: through visual mapping and through coding of narratives as emerged in the interviews. The collected narratives from the transcribed interviews, were coded by inductively deriving a set of categories by sorting the process steps across the organizations. Further, following Langley's framework for building theory from process data (Langley, 1999) we have selected the visual mapping strategy. Using graphical forms allows the presentation of large amount of information in little space and is a useful tool to develop and verify ideas in theory development (Langley, 1999). According to Langley (1999) this strategy requires at least five cases in moderate level of detail to begin with pattern identification. The process descriptions as collected within the interviews have been carefully modeled according to the descriptions of each participant. All process diagrams were modeled and discussed by both authors and sent to the participants for feedback. By embracing textual narratives and visual representations we were able to capture the process, its structural dependencies and discuss them with participants.

7.4 Results

In this section we will describe the practice related findings in our data. Table 7.2 presents an overview on the organizations and their descriptive variables, such as the organizational structure, predominant project management frameworks and the roles of the interviewees. Due to privacy reasons and ethical considerations we anonymized our data and will identify the described organizations with the letters A-O. We will now begin describing our case organizations and their portfolio practices.

7.4.1 Case Organizations

As we can see in Table 7.2 the majority of organizations in our data set are from the financial, governmental, and telecommunications sector from the Netherlands, Germany and Sweden. Most organizations exhibit a functional structure. In all but organization B the adoption of agile methods begun bottom-up, originally starting with individual software development projects. In three organizations the portfolio is managed strategically with top management having an explicit role in identifying, prioritizing and authorizing the projects in the portfolio. In six organizations there is a single portfolio in the whole organization. Six out of the 14 organizations have multiple portfolios. In four organizations (D,E,F,K) the portfolios are part of a respective business unit with prioritization applied locally. The portfolios in that case are prioritized on level of the business unit.

All case organizations have a set of independent projects, thus a project portfolio (Cooper et al., 1999) or a business-cycle program (Ferns, 1991). However, although fitting this classification we observed that some participants call it *"portfolio and program management"*, pointing out the fact that there might be at times also related projects in the portfolio. Further, all organizations have different types of projects and initiatives in their portfolio, a range of supporting activities such as maintenance, replacement or upgrades or implementation of new technologies or techniques. *"Within our portfolios we have four types of projects which are continuity (IT), mandatory (Legal or branch agreements), integration (reduction of complexity) and strategic."*, says head of program and portfolio management (K).

Regarding the applied portfolio management methods, the first observation we made was that none of the organizations explicitly applied one of the frameworks by Leffingwell (2007, 2010), Krebs (2008), or Vähäniitty (2012). Rather, the majority of our participants describe their application of PRINCE2 or an own not further specified general project management framework with own portfolio management practices. PRINCE2 (Murray et al., 2009), acronym for: PRojects IN Controlled Environments, version 2) is a process-driven traditional method similar to the guidelines provided by PMI (2008). As a general project management method PRINCE2 is widely used as a basis framework by project managers without a software development background especially among the Dutch organizations.

As represented in Table 7.2 the majority of our case organizations applies a mix of

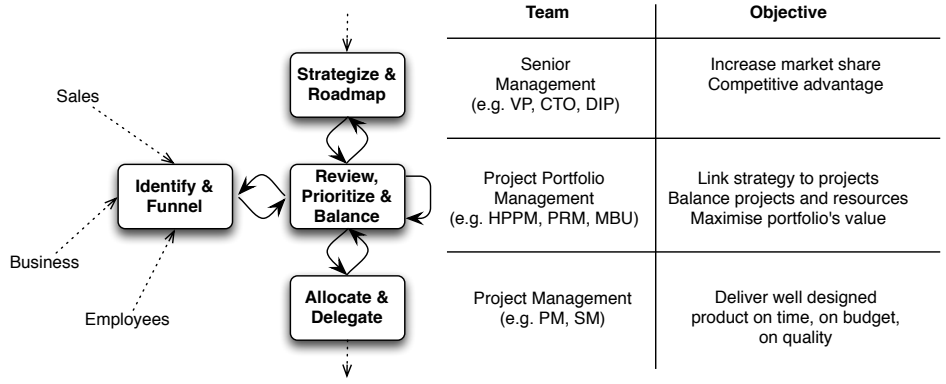


Figure 7.1: Domains of portfolio practice identified in case organizations

PRINCE2 as the general project management framework and Scrum or a derivate as the software development process. General PM methods are thus a major interface to agile techniques. *"We use a combination of Prince2 and agile (Scrum)"*, says a project manager (A). Responsibility for IT projects is generally divided among a project manager representing business and an IT project manager or team lead.

Scrum (Schwaber & Beedle, 2001) is the most applied software project management and development framework. The origins of Scrum lie in the "rugby" approach described by Takeuchi and Nonaka (1986). There a cross-functional team develops a product iteratively in overlapping phases instead of applying a linear process from initial product definition to delivery. Scrum is an adaptation of the ideas to the context of software projects. It defines a set of practices (e.g., reviews, standup meetings), roles (e.g., team, product owners and team leads or coaches, so called ScrumMasters) and artifacts (e.g., work backlogs) to guide the iterative process (Schwaber & Beedle, 2001).

Regarding the specific standard agile practices in use (Williams, 2012), the majority of our case organizations apply standup meetings (daily coordination meetings of the team), development in short iterations (intermediate project results are frequently delivered and reviewed commonly reviewed by the team and project owners), and retrospectives (reflective sessions of the team on work process). What we generally observe is that in each organization a set of practices from the available frameworks is adopted and mixed in practice: *"We use Scrum and Scrumban (Kanban with Scrum elements like Review and Retrospectives) and we make use of some XP elements (e.g. pair programming)."*, says a head of project and portfolio management (N).

Table 7.3: Example of narrative fragments as emerged from the interviews (sorted)

ID	Narrative fragment
...	*Identify & Funnel*
(N)	Management Board prioritizes according to strategy
(P-1)	Project wishes from different staff members and innovation team enter the portfolio
(P-2)	Management board selects projects as advised by portfolio team
...	*Review, Prioritize & Balance*
(A)	IT steering committee decides on budgeting and prioritization
(B)	Business director Product Management comments and prioritizes on road map and backlog
(C)	Strategic Product Managers keep the backlog up-to-date, prioritize and monitor projects
(D)	Portfolio meeting on progress and resources is held every two weeks
(E)	Review of projects ad-hoc
(F)	Steering committee of each primary process prioritizes and monitors projects according to own budget
(G)	A triangle of portfolio manager, lead business change manager and enterprise architect discuss priorities and resources every three months
(H)	Ad-hoc reviews by management, evaluation of large projects
(K-1)	Business unit reviews portfolio every three months
(K-2)	Business unit management decides on projects for a year
(M)	Management team of IT business unit prioritizes portfolio
(N)	PMO prioritizes, facilitates reporting structure and external projects. Reviews take place in company wide meetings every 4 weeks
(P)	Reviews and prioritization is done by portfolio management based on capacity (CAPEX)
...	*Allocate & Delegate*
(B)	Business Director Product Management assigns tasks to six teams (according to their field of expertise)
(C)	Planning Board (SPM+R&D) meets every week, discusses the detailed requirements and delegates to 12 agile teams
...	...

7.4.2 Identified Domains of Practice

During our analysis we identified 49 narratives related to the portfolio practice, we then sorted and organized all narratives into a chronological stream. An example of the sorted narratives can be found in Table 7.3. Considering the narratives and the visual process models created for each organization we clustered the reappearing patterns of action. After a number of iterations including the feedback of participants we identified the practice across three groups of actors (senior management, portfolio management and project management) and grouped the activities into the four following practice domains in Figure 7.1.

Strategize and roadmap: describes the actions taken to define the strategic course of the organization, generally done by highest management (e.g. board of directors (A), governance board (F) or escalation group (K)). In our case organizations such a definition of the strategic course took place between one (A) and three years (B).

Identify and funnel: describes actions where project ideas are collected and enter the portfolio funnel of possible projects. While ideas are obviously created all across the organization, the entry point is generally provided by middle management.

Review, prioritize and balance: is the core of the portfolio management process. These actions generally occurs within portfolio meetings with steering committees. *"Prioritization*

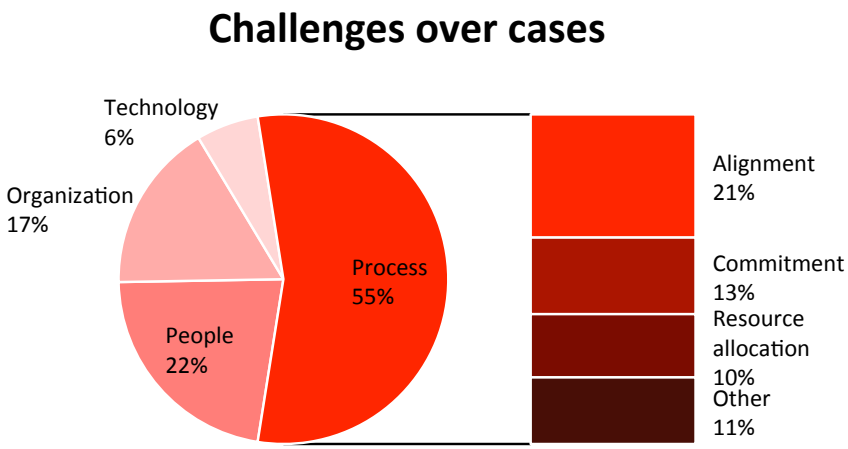


Figure 7.2: Challenges reported in case organizations

and allocation of resources is done by portfolio management (VP level). Within the projects we speak of delegated commissioning; decisions about priorities and resources are always made at portfolio level.", says the project portfolio manager in organization O. These review meetings take place between two weeks (D) and 12 months (K).

Allocate and delegate: Allocation of resources is generally done by a specific portfolio project manager, while delegation is done by project management. A speciality of agile methods is that teams *pull* their work items from the respective backlog. Instead of a project manager defining and delegating the tasks to the team, a backlog of all work items is created, and updated in each iteration. The team members then actively 'pull' their tasks from there. This generally happens in iterations of 1-4 weeks.

7.4.3 Perceived Challenges in Practice

After transcribing and coding all interviews we identified 25 exclusive thematic codes and 51 sub-codes related to the application of agile methods within the portfolios. The coded themes were mentioned in the transcripts in 179 instances of which 99, 55% were related to the process, 22% were related to people, 16% to the organizations and 6% to technology. In this chapter we will focus on discussing these process related themes. The technology related challenges were mostly related to legacy systems, the people related themes were culture and trust, and the organizational challenges related to difficulties in structure (e.g. hierarchies, bureaucracy), making an organization more agile and portfolio governance. Out of the process related themes we found the three most frequently mentioned types of challenges related to

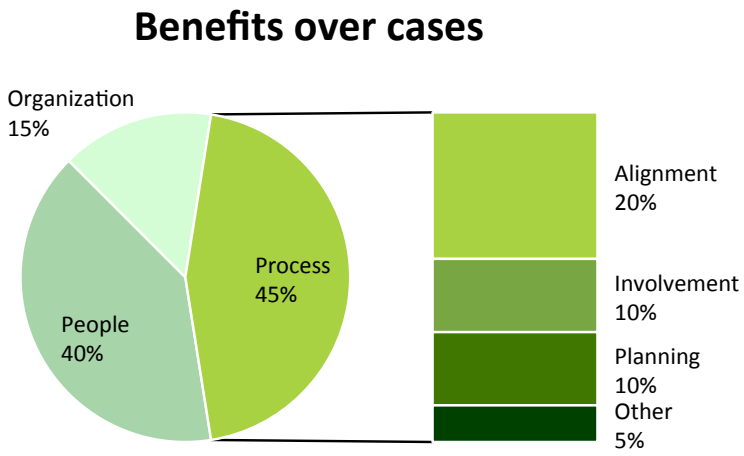


Figure 7.3: Benefits reported in case organizations

alignment with existing processes (37 instances), commitment (23) and resource allocation (19).

Alignment to exiting processes: The alignment to existing project management, software production and business practices was the biggest challenge in the portfolio management process mentioned by participants.

Commitment: In about a quarter of the identified process challenges participants mentioned issues concerning the commitment of staff. Predominantly mentioned is the lacking commitment and involvement of senior management to the software development process.

Resource allocation: The third challenge is related to resource allocation, especially the allocation of teams to multiple projects simultaneously and reshuffling of teams.

7.4.4 Perceived Benefits in Practice

As our interview guide has been tuned to the collection of practice descriptions (i.e., the challenges and implications on portfolio management), the identified benefits emerged directly from the interviews. The benefits reported here have been mentioned by the interviewees and identified during the coding process alongside the challenges. Similarly to the challenges the main identified categories were: process (mentioned 18 times), people (16) and organization (6). Out of the process related themes we found benefits related to alignment and coordination of customer needs (8), involvement of business, customers and maintenance teams (4), as well as planning (4).

Alignment and coordination: Alignment to customer needs has been found beneficial with

agile project management methods. Participants report that agile methods are not always faster but more in line with wishes of customer. Backlogs further improve coordination by providing a shared view on the work items.

Involvement of business, customers and maintenance teams : Involvement of business and customers (e.g. by showing working software on preproduction environment) and IT maintenance teams (e.g. early involvement during development).

Planning: Agile methods have been reported provide more insight and transparency on actual status of projects.

7.5 Analysis and Discussion

Based on the shared understanding of the cases and the collected narratives we will now discuss our findings. Table 7.2 depicts the descriptive variables and the concrete practices perceived as agile in our case organizations. A further comparison of these practices and characteristics across literature as well as the presented cases is depicted in Table 7.1. In order to create a better visual understanding of the interconnections across domains of practice, we depicted the involved actors and their objectives in Figure 7.1.

Agile organizations are generally considered those that learn fast and are effective (Laanti et al., 2013; Conboy, 2009; Booth & Harmer, 1994). In practice, this is enabled by a set of routines stimulating interaction such as project team standup meetings, or reviews with product owners and users. We observe that in our case organizations these routines are expanded towards neighboring practice domains such as general project and portfolio management and production. This closer interaction across the domains is perceived as the biggest benefit. The alignment to existing practices and routines is perceived as the biggest challenge. *"After introducing Scrum within software development we see now that people around us start to stir (business and software management)."*, says director of product development (B).

Due to the self-managing and rather autonomous character, agile teams take over many traditional project management tasks. For example, they coordinate and plan their own work tasks, pull their work from *backlogs* co-defined and prioritized by management (Leffingwell, 2010). Here we observe the preference to work in dedicated, stable project teams as well as a shift in culture towards collaboration based on transparency, trust and frequent interaction. *"The Agile teams are self-managed, which is not yet fully accepted by teamleads of the individual Agile team members. There are actually no more projects, but work that needs to be done. Higher management is used to be in control of projects and now have to trust the Agile teams that work gets done."*, says head of PPM (D).

Agile methods have been implemented bottom-up in the majority of our cases. This is reflected in the fact that characteristics perceived as agile can be mostly found on the project level and portfolio level. The characteristics in Table 7.2 are ordered according to their origin starting with top management on the left to project management on the right. As presented in

the table, dedicated project teams are a characteristic shared across the majority of our cases, followed by frequent portfolio reviews and embracing other activities than projects inside the portfolio to improve transparency of resources. Commitment of senior management to a more active role in the portfolio process is frequently pointed at, however, often lacking. These findings indicate that processual, routine related aspects (e.g., frequent portfolio reviews) as well as structural aspects (e.g., dedicated teams, one portfolio) are associated with and have implications on agile portfolio management.

In order to draw a richer picture on our data we will now proceed to discuss the characteristics across the three groups of actors involved: project management (7.5.1), portfolio management (7.5.2) and senior management (7.5.3). As it is difficult to delineate what is "agile portfolio management" and what not, we compare the collected narratives in context to: (1) the definitions of Laanti et al. (2013), (2) the characteristics shared in the frameworks of Leffingwell (2010, 2007), Krebs (2008), Vähäniitty (2012), and (3) our interviewees' perceptions of agility.

7.5.1 Project Management

According to literature agile project teams take over many traditional project management responsibilities such as assignment of individual tasks, estimation and planning of iterations. In our sample this is especially reflected in the fact that project teams actively pull their tasks from the portfolio. Agile teams are granted with a large degree of autonomy and self-management (Schwaber & Beedle, 2001; Dybå & Dingsøyr, 2008b; Stettina & Heijstek, 2011a). Our interviewees mention their preference towards working in stable teams. This is also reflected in the fact that 9 of the 14 organizations dedicate developers to one team. Furthermore, the responsibility for project success is divided among (1) the *Product Owner* (Schwaber & Beedle, 2001), a formal project manager representing business and (2) the team, represented by the ScumMaster. *"Try to make a good tandem of project manager and Scrum Master"*, says a project manager in organization A.

A: Team orientation

Dedicated software development teams: Resource allocation is recognized an major issue in PPM (Engwall & Jerbrant, 2003). In matrix organizations team members are often allocated to different projects and teams at the same time. *"Frequent switching between projects (determined by management) creates unrest. From a lean perspective this is waste. [..] Although we have scarce resources the amount of projects is too high."*, says coach (A).

Engwall and Jerbrant (2003) discuss that the two main reasons for this challenge in project portfolios are (1) the failure of project scheduling and (2) over commitment (to too many projects at the same time). Project organizations often try to allocate personnel to official project

schedules and priorities vie complex resource planning systems. Due to project reality (e.g. frequent delays, change of plans) this resource allocation, however, often becomes obsolete and resources cannot be available at the scheduled point in time. An untransparent network of actors across multiple teams, assigned to several projects and across diverse departments creates unrest and organizational overhead. Krebs (2008) further discusses project-switching penalties taking into account the return of investment (ROI) (Krebs (2008), p.119).

To counter this half of our case organizations prefers to have dedicated project teams. Having dedicated teams means that software developers are preferably dedicated to one (or two) project(s) at the same time. Our participants mention improvement of quality of their work and less unrest as positive consequence. *"As we have steady dedicated teams, they own and maintain the code also after a project ends. [..] ...we assign work to teams and not teams to work, this is more a steady flow from the company backlog to the teams."*, says director agile project management (N).

7.5.2 Portfolio Management

Linking strategy to budget and projects is a major goal of portfolio management (Martinsuo & Lehtonen, 2007). Enabling transparency by traceability of resources and work items is a recurring theme in literature and our cases. While six of our case organizations have one central portfolio for the entire organization, eight organizations have different project portfolios in the organization without a shared view on allocated resources across the individual portfolios. Transparency is further enhanced by the use of strategic portfolio *backlogs* (Leffingwell, 2010; Vähäniitty, 2012). Such lists of prioritized high-level work packages are further specified and divided into subsequent product and iteration backlogs in project teams. Such enable and maintain traceability of work items throughout the domains of practice. The roles of portfolio and program managers can be affected by an agile transition due to the empowerment of teams, which collaboratively organize their tasks. Transparency can help improving trust and collaboration.

A: Transparency

One portfolio for the entire organization: Literature does not reject having more than one portfolio within an organization (Krebs, 2008; PMI, 2008), however, having more than one portfolio might lead to an untransparent allocation of resources across the projects. Cooper et al. (1999) concluded that one of the clusters of businesses they studied used high quality rated portfolio methods which fit management well. One of these portfolio methods is treating all projects together as one portfolio which is confirmed by Reyck et al. (2005). Participants of our study experienced difficulties having more portfolios and dependencies between projects within different portfolios. *"From within IT we have limited impact on which and how many*

projects are started from the three 'businesses'. Last year one of the businesses started all their projects at the beginning of the year which left little resources for other businesses.", says a manager (F).

Other initiatives grouped within the portfolio: Project portfolio management considers the entire portfolio of projects a company is engaged in (Reyck et al., 2005; Krebs, 2008). All case organizations have different types of projects and other initiatives drawing from the same pool of resources. For instance replacing or retiring systems, maintenance projects or implementation of innovative systems. *"Next to product initiatives we have other initiatives within the portfolio (infrastructure, marketing, legal) which are prioritized by product and validated by the Product Council."*, says the head of agile PPM (N).

While actually all organizations have different types of projects and initiatives, only six of the studied organizations have all initiatives within one portfolio. Blichfeldt and Eskerod (2008) point at the importance to keep all initiatives in sight as invisible projects and initiatives often drain resources originally assigned to the portfolio. Participants of the other eight organizations expressed their frustrations and worries about this situation. Head of system development department of organization F is not in control according to what projects are started when and making IT projects more visible.

Strategic backlogs: Usage of strategic product backlogs encapsulating highest strategic objectives has been mentioned amongst several participants as a key link to agility. Literature discusses those in form of portfolio backlogs (Leffingwell, 2010), product content backlogs (Laanti, 2008) and roadmaps (Vähäniitty, 2012). These backlogs consisting of "epics" as the highest level objectives (Vähäniitty, 2012; Leffingwell, 2010, 2007), are broken down, further specified and linked to concrete team backlog work items as the teams move through the iterative process. *"Projects are managed by a company backlog approach on initiative levels, means that the PO-group is prioritizing all initiatives and teams pull the work from there."*, says head of project portfolio management (N).

B: Collaboration

Close collaboration across the domains: Increased collaboration across the domains of practice is frequently associated with agile methods in our case organizations. Shared understanding on strategy and projects is constantly negotiated and evaluated by the involved actors. In order to establish such a shared vision the actors need to be willing to collaborate across their domains to establish and pursue a common vision.

Collaboration based on recurring patterns of action is discussed by Leffingwell (2007) across team, program and portfolio levels, and by Vähäniitty (2012) across "top management", "strategic release management" and "software development management". Further, Hanssen and Fægri (2008) discuss the integration of agile software development and software product line engineering to support the company's strategic and tactical goals by combining three

interacting customer-centric processes: strategic (roadmapping, business cases), tactical (agile methods) and operational (day-to-day SE activities). *"I see much more communication amongst I&A and the Business and also amongst departments of I&A. While previously developers transferred their software to network- and system engineers, now they help them implement their software."*, says team lead PPM in organization D.

Agile methods largely rely on direct communication. In organizations consisting of multiple teams documented knowledge becomes necessary and needs to be supported by appropriate artifacts and templates. Such have an influence on success and sustainability of a practice and need to be chosen carefully (Stettina, Heijstek & Fægri, 2012). *"Agile goes beyond the software development department of an organization. [...] All documents (FO/TO, etc) offer false security about the quality of a project. The result is a moving target and the world has changed during preparation."*, says manager ICT (B).

Sufficiently frequent portfolio reviews: Agile methods stimulate collaboration on project level through recurring routines, however, they make frequent collaboration also more necessary on portfolio level. If project teams can deliver intermediate results more frequently, they neutrally need to receive more frequent feedback on what they should deliver next. *"..keep peace in the portfolio process..."*, says manager (B). How often portfolio reviews take place depends on the particular context. For example, if an organization operates in high velocity markets exposed to a big competition and project teams can deliver in intervals of 2 weeks, portfolio reviews in annual cycles will not be frequent enough to provide the teams with sufficient feedback. The majority found monthly reviews appropriate.

7.5.3 Senior Management

Top management support is considered one of the most important factors for success of individual projects (Young & Jordan, 2008) and is frequently mentioned by our interviewees. However, it also is one of the biggest challenges for organizations to link strategy to projects, especially when implementing the concrete actions (Aubry, Hobbs & Thuillier, 2007). Although highest management should have an explicit and important role within the portfolio practice, in only three out of the 14 organizations it is the case.

Commitment - Management commitment to strategically managed portfolios: The participants in our study repeatedly name involvement and commitment of senior management to the practice and the integration of IT with remaining businesses as crucial for an agile portfolio. Literature underscores the importance of having strategic management decide on project portfolios (Dye & Pennypacker, 2000; Cooper et al., 1999) and is a success factor for software projects (Chow & Cao, 2008). However, strategic management seems not aware of the possibilities this offers. While top management acknowledges the success of agile methods active participation is often missing. Almost all interviewees are not satisfied with the lacking exchange.

Getting commitment of strategic management demands a management view on agile software development. In most of our case organizations (all but B) agile methods were implemented by individual teams and then spread throughout the organizations without little or passive notice of senior management. After performing try-outs, which are ofnen not at strategic management level, people want to continue but get stuck on management (Boehm & Turner, 2005). We have observed uncertainty about possible shifts in organizational roles, especially among managers as agile teams take over certain aspects of traditional project management such as planning and coordination. *"But there is a point at which the organization cannot be effective without executive leadership taking a role."*, comments Leffingwell (2007), (p.299). He highlights the importance of executives sponsoring the adoption, awareness and appropriate communication (Leffingwell, 2007).

7.5.4 Limitations

Although we employed a rigorous method and payed particular attention in selecting our case organizations and establishing a shared understanding on their practice, there are limitations to our study. The main limitation of this report lies in the limited amount of cases. Although we obtained a relatively large data set including the perceptions of 30 participants on their practice in 14 organizations, our sample might be difficult to reproduce and is not representative. To address external validity we use a replication logic strategy and compare our findings to the existing frameworks of Krebs (2008), Vähäniitty (2012), and Leffingwell (2007, 2010). A further limitation is the qualitative design of our multiple-case study. Our data is based upon perceptions of participants who might have a biased view on their work process (Pentland & Feldman, 2007). To improve construct validity and overcome intrinsic biases we applied triangulation by using multiple informants (e.g. conducting interviews on portfolio, project and development team level) and establishing a shared understanding through the application of active interviewing. An in-depth ethnographical research (Salvato, 2009) is advisable at a further stage to explore the interaction across the routines. However, considering the explorative nature of this work, the amount of organizations and participants ensures a good foundation for further studies.

7.6 Recommendations for Research and Practice

The results of this study point to a number of recommendations to practice and interesting questions for further research. To align project management and IT project management organizations often bind two respective roles: a formal project management representing business and a ScrumMaster representing the team. Commitment of senior management is one of the biggest issues when establishing an agile portfolio. As most of the adoptions of agile practices happen button-up, it is advisable to find a top management sponsor who supports

the adoption. Awareness sessions and clarity about the implications are crucial to gain staff commitment.

The domains of practice identified enable further research on more detailed activities important to consider while implementing an agile portfolio. What is the best governance structure for an agile organization? How to enable strategic management in agile portfolios? How should a good contract look like when working in agile projects? Legacy processes are to be found in all established organizations, what are good strategies to adapt existing practices in context? The micro-activities and organizational routines involved are important for the development of capabilities (Salvato, 2009). If we want to understand agility on the level of organizations we need to better understand the interplay of practices across functional roles. Further, there is an overlap of principles in agile project management and concurrent engineering. Comparing portfolio management experiences in concurrent engineering settings are likely to contribute to further understanding of multi-project management in fast learning and effective organizations.

7.7 Chapter Conclusions

As an answer to RQ1c *"What are the characteristics of agile portfolio management in use?"*, and as a partial answer to RQ1: "*How can governing bodies in organizations of knowledge worker teams understand and steer multiple knowledge worker project teams in practice?*", we discuss the characteristics of agile knowledge worker organizations as follows: (1) transparency, (2) collaboration, (3) commitment, and (4) team orientation.

As a partial answer to RQ2: "*What are the components of governance necessary to understand knowledge worker project organizations?*", we point out (1) routines, (2) organizational structure, and (3) values, as important components of governance in knowledge worker organizations.

In this chapter we contribute to the understanding of portfolio management in organizations applying agile project management methods. Existing literature provides either little empirical evaluation of agile portfolio management frameworks in use, or provides evidence from individual cases only (Laanti et al., 2011; Laanti, 2008; Rautiainen et al., 2011; Kalliney, 2009). In line with research on actuality of projects (Cicmil et al., 2006) we thus compare our data on the practice in use to the frameworks proposed by Leffingwell (2007, 2010), Krebs (2008) and Vähäniitty (2012).

Stemming from interviews with 30 participants in 14 organizations, in total 1600 minutes of recorded material, our analysis indicates a common ground with shared characteristics across the frameworks proposed and our cases as presented in Table 7.1. In the vast majority of our case organizations agile methods have been initially adopted in individual projects not following a particular agile PPM framework. After a successful application in projects the importance to align the portfolio management practice becomes visible. Our data indicates

that agility enabled on project level by recurring routines such as iteration reviews (Williams, 2012) is expanded towards neighboring domains of practice such as portfolio reviews. Our participants indicate a demand for more interaction across the domains and across strategy, tactics and operations (Hanssen & Fægri, 2008). However, with the increased frequency of interaction in projects and with the self-managing character of agile teams, current portfolio management practices might need to be adjusted to fit this enabled agility. Based on our observations above we have found implications of agile methods on three aspects of the portfolio practice:

1. *Routines: the frequent interaction based on routines in projects (e.g. reviews, standup meetings) stimulates the need for an appropriately frequent interaction in neighboring domains of practice (e.g. in PPM).*

2. *Structures: due to the self-managing nature, agile teams take over aspects of traditional project management. This has implications on the role of project and portfolio management. Further, work in stable teams is preferred in our case organizations.*

3. *Values: in order to support a closer interaction across domains of practice, a shared understanding how such a closer interaction could look like needs to be in place.*

Agile organizations are considered those that learn fast and are effective (Conboy, 2009; Booth & Harmer, 1994). While it is difficult to delineate what is agile and what not, we follow the advice of Laanti et al. (2013) and compare the concrete practices applied. Based on those we observe the following characteristics shared across the existing frameworks and our cases.

1. *Transparency of resources and work items, improving trust, decision making, and resource allocation.*

2. *Collaboration, close collaboration based on routinized interaction and artifacts enabling frequent feedback-loops across the domains.*

3. *Commitment, to strategically managed portfolios.*

4. *Team orientation, removing unrest in resource allocation and building capabilities in teams.*

From these observations we may conclude that agile software development evolves into agility in project management. It is a learning process which requires a consideration of routines, structure and culture. Long-term experience with agile methods in individual projects alone is not sufficient for an appropriate integration of the practice into an agile portfolio. It takes time to overcome the challenges in resource allocation and silo thinking. However, if large organizations want to learn fast, be more effective and integrate entrepreneurial spirit in their operations they might want to address these challenges and reflect upon the underlying routines in context.

8

Creation and Transfer of Knowledge Deliverables across Agile Teams

In this chapter we provide an in-depth view on (1) the patterns of action that agile software development teams apply to develop software products and (2) that artifacts they use to support the development of the project and the planned project handover. This Chapter investigates once more RQ2 ("*What are the components of governance necessary to understand knowledge worker project organizations?*") and RQ3 ("*How can governance address the dynamics of knowledge work across multiple knowledge worker teams?*"). In order to make RQ2 suitable for our field research, we formulated the following subquestions RQ2d: *"Which patterns of action do agile project teams apply during software project handover, maintenance and continuous development?"* and RQ2e: *"Which documentation artifacts do the teams perceive useful during the process of project handover, maintenance and continuous development?"* The attempt to provide a potential answer to RQ2 and RQ3 will be supplemented by the answers from Chapters 9-12.

This chapter is based on the following publication[1]:

Stettina, C. J., & Kroon, E. (2013, June). **Is There an Agile Handover? An Empirical Study of Documentation and Project Handover Practices Across Agile Software Teams.** In 19th ICE & IEEE-ITMC International Conference, The Hague, Netherlands. [1]

[1]The author would like to thank IEEE and his co-author for permission to reuse relevant parts of the article in this thesis.

8.1 Agile Project Handovers

The increasing adoption and acceptance of agile methods in large software organizations leads to a reconsidering of the possibilities. It is feasible to move the techniques from a team mentality into the enterprise level? In this Chapter investigates RQ1 (" *How can governing bodies in organizations of knowledge worker teams understand and steer multiple knowledge worker project teams in practice?*") and RQ2 ("*What are the components of governance necessary to understand knowledge worker project organizations?*").

Agile practices (cf. Williams (2012)) heavily rely on direct communication of team members since traditional documentation is perceived as burdensome and ineffective by software engineers (Lethbridge et al., 2003). Knowledge sharing based on recurring practices, such as agile stand-up meetings and socialization of team members, enable the development of agility and oppose communication errors on the team level (cf. Melnik and Maurer (2004)). Yet, agile teams are not isolated and need to share common a infrastructure. A range of challenges related to the sharing of inter-project knowledge has been reported (Rautiainen et al., 2011; Stettina & Heijstek, 2011b). These challenges have an impact on the project handover. Even stronger, they are likely to affect the success of project handovers. Below we mention the examples of such challenges.

Rautiainen et al. (2011) mention project handovers as a source of delays in agile software production programs. Petersen and Wohlin (2009) mention troubles on handovers between requirements and design work. Kettunen and Laanti (2008) discuss that agile organizations are moving towards a collection of dynamically reconfigured virtual organizations combining the advantages of small entrepreneurial companies with large-scale production economics. This involves collaborators and subcontractors. Solely relying on direct communication to maintain agility seems impossible. In agile teams this can lead to gaps of knowledge and architectural integrity when original participants become out of reach or when team boundaries are reached (Abrahamsson et al., 2003; Ramesh et al., 2007; Stettina & Heijstek, 2011b). When projects and knowledge are not transferred seamlessly these gaps can hinder agility of organizations on the program and portfolio level.

Although being an integral part of the software lifecycle there has been little research on the project handover (Khan & Kajko-Mattsson, 2010). To shed some light on the process, this chapter takes the perspective of *patterns of action* which the involved teams follow and the artifacts they use. Our exploratory research contributes to (1) the understanding of patterns of action enacted across the teams and (2) the role artifacts play in establishing agile organizations. To software project managers and coaches it provides understanding of the handover as a process and how to improve it in practice.

Table 8.1: Documentation artifacts in software organizations and respective examples in practice.

Strategic	Vision (S1), Mission (S2), Operational Model (S3), Year Plan (S4), Road Map (S5), Epic (S6), Project Proposal (S7), Business Case (S8), Make or Buy SWOT (S9)
Project Management Office	Project Portfolio (PO1), Project End Report (PO2), Protocol of delivery (PO3), Stakeholders RACI (PO4), Estimate Sheets (PO5),Status Reports (PO6), Lessons Learned (PO7), Customer Evaluation (PO8), Request for Quote (PO9), Mid term Evaluation (PO10)
Project Management	Project Initiation Document (P1), Project brief (P2), Meeting Minutes (P3), Contracts Supplier (P4), Backlog (P5), Project Roadmap (P6), Communication Plan (P7), Conguration Management Plan (P8)
Requirements	Requirements Specification Document (R1), User Stories (R2)
High-level Design	Software Architecture Document (HD1), UML Package Diagram (HD2), Software Architecture Model (HD3)
Detailed Design	Functional Design (DD1), Interaction Diagrams (DD2), Database Architecture (DD3), UML State Diagrams (DD4)
Low-level Design	Technical Design (LD1), Code and Comments (LD2), UML Class & Object Diagrams (LD3), Visual Elements (LD4), Wireframes (LD5)
Specifications	Software Interfaces (SP1)
QA	Test Plan (Q1), UAT Plan (Q2), Stress Report (Q3), Server Analysis (Q4), Technical Acceptance Criteria (Q5), Deployment Manual (Q6), Organizational Positioning (Q7), Known Issues (Q8), Issue and Bug Log (Q9), Risk Log (Q10)
User	Business Process Description (U1), User Manual (U2), Editorial Notes (U3), Work Arounds (U4), Installation guide (U5), Introduction (U6), FAQs (U7)

8.2 Related Work

As stated above there has been little research on the handover. This also holds for the knowledge transfer within the project. Khan and Kajko-Mattsson (2010) acknowledge the complexity of the process. They observe a high dependency of (1) the process on context, (3) wide lifecycle span and overlap with development, (3) pre-delivery and post-delivery maintenance processes.

Knowledge sharing and creation routines are important for project handovers. Knowledge creation is tightly linked to human action (Nonaka & Takeuchi, 1995) and if we want to understand knowledge creation we have to consider patterns of human action. Routines based on direct communication, such as stand-up meetings or iteration reviews in agile teams, have drawn our attention, not at last with the studies of knowledge management in Japanese companies (Nonaka & Takeuchi, 1995). As such we understand routines as *patterns of action* as defined by Cohen et al. (1996). They are: (i) recurring (they can take place at different times, or involve different actors), (ii) selectable (meaning that there are forces that make them more or less likely to happen) and (iii) set in an organizational context (the actions are not those of a single individual).

Nonaka and Takeuchi identify four modes in which new knowledge emerges: socialization (e.g., standup meetings), externalization (e.g., writing, creating a product backlog), combination (e.g., creating design from requirements), and internalization (e.g., reading and making the contents part of own line of reasoning). As agile methods mostly rely on direct communication and socialization of knowledge, team members must share a common understanding in order to contribute effectively. When a team adopts a project (e.g., a maintenance or continuous development team receives software code created by a supplier after a project handover) it does not have access to that shared understanding. This team then needs to acquire the knowledge either through socialization (with the original team members) or through internalization (by reading project documentation). In case of project handover, documentation artifacts thus should ideally represent effective boundary objects promoting an effective (re-)construction of a shared understanding across different functional domains (cf. Carlile (2002)).

Next to human action documentation, artifacts carrying the information transported are important to consider (Stettina et al., 2012). In their studies on industrial application of documentation in software projects Lethbridge et al. (2003) found that (1) documentation is frequently out of date, (2) systems have frequently too much documentation and (3) poorly written documentation. Forward and Lethbridge (2002) suggested in this respect that documentation should be used as a tool for communication rather than an accurate up-to-date artifact. In Table 8.1 we expanded the categories of Lethbridge et al. (2003) to include inter-project documentation and added examples from our data set. The table consists of documentation artifact types and concrete artifacts as applied across our cases. The categories will be used in our research design.

Although addressed in the Agile Manifesto by its second guiding principle, research on documentation in agile teams and project adoption is rare. Stettina et al. (2012) found that software engineering teams perceive updating formal design documentation iteratively as a burdensome task. Especially when represented in textual documents, it is not sufficiently adding feedback and hampering collaboration among team members. In Chapter 5 of this thesis I discuss that a majority of agile practitioners perceive documentation important or even very important but that too little documentation is available in their projects. As a consequence knowledge of older projects las been lost (compare Stettina and Heijstek (2011b)). Hoda et al. (2010) count writing documentation to the context-dependent practices in agile methods. They describe the use of a project dictionary to bridge the language gap between development and customers by using an editable online document. de Souza, Anquetil and de Oliveira (2005) found code and comments, data models and requirements as the most important artifacts to software maintainers. Karlström and Runeson (2006) report in their case study that XP teams often maintained all documentation needed, which was, however, not realized by management due to its informal nature.

In this chapter we are interested in the patterns of action that the teams follow during project handover and the documentation artifacts which they perceive useful. Based on the

present state of the art we thus find it appropriate to pose the following research questions:

- *RQ2d: Which patterns of action do agile project teams apply during software project handover, maintenance and continuous development?*
- *RQ2e: Which documentation artifacts do the teams perceive useful during the process of project handover, maintenance and continuous development?*

8.3 Method

In this chapter we want to create an understanding of handover practices in the real-world context of software development projects. With our research questions in mind we thus followed the advice by Miles and Huberman (1994) and applied a mixed methods approach to collect data. We used qualitative process descriptions, ethnographical notes, semi-structured and informal interviews (see 8.3.1) and linked them to quantitative questionnaires and artifacts developed in course of the project (see 8.3.2). A mapping of data sources can be found in Figure 8.1. We further divided our research design into a prestudy and a main study which enabled us to update the design during the study while including two perspectives: an in-depth perspective on small size projects and a broader view on medium and large size projects.

The prestudy was conducted at Leiden University in three software development projects with 17 teams and a total number of 61 software engineering students working between 6 and 14 weeks on their assignments. The goal was to observe in-depth which steps software engineering teams would follow during project transfer and adoption in small size and short-term projects. The main study was executed with a Dutch service provider, offering telephone, broadband Internet and television services. To support its operations the company maintains its own IT department of about 230 employees (including business analysts, project managers, application developers, architects, application managers, testers and management) and runs several outsourced IT projects. While the company was the initiator of each of the here presented projects, the development was executed with different third-party software development organizations. The goal here was to see how medium to large projects transferred from external parties are being adopted within the customer organization.

For the scope of the process we choose the moment of the initial contact of the two teams (e.g. a first e-mail exchange or a handover presentation) for the prestudy. For the main study it was more important for us to present a bigger picture as the processes were much more conjoined, we thus chose *requests for quotation* as a starting point.

8.3.1 Qualitative Data Sources

Collecting practices of software project adoption such as human behavior requires qualitative data and qualitative observations in real world contexts. Following Langley's (Langley, 1999)

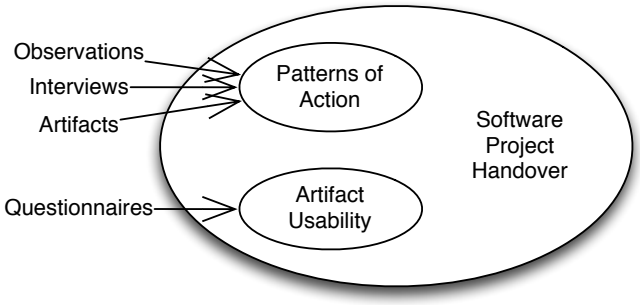


Figure 8.1: Mapping of empirical data

framework for building the theory from process data, we have selected visual mapping which requires at least five cases in moderate level of detail. It begins with (1) pattern identification. Then, using graphical forms (2) allows the presentation of large amount of information in little space and is a useful tool to (3) develop and (4) verify ideas in theory development (Langley, 1999).

In order to collect patterns of action after system delivery of the prestudy the author of this thesis was involved in the projects as a coach and product owner. We asked the participants of each project to discuss and draw their handover process in a sequence of steps as an UML activity diagram on paper. Thereafter we discussed the process with each group. During the main study we conducted observations, semi-structured and informal interviews on site and had access to all relevant artifacts. The co-author Kroon was closely aligned with the company and engaged in daily activities with the software teams. To broaden the scope of our research we have conducted a number of open ended and semi-structured interviews to collect qualitative feedback on the project environment from within application management, development and maintenance. In total we conducted over 10 interviews. Sample questions were: *What was your major challenge during/after project handover? How was the cooperation with the external party? What was the role of documentation in your project? Who received what documentation? What sources of information did you find most useful after project handover?*

8.3.2 Quantitative Data Sources

In line with our second objective we measured the applied project handover process against the usefulness of documentation artifacts. As an objective measure of artifact usefulness we applied a method discussed by Dybå and Moe (1999). We administered surveys to the prestudy and main study participants using the categories as inspired by Lethbridge et al. (2003) and

assessed the usefulness by placing two opposing subjective rating scales accompanying each question: (1) the actual usefulness vs. (2) the believed future importance. The gap between believed importance and usefulness can be regarded as a measure for disaffection with a given solution (cf. Dybå and Moe (1999)). To prevent inconsistency among the rating items we used *Likert scales* consisting of 5 items: Extremely =5, Very=4, Moderately=3, Slightly=2, Not at all=1. To measure the level of agreement among the respondents we calculated the population variance among the answers. Variance ($\sigma 2$) is a measure of how far each value in a set of responses is from the mean. A lower variance corresponds with a greater level of agreement within the quantitative data. The maximum variance in a team is the variance of the maximum and minimum values that can be given in response to an answer. The minimum variance is 0, denoting complete agreement. On a Likert scale ranging from 1 to 5, the maximum variance is 4.

8.4 Results

In this section we first describe the data collection within the prestudy and the main study projects and then proceed to address the two research subquestions (RQ2d and RQ2e). To support our findings we will introduce figures guiding the reader through: 1() the descriptive variables for each of the projects (Table 8.2), (2) the adoption process the projects followed (Figure 8.2), and (3) the participants perceptions regarding usefulness and future importance of the respective categories (Table 8.3).

8.4.1 Patterns of Team Practice

Figure 8.2 depicts the adoption process of the three prestudy projects (A-C) and the four main study projects (D-G) as emerged from the observations, semi-structured interviews and the developed artifacts.

In the prestudy each handover began with a project demonstration of the team that developed the software, then the maintenance teams would receive access to the repository. As each of the three projects had at least three teams assigned, we collected the process steps for each team and grouped them according to the visual mapping strategy (Langley, 1999) into one single process thread in Figure 8.2. When asked about their major challenges during the adoption, team members generally referred to issues related to building and understanding the code and its dependencies.

In our main study organization, each incoming release stemming from internal development or third party followed a 4-step process accompanied by a deployment manager: (1) transfer of incoming release to test management, (2) perform integrated testing, (3) user acceptance testing and (4) production. Projects with external development continued internally also follow this process. Steps 1+2 were performed by maintenance staff. The patterns that the main study

Table 8.2: Descriptive variables and project results (O=Project Owner, S=Supplier)

	Small size and short term projects			Medium and large size projects			
Project	**A) Image-to-UML**	**B) UML-Enhance**	**C) Open-LearningLab**	**D) Retail Information System**	**E) Customer Order Manager**	**F) Corporate Website**	**G) Service Delivery Platform**
Persons in project	28	4(O)+3(S)	2(O)+30(S)	16(O)+8(S)	10(O)+7(S)	30(O)+37(S)	23(O)+4(S)
Teams in project	8	3	6	2	2	6	3
Project length (weeks)	14	8	6	30	24	43	65
Sprint length (weeks)	1	1	1	4	4	4	4
Project size	Small	Small	Small	Medium	Medium	Large	Large
success adoption	●●●●	●●●○	●●●○	●●●○	●●●●	●●○○	●●●○
success time	●●●●	●●●○	●●●●	●●○○	●●●●	●●○○	●○○○
success budget	-	-	-	●●●●	●●●●	●●○○	●●○○
success quality	●●●○	●●○○	●●●○	●●●●	●●●●	●●○○	●●●●
doc transfer	at the end	at the end	at the end	at the end	each sprint	at the end	later by sprint
major challenges	understanding code	building, dependencies	understanding the system,	business participation, roles, responsibilities	maintaining system knowledge	handover in running organization, poor docum.	instruction maintenance staff, late documentation

teams followed were much more interdependent with the development process so, we decided to include these steps in Figure 8.2. We will now proceed discussing the four main study projects in detail.

Project Retail Information System (RIS) is a project delivering sales and knowledge base functionality to large retail stores allowing the personnel to sell the companys products on-site and help answering most important customer questions. The project had the biggest gaps on design and requirements documentation. Only two documents were transferred at the end of the project: functional and technical design. Requirements were originally documented in the PID (Project Initiation Document) 1.5 years before the project start. During project execution, however, requirements were edited and added in JIRA[2] by the software supplier. After the project, only the original requirements were available. An extract of the updated requirements stored in JIRA was not transferred. Despite a review of the received design documents by the end of the project, the software design had to be completely redone (compare Figure 8.2, box D2). The project was hosted internally while the software was transferred to maintenance when functionality piled up in a few iterations. *"There was too little documentation and it did not say anything about the application itself. I can't say that these documents were of any use to me. There was only a small user guide delivered in the very end, but no service level*

[2]http://www.atlassian.com/software/jira/

documents or documents about the working of the system. For example, if I push the button to send a message, who receives the message? I really don't know. The working of the system had to be found out by myself. From time to time I contacted the development company for some questions", says the functional application manager from project D.

Customer Order Manager (COM) Delivered an end-user portal where potential customers can register and order products from the company's catalogue including personal details, products and installation appointments. In case of COM we see that (1) formally the software has never been handed over from the software supplier, and (2) the project has never really been closed but instead went straight into maintenance. Incidents and improvements were registered through a JIRA dashboard shared among the parties. *"In JIRA we keep all changes important for communication. All the changes, we also collect documents, communication and all documentation. [...] Internally we use it a lot. I have an internal account and one with the supplier."*, says the product owner. Maintenance personnel were involved since the first sprint delivery (Figure 8.2, box E2). When a certain amount of requirements was reached the requirements are handed over to the supplier and a sprint has been initiated. The main challenge for the project was not during but after project adoption on the side of the project originator. While the received artifacts were described as useful, they were not kept up to date during the maintenance phase. This led to a large number of exceptions and information on products stored in the system was only known to the product owner and not documented. For new colleagues helping to maintain the system, further development was impossible without documenting that tacit knowledge.

Within Corporate Website (CW) different web portals of the company had to be united to create a common brand identity. To enable communication with the customer (account management) a single sign-on and a personal environment were to be unified with support of widgets and 3rd party applications. The transfer took place in several stages. Similarly to Retail Information System the software was delivered when sufficient functionality piled up in iterations. Documentation handover was first: one big ZIP file with a large amount of documentation (too much to really go trough to find what you need, most used documents were filtered and stored), later after pushing the supplier: bits and pieces of detailed design documents got handed over, but not everything, so designs had to be recreated. Documentation was old as soon as it got transferred (this impacts usefulness), because changes were already made. An interviewee stated that documentation on designs should be up-to-date with production, but this was not the case. There was also too little documentation about proprietary software parts such as the backend CMS, which the project originator had to work out then.

Service Delivery Platform (SDP) is a project to reduce the number of applications and simplify the process of delivering streamed content to the end-user on demand. SDP was ori-

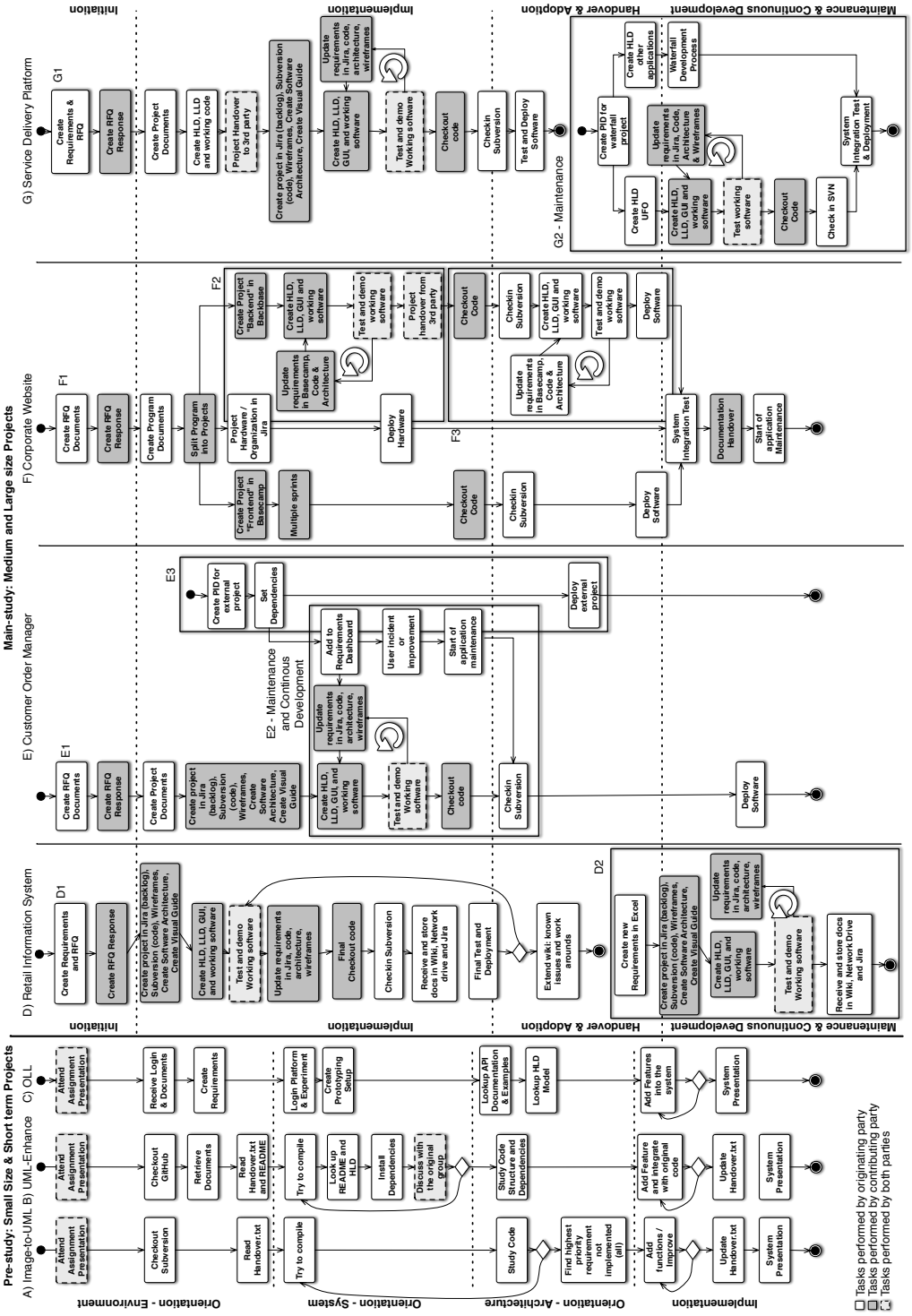


Figure 8.2: Project team practices during software project adoption

ginally started as an internal project. Moreover, it was outsourced, as it was not a core business of the company. Similarly to Customer Order Manager (project E), maintenance personnel was involved at each sprint delivery presentation (compare Figure 8.2). A characteristic of SDP was that, while maintenance was with the supplying party, hosting and high-level design was still provided by the initiator (compare Figure 8.2, G2). After the handover instructing maintenance team members who were not originally involved with the project became a problem. *"There was a steep learning curve for the rest of the application management team that was not directly involved. They asked me a lot of questions, which I had to answer off the top of my head. I didn't have a reference document and I got all these questions from my colleagues. With the document it was much easier to transfer knowledge to my colleagues."*, says the application manager. After a request of the application management during the project a software application development handbook was created and used to brief new colleagues.

As we can see in Figure 8.2, during the adoption we observed three orientation phases before team members would start implementing new requirements into existing software code: Environment, System and Architecture. The first phase, Environment (1a), describes how the team assigned to maintenance and/or (1b) by further development of the project it learns about the environment of that particular project, that is the technical dependencies such as operating system, necessary software, and how to install the software are documented. The second phase, System, describes how the team or the particular engineer learns about the (2a) low-level design, (2b) the source code, (2c) how to compile it and the (2d) external dependencies to be used. In the third phase, Architecture, (3) the team learns how the software has been designed architecturally in order to know where new functionality should be added in the existing structure of the software. Of course, the division into three process phases is a strong simplification as the phases are highly interdependent. Especially as software developers like to work on program code as the main artifacts (Stettina et al., 2012; de Souza et al., 2005) they are likely to try compiling the software, installing dependencies and compiling again. However, one can expect that software developer have to go through these phases in order to contribute efficiently to the project.

8.4.2 Usage of Artifacts throughout the Process

In course of the research, we conducted a document analysis and sorted all artifacts according to their creation and usage in time. The majority of artifacts created and used during the project emerged in the implementation phase. In course of project adoption towards maintenance and continuous development generally requirements documentation (R1), detailed design (DD1), then high-level design (HD1) and low-level design (LD1) artifacts are being used. A variety of documents was also produced for project management and the project management office, mainly used for reporting (e.g. project end reports). During project closure it is on one hand

project end reports (PO2), customer evaluations (PO8) and lessons learned documentation (PO7) created, on the other hand known issues (Q8) of the delivered software are collected. The majority of the artifacts serves as input for the next respective phase in the process while not being used otherwise. Following the adoption steps the usage of artifacts is (1) Environment: User, quality assurance, requirements and detailed design, (2) System: High-, low-level and detailed design and (3) Architecture: High-, low-level design and software interface specifications. The majority of artifacts reused after adoption are design artifacts, user and QA documentation. A variety of documents are also produced and used by project management, the project management office and QA for reporting (e.g., project end reports). These are of course quite helpful to trace issues after something went wrong, however, are not so during the adoption.

8.4.3 Usefulness of Documentation Artifacts for Project Transfer and Maintenance

Table 8.3 depicts the results of the questionnaires and the participants' perceived gap between usefulness and future importance. The surveys were administered on paper for the prestudy (N=61) and electronically for the main study (N=89). The response rate was: applications manager (10.7%), business analyst (11.3%), business user (2.4%), department manager (7.7%), developer/designer (43.5%), project manager (11.3%), technology architect (5.4%) and other (7.7%).

As we can see in Table 8.3 generally the prestudy teams perceive requirements and user artifacts as most useful while perceiving the biggest gap in usefulness for design documents. Main study project participants perceive design and project management documentation as currently most useful. The categories with the biggest gap between current usefulness and future importance were: requirements and design documentation. Project management artifacts had the least perceived future importance for all projects. The medium and large projects perceived documentation generally more useful. Interestingly the smaller teams perceived almost no gap for requirement documentation while it is the biggest gap for the large teams. Following our handover phases we can see that while the participants were relatively satisfied with these artifacts now, they are concerned about their application in the future.

8.5 Discussion

In this section, we (1) discuss the patterns of action which have been discovered as part of the thesis work (see 8.5.1). Moreover, we elaborate (2) the team members expressed satisfaction with the usefulness of artifacts, and (3) we point out artifacts which have been reported to support an agile project handover (see 8.5.2).

Table 8.3: Perceived usefulness of artifact categories (**min** & **max**) and variance (σ^2)

		Small size and short term projects (A-C)			Medium and large size projects (D-G)		
		Current Usefulness	**Gap**	**Future Importance**	**Current Usefulness**	**Gap**	**Future Importance**
Project Management	x	**2.30**	**0.04**	**2.34**	**3.29**	**0.34**	**3.62**
	σ^2	*(0.90)*	-	*(0.83)*	*(0.68)*	-	*(0.95)*
Requirements	x	**3.31**	**0.005**	**3.27**	**3.05**	**1.14**	**4.19**
	σ^2	*(0.84)*	-	*(0.89)*	*(0.85)*	-	*(1.68)*
Design	x	**2.78**	**0.65**	**3.58**	**3.34**	**0.81**	**4.14**
	σ^2	*(0.81)*	-	*(1.06)*	*(0.78)*	-	*(1.46)*
QA	x				**3.24**	**0.45**	**3.69**
	σ^2				*(0.37)*	-	*(0.98)*
User	x	**3.08**	**0.38**	**3.46**	**3.24**	**0.39**	**3.62**
	σ^2	*(0.83)*	-	*(0.83)*	*(0.45)*	-	*(0.69)*

8.5.1 Which Patterns of Action are Applied During Handover?

Below we answer the subquestion RQ2d: *"Which patterns of action do agile project teams apply during software project handover, maintenance and continuous development?"*

Throughout the study, we found the reappearing pattern of software engineers reestablishing the mental model of the original developers while learning about the Environment, System, and Architecture. Following the model by Nonaka and Takeuchi (1995) we recognized all modes of the knowledge creation model reappearing during the adoption: socialization (e.g., contact to the original team), externalization (e.g., documenting design), internalization (e.g., reading the application development handbook) and combination (e.g., capturing known issues and manipulating them through an online system). This means that in all cases the teams based their adoption on the delivered documentation as well as on direct communication. In cases where documentation was not sufficient, teams had to request additional artifacts. In cases where the original team could not be reached and not sufficient documentation was available to reestablish the shared understanding, a design had to be reengineered or even redesigned.

Our data on the perceived usefulness of artifacts shows a tendency towards the differences per project being bigger than those per artifact category. This implies that the adoption process has a big impact on the artifacts applied and vice versa. It is in line with earlier findings on organizational routines, the repeated patterns of team performance, these point out that the application of artifacts alone does not necessarily lead to a desired behavior (Stettina et al., 2012; Pentland & Feldman, 2008). When a process is not aligned with applied artifacts, team members are unlikely to perceive the artifacts as useful. In the observed projects we found the following factors affecting the patterns of action and the usage of artifacts: (1) Project size, (2) Time pressure, (3) Commitment to creation of artifacts.

The first directly affecting the adoption process was the project size. With growing com-

plexity of the system and the number of individuals involved, the maintenance of a shared vision and artifacts supporting it becomes more difficult. The second affecting factor was time pressure. *"Time pressure had the biggest impact on written documentation. It was not delivered on time, because sometime a sprint had to go live. Also, documentation was not always in sync with what was developed"*, says an application manager of project G. The third reappearing factor affecting the adoption success in our data was the perception that handover documentation was created for someone else. *"Docs are mainly something for other people. I don't really use them myself."*, states the product owner of project E. The functional application manager from project D comments, *"It is useful when I get sick, so that my backup knows how the system works"*. In an earlier experiment (Stettina et al., 2012) we pointed out that team members did not like writing documentation as they saw no feedback or other value added next to source code as the main artifact. If there is no clear purpose creating artifacts just for the sake of documenting this is is a waste of monetary resources as well as it hampers team morale.

8.5.2 Which Artifacts are Useful?

Below we answer the subquestion RQ2e: *"Which documentation artifacts do the teams perceive useful during the process of project handover, maintenance and continuous development?"*

Asked about the usefulness of artifacts project manager for project G says: *"Very useful, but you have to deliver what they need, in our case it was the flow document."* Our data shows here that different artifacts are useful at different phases of the adoption because each adoption phase has its own challenges.

In our analysis we assume that the environment is challenged by tacit knowledge of work arounds, dependencies, known issues and application of the system in practice. Similarly to Karlström and Runeson (2006), we observe that informal documents were created to support the handover. It indicates that the existing lexicon is being expanded to create a shared understanding. Hoda et al. (2010) describe the use of a project dictionary to bridge the gap between development and customers. Such can be aided with a software development handbook, or a wiki, and artifacts with shared ownership and tools that are easy to manipulate by the involved parties.

During the orientation within the system and architecture to be maintained, the design artifacts (code, UML models) are generally perceived as useful. However, in many cases those are not up-to-date or not transferred at all. Similarly to the findings by de Souza et al. (2005) we find that the source code and comments are perceived most useful and used most by software maintainers. Adoption in our data, however, was especially difficult for the larger projects, which indicates that with growing complexity of the system the complexity of the mental models to be acquired grows. This creates pressure for the teams assigned to

maintenance and continuous development teams.

8.5.3 Validity Considerations

Considering the ease of use and the findings that scales with two, three, or four response categories yield least reliable scores (Preston, 2000), we have decided on using a 5-point, unipolar scale. To reduce bias we encouraged the respondents to provide their honest opinions by emphasizing the anonymous treatment of data.

8.6 Chapter Conclusions

In this chapter we discuss the patterns of action teams perform during a project handover and the artifacts they perceive as useful. As a partial answer to RQ2: "*What are the components of governance necessary to understand knowledge worker project organizations?*", we discuss the relation of routines and artifacts as components of governance. As a partial answer to RQ3: "*How can governance address the dynamics of knowledge work across multiple knowledge worker teams?*", we would like to point out the context sensitivity of projects and unpredictability of knowledge, requires an iterative governance routine.

In this chapter, we present an in-depth analysis of project handover practices in software teams. Our combination of qualitative and quantitative data analysis allowed us to shed light on process-centric aspects of the adoption practices as well as the perceptions on the usefulness of the applied artifacts. Our findings stem from a representative data set of 30 teams in seven small to large size projects, covering observations and opinions of developers, application management, project management and maintenance staff.

The empirical data presented here emphasizes the importance of the actual handover practices applied, regardless if an agile software development approach is followed or not. An example of such a practice is the inclusion of maintenance staff in course of the development. In this study we encountered projects where participants felt they have too much (project F) and too little documentation (project D). Following the experiences from our earlier studies (Stettina & Heijstek, 2011b; Stettina et al., 2012) we argue that project members perceiving a deficit of documentation are exposed to the complexity in building an inter-team understanding, a shared mental model of the software and surroundings to be adopted. Due to the complexity and volatility of software systems trying to provide means to establish such an understanding by explicit documentation only can be uneconomic or even impossible at times (cf. Carlile (2002)).

We found a reappearing pattern of the project adoption as a learning process in three orientation phases: *(1) Environment, (2) System, and (3) Architecture*. These are highly inter-dependent, however, one can expect that each team member has to go through these phases in order to contribute efficiently to a project. Our data suggests that iterative steps covering

all modes of Nonaka and Takeuchi's knowledge conversion model (Nonaka & Takeuchi, 1995) have a positive impact on the adoption process. This means that for an effective or agile adoption, teams used documentation as well as direct communication. If either one was missing the process was slowed down. Our findings show that different artifacts are useful at different stages. (1) Learning about the *Environment* is guided by user documentation, known issues and work arounds, and challenged by making specific system knowledge explicit and maintaining it. Learning about the (2) *System* and (3) *Architecture* is guided by design artifacts, design decisions, and challenged by the fact that the artifacts are not always up-to-date or not available. Artifacts should be covering all these phases rather than just one in a concentrated manner (e.g., design). By doing so one establishes a reference guide that can help guiding both parties through the process. Within the projects we found especially project size, time pressure, and commitment to creation of artifacts affecting the handover and the patterns of action the teams would follow. Involving maintenance staff in the development process significantly helped improving the handover.

In this chapter we would like to put an emphasis on project adoption rather than handover as we believe it is important to consider a broader view than that of the pure handover and project closure. The difficulty lies in the fact that design documentation is difficult to maintain and that issues to arise for the users and QA are difficult to foresee. Time invested in creating one artifact thus might be lost while the issue appears somewhere else. We believe that the initiator, the party owning the project, should be pro-active in this process as being dependent on the project results. The initiator should be a part of the process and aware of the requirements regarding allocation of knowledge and deliverables within the own organization.

9

Routines and Boundary Objects in Achieving a Sustainable Practice

In this chapter we use two experiments to highlight the importance of routines their interplay with artifacts and their impact on the sustainability of a practice. Moreover, the importance of routines is presented by two experiments.

We address the research question RQ2 ("*What are the components of governance necessary to understand knowledge worker project organizations?*"). In order to make RQ2 suitable for our field research, we formulated the following subquestions RQ2f: *"How do externalization formalisms influence documentation practices in agile teams?"* and RQ2g: *"What are the implications of iteratively updated internal documentation on the quality of artifacts, perceived amount of work and project satisfaction?"*

This chapter is based on the following publication[1]:

Stettina, C. J., Heijstek, W., & Fægri, T. E. (2012). **Documentation work in agile teams: the role of documentation formalism in achieving a sustainable practice.** In Agile Conference (AGILE), 2012 (pp. 31-40). IEEE. [1]

[1]The author would like to thank IEEE and his co-author for permission to reuse relevant parts of the article in this thesis.

9.1 Documentation: Required or Requested?

When compared to document driven traditional software development, agile software development practices advocate an emphasis of direct communication (Abrahamsson et al., 2003; Dybå & Dingsøyr, 2008a; Clear, 2003). This focus is embedded in reoccurring practices such as iterative development, frequent customer involvement, daily stand-up meetings or team-based effort estimation (Fægri, 2010). During the course of a development project such routines make work more predictable, support the transfer of knowledge and ensure communication within the team. While knowledge sharing practices based on socialization of team members enable agility and common understanding at the team level (Melnik & Maurer, 2004) they can also result in problems such as gaps of undocumented knowledge and lacking architectural integrity between projects. This is particularly problematic if people leave the team or organization (Abrahamsson et al., 2003; Ramesh et al., 2007; Stettina & Heijstek, 2011b). In this sense, agile development may make project management and program level management more difficult.

Efficient teamwork in agile development relies fundamentally on shared mental models (Moe et al., 2010). Artifacts in agile development lose much of their effectiveness if participants lack a shared, overall understanding of project objectives. Sharp et al. (2009) point out that the artifacts largely lack detailed information about the application under development. Non-functional requirements, for example, are difficult to link to user stories and tend to be ignored or ill-defined (Ramesh et al., 2007). Code comments are generally accessible to technical staff only. Contrarily, in highly regulated and contract-driven environments formal documentation is a must. In European research projects, for example, project partners are encouraged to use documentation deliverables to disseminate their knowledge and to stimulate collaboration among the participating projects.

Software practitioners tend to perceive writing documentation as a burdensome side-task and literature suggests that this results in deliverables usually compiled and submitted at the end of a project (Clear, 2003). Software projects are executed under strict constraints of time and budget. Team members are often needed for new projects and committed to these projects early on. There is rarely enough time to document all experiences and knowledge after the project delivery. Small teams might perceive codification of their knowledge less important, according to earlier findings (Stettina & Heijstek, 2011b) and according to our own experience this can lead to gaps in knowledge. As knowledge is not transferred seamlessly these gaps can hinder the agility of an organization as a whole.

To improve our understanding of knowledge codification strategies in software development we designed and executed a quasi-experiment in which students were required to iteratively develop a small-to-medium size application in teams. The 28 students were divided into 8 teams and two groups: SAD and UML. Group SAD (Teams A-D) was to update and deliver their high-level software architecture in form of a textual description defined by RUP

templates. Group UML (Teams E-H) was instructed to update and deliver their low-level software design in form of UML models. In this chapter, we report on this study into the team's practices to keep documentation up-to-date.

9.2 Related Work

In their seminal theory of knowledge creation, Nonaka and Takeuchi (1995) explain knowledge creation as resting in conversions between explicit and tacit forms - and between the ontological levels of individuals and groups. Nonaka and Takeuchi identify four modes of conversion in which new knowledge emerges; socialization, externalization, combination and internalization. A key point in their argument is that innovation (a result of knowledge creation) must be enabled by encouraging and facilitating the occurrence of all conversion modes in daily practices within the organization. Because software development is fundamentally driven by the creation of new knowledge we believe that Nonaka and Takeuchi's model is a relevant conceptual model to gain understanding of knowledge processes in agile methods.

Agile methods derive much of their agility by giving preference to social interaction among people rather than documentation. The socialization mode of Nonaka and Takeuchi is therefore more prominent. Agile methods encourage knowledge creation through practices of frequent interaction among the the team members (Melnik & Maurer, 2004). Consequently, however, agile methods may demand a more 'social attitude' among team members compared to plan-driven methods (Nerur, Mahapatra & Mangalaraj, 2005). Furthermore, socialization incurs inherent scalability limitations. Socialization works well for small teams or teams that can be synchronized via frequent meetings (Laanti, 2008) but for teams whose members are spatially separated or knowledge must be passed on beyond the team members, the organization is often forced to rely on explicit forms of knowledge (see Figure 9.1). In this chapter we refer to the knowledge recipient at the program level - in contrast to the project level, which refers to the project team.

Few empirical studies have examined practices of inter-project knowledge sharing and externalization in agile development, and most work is focused on requirements engineering (Ramesh et al., 2007) and software process improvement (Dingsøyr & Hanssen, 2002). Laanti (2008) proposes that knowledge sharing in Scrum can be up-scaled to the program level by means of two nested control loops consisting of a team level backlog and a program level backlog. This way, next to the common Scrum daily and sprint meetings, the program level would include a *program Scrum* once or twice a week to synchronize the program level with the team level. Postmortem reviews have been successfully applied in agile projects to externalize and transfer project experiences enabling software process improvement (Dingsøyr & Hanssen, 2002). Sharp et al. (2009) point out that agile artifacts such as user stories and project wall work well supporting the project process but largely lack detailed information about the application under development.

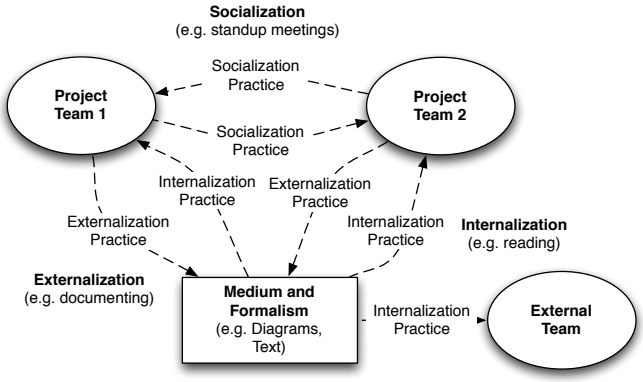


Figure 9.1: Problem domain: Model of knowledge transfer through agile practices

Current studies on the use of documentation during software development are few and present mixed results, studies on documentation in agile settings are even fewer (Dybå & Dingsøyr, 2008a). Stettina and Heijstek (2011b) found that agile practitioners in their data set perceive documentation important or even very important but that too little documentation is available in their projects, and indicate that knowledge is lost for older projects. Clear (2003) points at the behavior of students and observes documentation being seen as a burdening by-product and being hurriedly pieced together at project end. Past studies suggest that documentation is frequently out of date and should be rather seen as a communication medium than an accurate or up-to-date artifact (Forward & Lethbridge, 2002).

Based in the present state of the art we find it appropriate to explore in more depth how documentation work is carried out in agile teams, and in particular the relationship between documentation formalism and developer practice. Our guiding hypothesis is that documentation formalism influences agile practices (compare Figure 9.1). We thus pose the following two sub research questions:

1. *RQ2f: How do externalization formalisms influence documentation practices in agile teams?*

2. *RQ2g: What are the implications of iteratively updated internal documentation on the quality of artifacts, perceived amount of work and project satisfaction?*

9.3 Method

We applied a mixed methods approach to collect data (Miles & Huberman, 1994). Qualitative process descriptions were compared and contrasted with quantitative questionnaires and

the artifacts developed in course of the project. Ecouraged by agile methods' emphasis on self-managing teams and the integration of individuals into the software development process (Stettina & Heijstek, 2011a), we sampled individual team members' perceptions using anonymized questionaires. To create a deeper picture supporting our research we further used informal interviews and conducted a document inspection on the created artifacts.

Collecting practices of writing documentation as human behavior requires both qualitative data and observations in a real world context. Following Langley's (Langley, 1999) framework for building theory from process data we have selected visual mapping which requires at least five cases in moderate level of detail to begin a pattern identification. Using graphical forms allows the presentation of large amount of information in little space and is a useful tool to develop and verify ideas in theory development (Langley, 1999).

9.4 Research Design

To address the research questions we embedded the study within a software engineering project course at Leiden University with a class of 28 students divided into 8 groups. This provides a suitable environment allowing a comparison of several teams working on the same project.

The project was designed as a fourteen week quasi-experiment including seven development iterations with the aim to develop a program being able to recognize and extract UML class diagrams stored in bitmap images. Guided by Richards (2009) we devised project-based courses that encouraged students to engage in practical work, form their own groups and stimulated students' individual reflections on ongoing accomplishments. While we had given recommendations, the students were given free choice of development platform and libraries. The coaching took place after the weekly software engineering lectures in nine sessions. In each session we discussed the progress of the teams, the challenges and the next steps. For instruction we used handouts in class explaining the course of the project and actions to be taken.

Students were instructed to document their development efforts focusing on architectural aspects (Lethbridge, 2000). As we wanted to study the implications of documentation formalisms we have chosen two of the most common artifacts to be updated: (1) high-level software architecture in form of a textual description and (2) low-level software design in form of UML models. We explicitly did not incorporate requirement descriptions in our study as those are already addressed by agile artifacts (Sharp et al., 2009).

Due to the lectures being originally aligned to the Rational Unified Process, we decided to use RUP templates for the textual deliverables: Software Development Plan, Software Requirements Specification, Configuration Management Plan and the Software Architecture Document. One reason to choose RUP templates was that in many established software companies such are seen as standard and are expected to be in use even while applying agile methods. We emphasized throughout the project that the formal templates should be seen

Table 9.1: Project course

Project Planning and Initial Design
08-09-2011: (Session 1) Introduction
15-09-2011: (Session 2) Requirements Elicitation Session
22-09-2011: (Session 3) Requirements Elicitation Session 2
29-09-2011: (Session 4) Requirements Review Presentation
06-10-2011: (Session 5) Architecture and Design Presentation
Development
13-10-2011: Sprint 1 - User Interface
20-10-2011: Sprint 2 - Integrate shape recognition and OCR libraries
27-10-2011: Sprint 3 - Recognize outer UML shapes (Session 6)
03-11-2011: Sprint 4 - Integrate shape and character recognition
10-11-2011: Sprint 5 - Recognize relationships
17-11-2011: Sprint 6 - Optimization
24-11-2011: Sprint 7 - Additional Sprint (Session 7)
Delivery and Postmortem
01-12-2011: (Session 8) System Demonstration and Handover
07-12-2011: (Session 9) Project Closing: Discussion & Postmortem Review

as guidelines rather than as stiff documents to be filled. The UML models to be delivered consisted of a class diagram, a sequence diagram, an activity diagram and a state chart diagram.

After the definition of an initial architecture in session five the student teams were divided into two groups: Teams A-D were required to update and deliver the low-level software design models with every sprint. Teams E-H were required to update and deliver the software architecture document with every sprint. From now on we will refer to those teams as the UML and the SAD groups. The students were graded in the following schema: 25% Requirements Engineering, 25% Software Design, 25% Implementation and 25% overall quality. The updated artifacts have been made an explicit "customer requirement" the students would be graded for.

In order to test how the students perceive the importance of documentation artifacts after project delivery we chose a setup similar to the one used by Smith, Mann and Buissink-Smith (2001) where at the end of the development phase the student groups would swap their project source code repositories and documentation to simulate a maintenance environment. The project is hosted at Google Code and is freely accessible[2].

9.4.1 Qualitative Data Sources

To address the research question we first have to clarify our research lens on human action. As patterns of action we understand those as defined by Cohen et al. (1996) as (i) Recurring (they can take place at different times, or involve different actors), (ii) selectable (meaning that there are forces that make them more or less likely to happen) and (iii) set in an organizational

[2] `http://code.google.com/p/image-to-uml/`

context (the actions are not those of an isolated individual).

In order to collect patterns of action after system delivery we asked the student groups to discuss and draw the process of documenting their software as UML activity diagrams on a paper and to present them one after another at the board during the closing meeting (Session 9). We chose to use activity diagrams as we assumed those to be most accessible to software engineers. At the end of the session we took pictures of the models on the board and collected those drawn on paper earlier. We also kept an audio recording of the session for reference. We conducted direct observations only during the sessions in class. However, as the artifacts were mainly produced outside the classroom we conducted informal interviews and asked the students for a better level of detail or for clarification regarding certain choices or team member roles (*"How did your team assign the work related to documentation?"*, *"Why did you choose to work on documentation at that particular stage?"*). During the final session we also conducted a postmortem review (Dingsøyr & Hanssen, 2002) of the project to identify good and bad experiences of the development teams. This final session lasted for about three hours. At the end of the project the created documents were collected from the repository and assessed according to their size in word count and quality. We analyzed the content of the deliverables for each group according to the structure of the templates and readability.

9.4.2 Quantitative Data Sources

We had access to the following quantitative data sources: (1) the longitudinal survey answers collected each week in class, (2) longer questionnaires conducted in each development phase and the (3) online repository. First, a short questionnaire for a longitudinal collection of the individual team member's satisfaction with the project, the amount of work and the documentation was used. This part of the questionnaire was collected every week during the project. The objective was to measure how team member perception on work environment and documentation changes during course of the challenging project, similarly to the project satisfaction graph as earlier presented by Moe et al. (2010). The questions were: *"How satisfied are you with the project?"*, *"How satisfied are you with the amount of work?"*, *"How satisfied are you with the teamwork in your team?"* and *"How do you feel about documentation in your project?"*. Second, longer questionnaires were applied in the course of the development phase, at project delivery and after project handover to maintenance. The longer questionnaires were used to collect perceptions on the usefulness of specific artifacts, the distribution of roles within the groups and gave students the opportunity to anonymously comment on their projects. Sample questions were: *"What was your predominant role in the project?"*, *"How useful did you find documentation for your project?"*, *"How useful did you find updating and delivering documentation with every milestone?"*, *"What influence did the requirement of updating documentation have on your project satisfaction?"* and *"If you would know that after development the project will be maintained by another team, which documentation artifacts*

would you choose to keep up-to-date?".

9.5 Results

In this section we will proceed to address the two research questions. The project started on September 8, 2011 with an introductory session and proceeded as outlined in Table 9.1.

9.5.1 Patterns of Team Practice

Figure 9.3 visualizes each team's practice to create and update their internal documentation as emerged from observations, the final session in class, the informal interviews and the online repository. Actions marked dark gray were assigned to a single member of the team throughout the whole project, while actions marked light gray were rotated.

The project was somewhat challenging in terms of programming skills for the second year students as some of them expressed (e.g. *"Teamwork would be so much better if everyone was a skillful programmer..."*) and the majority of teams established specialization among their team members. While the initial version of the artifacts has been developed commonly, five of the eight teams (teams A,D,F,G, and H) defined a single team member to do the updates. because *"it was most convenient"* i.e. because that particular member was least good in coding or his own preference to work on documentation. As we can see two teams established a review process. In three of the teams the appointed member used SVN logs to update documentation.An exception here were team B and E as, we can recognize in Figure 9.3. In team B every developer was in charge of bringing the documentation up-to-date with his changes in code. The team had a skillful programmer leading the project and employed pair-programming. Team E kept track of hours spent on the project (keeping track of hours was a recommendation given to the students during the sessions), the person who had the least effort spent before end of the iteration had to update the documents. For team C updating documentation was not really an issue. After creating the initial documents the team was struggling to get the implementation to work, thus changes were not necessary. As emerged from the survey, Figure 9.2 represents the team member roles of the participants as specified. Multiple answers were possible and out of the three categories, "Code", "Documentation" and "Administration" the majority of the participants (63%) specified their predominant role as "Code" only, 11% as Documentation and 0% specified their role as purely administrative. When clustering the categories in Figure 9.2 we can see that 78% of the participants were involved in coding (coding, coding & documentation and administration & coding) and 30% were involved in documentation. Interestingly, as we can see in Figure 9.2 is that only 8% of the participants defined to be involved in coding and documentation.

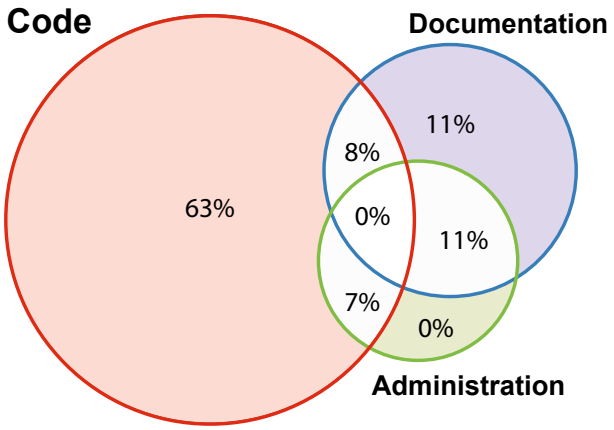


Figure 9.2: Distribution of team roles. Total number of participants involved in coding: 78%, in documenting: 30% and in administration: 18%

Table 9.2: Summary of team practices to write and maintain documentation

Team A	All members worked on initial versions, thereafter one team member was in charge of maintaining documents, "according to repository and team member comments"
Team B	Every developer was in charge of updating documentation with changes in code. Difficult parts of the program were developed in pair programming.
Team C	Updating models was not really an issue. Most of the architecture was done in UML models, and as the team was struggling to implement the initial design there was not really much to change.
Team D	All team members but one wrote the initial documents, the remaining team member reviewed them now and then.
Team E	Kept track of hours, the person "who had time" updated the artifacts. After asking the team how many were involved, a team member answered "basically all".
Team F	Initial documents were created by all team members, updates were done by team member TM1.
Team G	While one team member was in charge of updating the textual SAD documentation two others were in charge of UML models
Team H	Initial by all, updates were committed by a single team member because he was the "least good in coding"

9.5.2 Project and Teamwork Satisfaction

The students generally welcomed the project while finding it challenging with comments varying from *"It will be tough getting everything to work on the deadline"* to *"Very nice*

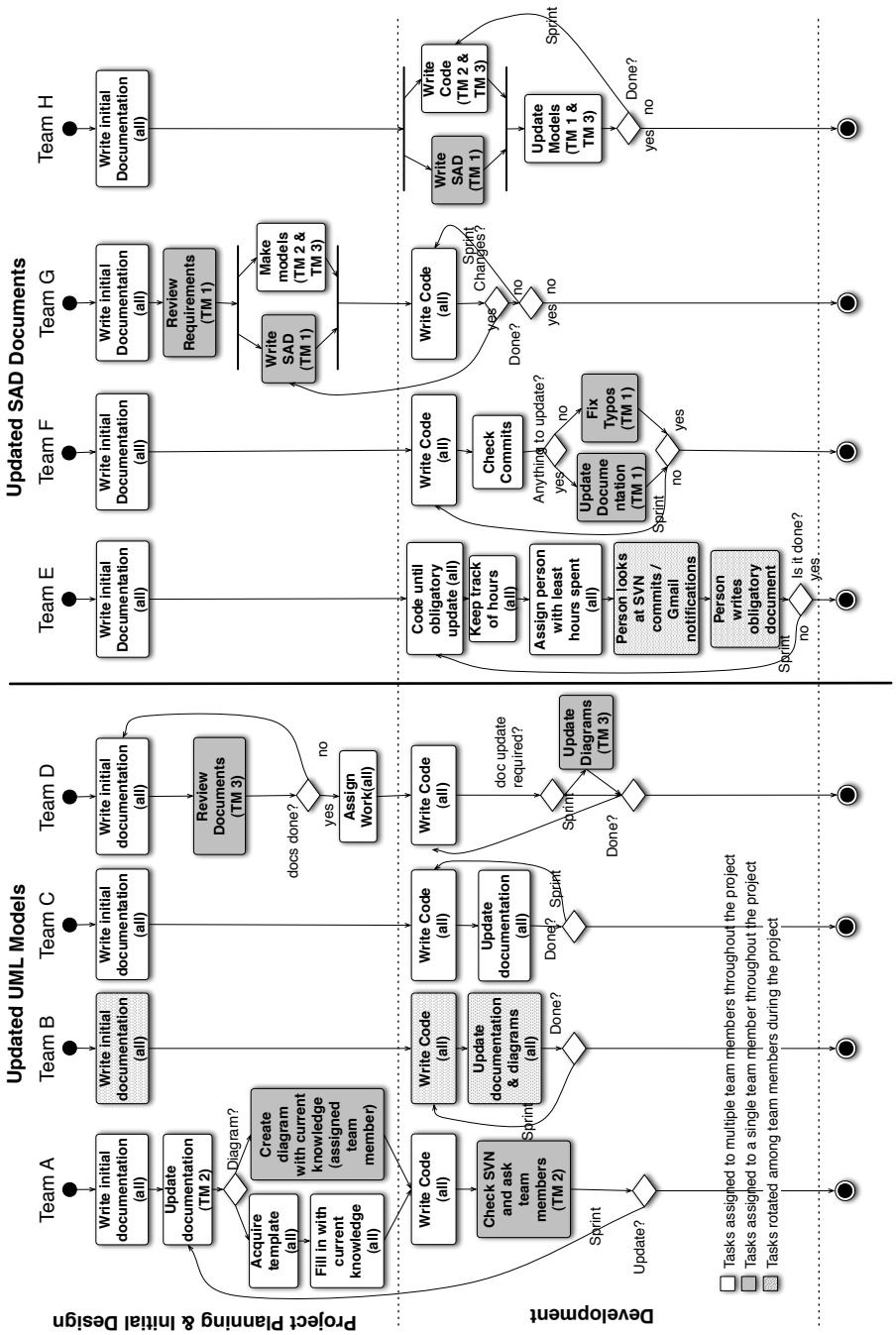


Figure 9.3: Team practices to update documentation

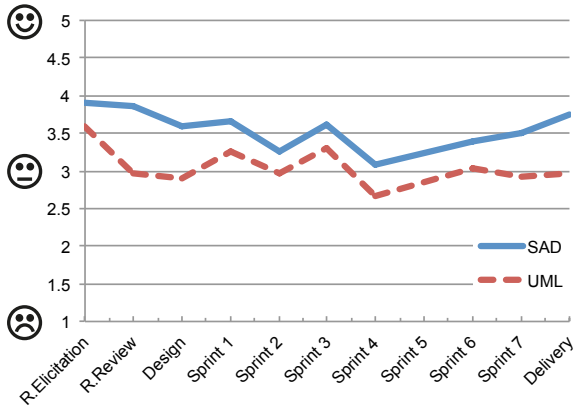


Figure 9.4: Project satisfaction. [Scale: 1=Not at all satisfied, 2=Slightly satisfied, 3=Moderately satisfied, 4=Very satisfied, 5=Extremely satisfied]

project. Challenging, yet very interesting.". During the requirements review session there was quite some uncertainty due to the free choice of implementation platform. During the implementation phase the project remained challenging, and at the end of the first iteration a student commented: *"Using the internet, we have made good progress. However, because the code is from mixed sources and adapted to our situation, the code is a mess. The documentation isn't up to date either, though I do think we are all on the same level."* During the feedback session one student remarks that after integrating the OCR and shape recognition libraries many discovered how much work it really was.

Figures 9.4 and 9.5 illustrate the satisfaction with the project and the amount of work for both customer groups during the course of the project. In Figure 9.4 we can observe that the curves for satisfaction with the project run parallel, with the UML groups being generally less satisfied with. Figure 9.5 illustrates the satisfaction with the amount of work for the SAD and UML groups. The teams perceived updating and delivering documentation as not very useful. When asked on how much influence documentation work would have on their project satisfaction, they said it would have only slight influence (compare Fig. 9.7).

Teams perceived documentation as a burden, as work 'that needs to be done'. Especially for the SAD group we can observe an increase in perceived amount of documentation in the period between Sprint 3 and 7 of the development phase. This increase of perceived documentation amount as visualized in Figure 9.6 happens at the same time as the team members perceive an increased amount of work (compare Figure 9.5). Interesting is to note that when asked which documents the teams would keep up to date if the project is to be handed over, majority of the participants chose software architecture document, software design models and software requirements specification out of the five deliverables.

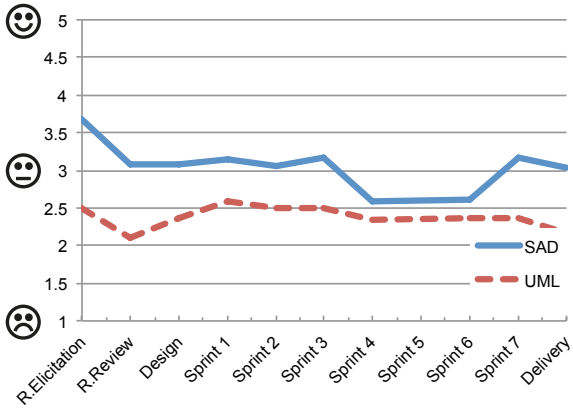


Figure 9.5: Satisfaction with the amount of work. [Scale: 1=Not at all satisfied, 2=Slightly satisfied, 3=Moderately satisfied, 4=Very satisfied, 5=Extremely satisfied]

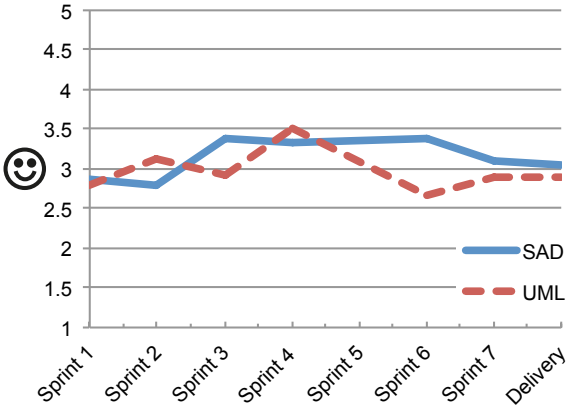


Figure 9.6: Perceptions on internal documentation in course of the implementation phase. [Scale: 1=Way too little, 2=Slightly too little, 3=Just about right, 4=Slightly too much, 5=Way too much]

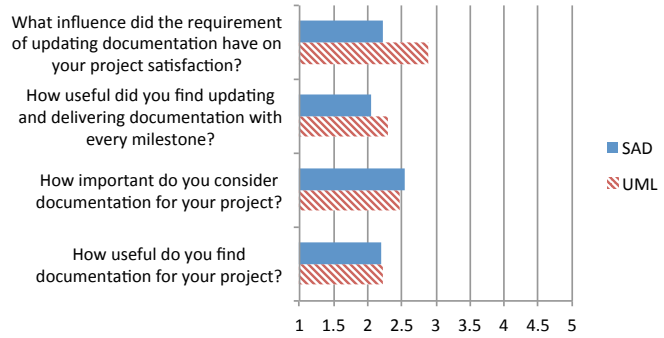


Figure 9.7: Perceptions after project delivery. [Scale: 1=Not at all, 2=Slightly, 3=Moderately, 4=Very, 5=Extremely]

9.5.3 Quality of Documentation

After completion of the project on December 7 we downloaded the latest versions of the deliverables and conducted a document analysis. Table 9.3 illustrates the results of our analysis. As we can see in the table the teams instructed to update their architecture document with every milestone generally produced larger and more elaborate documents. The quality of the artifacts delivered by the SAD groups was significantly better. Teams that were instructed to update their UML models did so, however as one could expect, the changes within the models were less visible. As they did not had to update other artifacts than the models, the quality of the documents delivered was lower significantly, for example the documents contained template explanations still. Table 9.3 also contains the assessment of the implementation, which indicates that there is no significant correlation of the updated documentation on the quality of the implementation. The software development plan was largely omitted by the teams which made it the documentation artifacts with the lowest quality delivered.

9.6 Discussion

In this section, we discuss the emergent patterns of action, the team members' expressed satisfaction and the quality of documentation deliverables. We will first consider the role of documentation as 'product', e.g. as tangible project deliverables. Second, we will consider the role of documentation as a medium of knowledge exchange and how documentation supports sharing and leveraging project knowledge at the organization's program level.

Table 9.3: Descriptive variables and team results (lowest & **highest** values)

		Updated UML Models				Updated SAD Documents			
	project team	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**
	persons in team	4	4	3	4	4	3	4	3
	environment	C++	C++/PHP	C++	C++	C++	C#	C++	C#
	implem. quality	●○○○	●●●●	●●○○	●●○○	●○○○	●●●●	●●○○	●●●●
wordcount	*SAD*	1051	670	508	670	1601	1036	1123	**2296**
	SRS	1238	850	1347	326	1116	2167	1757	**2393**
	CMP	757	377	700	578	1143	717	-	**1747**
	SDP	-	**1692**	648	602	-	912	-	-
quality	*SAD*	●○○○	●○○○	●○○○	●○○○	●●○○	●○○○	●●○○	●●●○
	SRS	●●○○	●○○○	●●○○	●○○○	●○○○	●●●○	●●●○	●●●●
	CMP	●○○○	●○○○	●○○○	●○○○	●●●○	●●○○	-	●●●○
	SDP	-	●●○○	-	●○○○	-	●○○○	-	-
	class diagram coupling	low	mid-high	low	low	mid-low	mid-high	mid-low	mid-high
	activity diagram complexity	mid	low	mid	mid	high	mid	mid	mid-high
	documentation participation	single	all, on changes	single	single	all, on effort	single	single	single

9.6.1 Documentation as a Product: Iterations Produce Better Textual Documentation

The qualitative comparison of the deliverables in Table 9.3 illustrates that textual documentation benefits from being build up iteratively. The benefits of iterative practices on diagrammatic documentation are less visible in our data. While the updates of UML models, as a more formal medium, were less visible, the updates of the teams assigned to update the less formal textual architecture documents produced more extensive artifacts.

The visualization of actual patterns of actions within the different teams (see Figure 9.3) shows how feedback resulting from multiple forms of knowledge conversion (Nonaka & Takeuchi, 1995) is embedded into the documentation. Hence, emergence of textual documentation artifacts can be clearly linked to the iterative development process. This may help to improve transparency towards the quality management of documentation deliverables (or 'documentation products'.)

However, one has to consider another aspect: In our designed experiment, team members did not perceive documentation as contributing value to their projects. Team members referred to it as *"a task that needs to be done."* Motivation to participate in an activity increases with the perception of importance and influence of the activity's results (March & Simon, 1993). Therefore, we find it reasonable to suggest that writing documentation becomes more attractive if someone, and preferrably an acknowledged stakeholder, has expressed an interest in the results. A comment of a Dutch Scrum coach in a recently conducted interview pinpoints the

issue: *'I always try to explain, "You define what amount of documentation you want, I am happy to provide you with all kinds of documentation, but I need someone who is happy with what I deliver"'*. Clients should acknowledge for the team that documentation is work of equal importance as other tasks, and their requirements towards documentation should be as clear as other requirements. In short, documentation should be considered a valuable product.

9.6.2 Documentation as a Medium: Creating Artifacts While Gambling With Collaboration

SAD teams expressed an increase in the amount of work on documentation tasks lasting throughout the development phase. This corresponds with their perception of an increasing amount of documentation being available. Among the UML teams the perception of the amount of work stayed more constant during the project and the UML teams perceived only a short peak in the amount of documentation.

One can argue that the SAD teams perceived a surplus of available knowledge during development as the team members work tightly together. This would be consistent with our earlier findings where a team member commented: *"During project development any needed documentation is generally available. However, finding documentation for older projects is not always easy, and sometimes this documentation is missing."* (Stettina & Heijstek, 2011b). However, one can also interpret this increase as a distraction that the teams perceived from the documentation task. This interpretation is supported by the perception of increasing amount of work. It suggests that updating UML models was considered less intrusive, and less demanding than updating textual documentation.

Our data shows that the majority of teams implemented strict roles dividing coding and documentation work. Most teams appointed documentation work to a single team member, most frequently the least qualified programmer. While all in all 78% of the participants were involved in coding, 30% worked on documentation and only 8% answered to have worked on both (see Figure 9.2). This is very far from the 'agile ideal' where team members are generalists and avoid specialized roles. The circles would then have much greater overlap. Agile team members do not have specific roles, instead each team member has a variety of roles. This encourages socialization within the team and is an important enabler for knowledge creation (Nonaka & Takeuchi, 1995).

An interesting aspect we discovered in the data was a strong tendency towards increasing role enforcement throughout the duration of the project. In the initial phases all team members were involved in writing documentation but gradually the 'documentation role' became the province of a single team member. We speculate that time pressure is an important explanation for this tendency. Time pressure has been found to have an impact on the choice and maintenance of team routines. Even if an inadequacy has been indicated beforehand, time pressure increases the likelihood to choose an established routine (Becker, 2004). Although perceived

as highly important, time pressure and cognitive demands of documentation tasks impede sharing of documentation work. Explicit rotation of roles and more effective documentation tools may create the necessary stimulus for sharing the documentation tasks (Fægri et al., 2010).

The analysis of the team practices in Fig. 9.3 and the distribution of roles in Fig. 9.2 illustrates that the joint effort of producing documentation faces other obstacles than the availability of documentation templates and stated documentation requirements. As we can see in Fig. 9.7, both groups similarly perceive the usefulness and importance of internal documentation. As one team member comments: *"Documentation was written because it was required, but because of the program's modularity, no documentation other than the original class diagram (and of course OpenCV, Tesseract [library] reference manuals) was necessary contributing to the project."*. While looking at the teams' documentation practices in Figure 9.3 one can recognize how the developed patterns of action, also called organizational routines (Becker, 2004), enable the feedback of team members and source code to be embedded in documentation. However, the circular knowledge creation processes are missing (Nonaka & Takeuchi, 1995). This suggests that increasing insights - stemming from documentation work - was not incorporated back into the source code.

Hence, we can see that the unbalance in documentation practices distracts from coding activities while not sufficiently adding value to the project. This led the team to perceive documentation as an intrusive task. As opposed to the incremental nature of development from agile practice, neither the formal documentation templates nor the UML models alone were very suited to support collaboration within the project teams. These observations contradict the studies of Sharp et al. (Sharp et al., 2009) which demonstrate how the notational attributes and social processes of agile artifacts such as user stories and the project wall applied in practice are mutually supportive for an agile team. For better externalization of development knowledge these reoccurring patterns of knowledge creation activity should be more balanced.

9.6.3 Agile Program and Portfolio Management: Usability and Importance of Knowledge Sharing

"There is one thing we did not implement, we did not use RUP", said team member of team B jokingly at the beginning of the system demonstration while summarizing which requirements have been met (Using RUP was not explicitly a requirement, but was widely covered in the lectures). In the short maintenance phase of the project (see project plan in Table 9.1) the teams reported not to have taken any substantial advantage of the produced documentation artifacts other than a short handover description that was to be prepared. However, team members do perceive that documentation is important for future development. When asked which documents the teams would keep up to date when the project was to be handed over, out of the five artifacts vast majority of the participants chose for software architecture document

(75%), software design models (71%) and software requirements specification (54%) to be kept up-to-date. This is also visible through the difference on perceived usefulness and importance as depicted in the two questions in Fig. 9.7. While the team members perceive it less useful within their teams now, they perceive that documentation is more important, thus indicating a tension between team and external stakeholder's interests.

As we can see in Fig. 9.7, the teams perceived updating UML models more useful and perceived it to have a bigger influence on their project satisfaction than updating SAD documents. In organizational science Carlile (2002) discussed documentation artifacts as types of boundary objects helping to bridge functional and organizational boundaries. Participants involved in cross-boundary knowledge exchange need to be aware of the personal costs and the necessity to transform own as well as influence knowledge in other domains. Effective boundary objects stimulating collaboration are a prerequisite to that. Taking our findings one can argue that the UML models are more effective boundary objects according to Carlile (2002) due to their visual representation and less effort necessary for alternation. This, however, is only true if they are created in a format easily adaptable to all participants. Software and file formats available to a single functional group only can hinder feedback of participants from other functional groups.

To conclude, for management of agile project programs and portfolios it is important to note that since team members perceive inter-project knowledge transfer as important, it is just that they oppose how the applied codification practices were implemented in their particular projects.

9.6.4 Recommendations for Research and Practice

In this chapter we have employed a new longitudinal approach to measure team and task satisfaction, based upon the project satisfaction graph as applied by Moe et al. (2010). The notation can help measuring the implications of events occurring during development on team satisfaction. Such events can be the dependency of team satisfaction on external factors. The two project satisfaction graphs (Fig. 9.4) running similarly indicate that both groups perceive a similar satisfaction in course of the project while the "humps" indicate the uncertainties encountered. We further discuss documentation as an intrusive task during development, we also present an approach how to quantify it, but the question to be solved yet is what non-intrusive documentation should look like.

We recommend that for *documentation as a product* (e.g., in regulated and contract-driven environments) it is recommended to (1) develop and deliver textual documentation iteratively by the use of easily accessible and adjustable artifacts as this will most likely improve the quality of deliverables and contribute to collaboration between people in different functional domains. In any case, however, (2) documentation needs to be communicated and accepted by the team as a proper product, product owners should motivate it as such and should be clear

on their requirements. Furthermore, (3) we argue that *documentation as a medium* (e.g. for sharing and leveraging of internal development knowledge) should be a non-intrusive task.

9.6.5 Validity Considerations

Although industrial teams are expected to have more experience than the chosen software engineering students, we argue that the challenging project environment with strict time pressure is very likely to provide an environment with similar implications on perceptions on documentation and project satisfaction. The extent to which graduate, undergraduate and doctoral students are representative of professional software developers and the threat of interaction of selection and treatment respectively, have been addressed in various other studies (e.g. (Briand, Labiche, Penta & Yan-Bondoc, 2005; Höst, Regnell & Wohlin, 2000)).

Considering the ease of use and the findings that scales with two, three, or four response categories yield least reliable scores (Preston, 2000), we have decided on using a 5-point, unipolar scale. To reduce bias we encouraged the respondents to provide their honest opinions by emphasizing the anonymous treatment of data. We re-iterated that the data collection had no implication on grading.

9.7 Chapter Conclusions

In this chapter, we presented an in-depth analysis of documentation practices in small software teams. Our combination of quantitative and qualitative data analysis allowed us to shed light on longitudinal and process-centric aspects of the documentation work carried out in the teams - covering a period of 14 weeks.

As an answer to RQ2f: *"How do externalization formalisms influence documentation practices in agile teams?"*, we may conclude that routines and boundary objects have a large impact on the sustainability and effectiveness of a routine. As an answer to RQ2g: *"What are the implications of iteratively updated internal documentation on the quality of artifacts, perceived amount of work and project satisfaction?"*, we may also conclude that iterations improve especially textual artifacts. Following our observations, as a partial answer to RQ2: "*What are the components of governance necessary to understand knowledge worker project organizations?*", we may conclude that (1) routines, and (2) boundary objects, as well as the concrete formalisms of these artifacts have an impact on the sustainability and effectiveness of a routine.

Dividing our analysis into the perspectives of *documentation as a product* and *documentation as a medium*, we found that primarily textual artifacts benefited from iterative documentation practice. Iterative practices thus not only help improving source code but can also improve the quality of documentation deliverables. We found that teams perceive documentation created for internal knowledge sharing as a distraction which causes them to choose

the least qualified programmer to update documentation. This perception of documentation as what we can call an *intrusive task* is a reason why developers in small teams don't like to write documentation - it distracts them from doing what they find most important: writing code. When developed for internal purposes one can conclude that the formal templates are able to capture the development knowledge. However, they seem not well suited to support collaboration within the team members as the majority of the teams chose a single team member to update the documents. This specialization is hampering collaboration in self-managing teams.

With respect to management practices for agile project programs and portfolios we conclude that the formal documentation was perceived as a burden by the teams as it was not contributing to the development of the software, however, knowledge sharing for future projects and maintenance was still perceived as important. Team members noted that if the project was to be handed over that they would keep software architecture document, design models and requirements specification up-to-date. This implies that the teams perceive knowledge sharing as important, however, they are unsatisfied how design knowledge has been documented.

10

Intensive Coaching and Team Routines

In this chapter we discuss the governance of knowledge worker team organizations based on continuous improvement of routines through intensive coaching.

We address RQ2 ("*What are the components of governance necessary to understand knowledge worker project organizations?*") and RQ3 ("*How can governance address the dynamics of knowledge work across multiple knowledge worker teams?*"). In order to make RQ2 and RQ3 suitable for our field research, we formulated the following subquestions RQ2h: *"What are the implications of individual intra-team stand-up meetings on coaching success and team satisfaction compared to bigger inter-team stand-up meetings?"* and RQ3a: *"How can we plan software engineering courses so that using agile process improvement techniques we can improve education and contribute to research at the same time?"*

This chapter is based on the following publication[1]:

Stettina, C. J., Zhou, Z., Back, T., & Katzy, B. (2013, May). **Academic education of software engineering practices: towards planning and improving capstone courses based upon intensive coaching and team routines**. In Software Engineering Education and Training (CSEE&T), 2013 IEEE 26th Conference on (pp. 169-178). IEEE.

[1]The author would like to thank IEEE and his co-authors for permission to reuse relevant parts of the article in this thesis.

10.1 Agile Practices, Routines and Project Skills

One of the biggest challenges in academic education of professional processes like software engineering is to balance students' practical education with academic reflection. Software engineering graduates on one hand need to get prepared for managing the engineering process. On the other hand, they are academic graduates who need to be trained in reflection and research activities.

Research suggests that agile methods stimulate learning and reflection (Stettina & Heijstek, 2011a). As such they are not only software engineering practices but can be applied in education as well. They have attracted educators and coaches delivering positive results in industrial (Silva & Doss, 2007; Padula, 2009; Paasivaara & Lassenius, 2011) and academic settings (Layman, Cornwell & Williams, 2006; Melnik & Maurer, 2005). Agile practices are concrete team-level routines such as frequent customer feedback loops, iterative delivery of intermediate results, and stand-up team coordination meetings. These routines further knowledge creation and faster feedback loops on organizational level. Such procedural knowledge is tacit (H. Taylor, 1999; Nonaka & Takeuchi, 1995) and therefore cannot readily be acquired through simple methods such as lectures or books. Agile routines, in professional practice as well as in academic education, thus need be trained through executing the process (H. Taylor, 1999). In order to be able to transfer routines educators and coaches thus themselves need to have collected sufficient experience executing the steps. This is equally true for software engineering researchers, who need experience with agile methods to design and execute meaningful studies.

In this chapter, we follow the call from literature (Hazzan & Dubinsky, 2007; Tierney & Holley, 2008) to advance academic teaching in software engineering by discussing our approach to continuous planning and improvement of software development capstone courses. Based upon intensive coaching and the notion of team routines we create a research-roadmap which enables professional education of students alongside with academic reflection. As a specific example we discuss a graduate course on *System Development and Project Management*. There, 30 students in 6 teams are part of an experiment while turning a created idea into a working demonstrator using agile practices. The chapter aims at helping academic educators in their creation of research-based software engineering courses for agile methods. To software project managers and coaches in large organizations it provides understanding of the impact of inter-team stand-up meetings on team satisfaction and coaching success.

10.2 Background and Related Work

Learning and knowledge creation are closely tied to human action (Nonaka & Takeuchi, 1995). Literature on learning theories suggests that students progress through two major stages during the development of a cognitive skill, a declarative knowledge stage and a procedural knowledge

stage (H. Taylor, 1999; Anderson, 1982). While the declarative knowledge (the knowing of what), can be acquired from traditional approaches such as lectures and text books, procedural knowledge (the knowing of how) is to be trained on the process (H. Taylor, 1999). From text books we use declarative knowledge about agile methods. Agile practices (Sharp & Robinson, 2004; Williams, 2012) like stand-up meetings (e.g., time boxed, frequent or daily team meetings providing a status update), iteration reviews (e.g., demonstrations at the end of each iteration/sprint to present working functionality) or pair programming (e.g., two programmers writing and reviewing source code at the same workstation) are suitable to create procedural knowledge. They stimulate direct communication and learning through the repeated, iterative interaction of the participating individuals. Agile practices are organizational routines (Salvato, 2009) that support both stages of the learning process and provide a comprehensible understanding of software engineering process in a single teachable framework (Hazzan & Dubinsky, 2007). Industrial practice knows agile coaches, who guide teams through their projects and the iterative steps so that teams improve their agile practice while undertaking the development process (Silva & Doss, 2007; Padula, 2009; Paasivaara & Lassenius, 2011). In the industrial process like in the class room especially stand-up meetings and iteration reviews are useful.

Agile practices contribute a number of benefits to course programs and literature especially describes this for basic under-graduate capstone courses (Hazzan & Dubinsky, 2007). Agile practices contribute to professional routines from several domains such as project management (Venkatagiri, 2011; Rundle & Dewar, 2006) and human-computer interaction (Memmel, Gundelsweiler & Reiterer, 2007) and have been applied in combination with supportive ICT environments (Arakawa & Yukita, 2006). Capstone course projects are often preferred as they provide a good opportunity for students to combine multidisciplinary knowledge acquisition through a coaching routine (Dugan, 2011; Richards, 2009). However, such courses are no vocational training but include academic reflection.

There are few contributions on how to balance coaching routines with academic activities and how to enable their continuous improvement in education (Chao, 2005). Critics have pointed out that the weak influence of research findings on academic educational practice is largely caused by (1) the low status of educational research, which is hampered by a lack of strict methodological rigor, frameworks, and norms (Tierney & Holley, 2008), and (2) the fact that the produced knowledge is not applied in the field, because universities do not emphasize educational research (Tierney & Holley, 2008). In conclusion some call for a more "evidence-based" education where knowledge produced can move in either direction (Stokes, 1997) with a greater involvement in the field based upon multidisciplinary and team-based work (Tierney & Holley, 2008). Agile methods provide access to a large research base through the fields of software process improvement and empirical software engineering with both rigorous methods evaluated in practice and large and active research communities. As such they could provide a stable theoretical ground for reflection on team coaching routines in education.

10.3 Objectives

Tierney and Holley (Tierney & Holley, 2008) argue that educational research should be an interdisciplinary study focussing on solving problems and team orientation to move away from silo orientation of separate academic disciplines. Agile methods, as one possible approach, provide an interdisciplinary framework to address software engineering related problems at both learning stages (H. Taylor, 1999; Anderson, 1982) while providing opportunities for evidence based reflection upon process improvement (Stettina & Heijstek, 2011a). Considering the earlier work, we would like to explore our possibilities by asking the following subquestion:

- *RQ3a: How can we plan software engineering courses so that using agile process improvement techniques we can improve education and contribute to research at the same time?*

To discuss one concrete example of integrating software engineering practice and research in education we have embedded a quasi-experiment in our course design. Within agile capstone courses an extended coaching success has been reported (Layman et al., 2006), however, connected to a higher workload for students and instructors (Layman et al., 2006; Chao, 2005). From our own experience with undergraduate courses applying agile practices (Stettina et al., 2012) we know that the preparation of handouts and the coaching sessions to steer the students through the development process can be very time consuming. How can we justify this additional effort? There is little in-depth research on concrete team-level coaching routines and the implications on workload for teachers and students. Stand-up meetings (Sharp & Robinson, 2004) are integral to coaching in agile methods and due to their iterative nature represent a big part of the workload. As a concrete example of a research activity within our course we thus would pose the following additional subquestion:

- *RQ2h: What are the implications of individual intra-team stand-up meetings on coaching success and team satisfaction compared to bigger inter-team stand-up meetings?*

10.4 Study Context

The context of the study is the *System Development and Project Management* course with the goal to prepare students for the multidisciplinary challenges of ICT projects. Following our research questions we designed a 6 weeks practical assignment for the interdisciplinary course integrating project and stakeholder management and requirements engineering techniques with an agile approach. While the regular course lectures provide the necessary background and theoretical (declarative (H. Taylor, 1999)) knowledge, the practical sessions aim to build up the students‘ procedural knowledge. To do so the students have to develop a project from an initial project bid towards a working demonstrator and present it at a trade fair. The project

is especially focussing at a project's Front-End activities thus those when a project team is not entirely in control of the scope yet and ideas still need to be strengthened within an organization. To stimulate the learning progress of the students we employ two agile practices: Stand-up meetings and iteration reviews with customers (Sharp & Robinson, 2004; Williams, 2012). Stand-ups in agile software development (Sharp & Robinson, 2004; Stray, Moe & Aurum, 2012) are daily team meetings providing a status update to team members. It facilitates information exchange among on potential challenges and enables coordination inside the team. The meetings are generally hold standing and are timeboxed to 5-15 minutes to frame its short and focussed nature. Each coaching session starts with a team stand-up where each group was asked the three common questions: *"What have you done since the last meeting?", "What are you planning on doing until the next meeting" and "What issues and impediments are you facing that prevent you from accomplishing these things?"*. Iteration reviews/demonstrations are applied to involve the customer in the development process and gives the customer a structured way to steer the product development. In this course we use the reviews to advance the students' learning outside the "agile sweet spot" (Hoda et al., 2010) as the scope of the projects needs to be worked out by the students. Due to the focus on the human aspects of software development, we pay particular attention on team building, teamwork and informal communications.

10.5 Method

In this chapter we want to understand the complex topic of planning and embedding experiments based on agile team routines into capstone courses. As this research is connected to a variety of factors such as process, coordination, teamwork and perceptions in a social context we performed a single-case study (Yin, 2009) combining qualitative and quantitative elements. This allows us to explore the complex problem while developing rich and and informative conclusions for further research.

10.5.1 Data collection and analysis

Agile methods put an emphasis on self-managing teams and the integration of individuals into the software development process (Stettina & Heijstek, 2011a). We thus made use of individual perceptions of team members and linked qualitative process descriptions with quantitative questionnaires and the artifacts developed in course of the project. Questionnaires would allow us to collect the individual perceptions in a measurable manner while enabling the participants to state their opinions in an anonymous way. To create a deeper picture supporting our research we further used data from informal interviews, ethnographical notes, observations and analyzed the delivered artifacts. Observations and informal interviews were conducted by the first author during the coaching sessions every iteration. The coaching notes were used

similarly to diaries (Balogun & Johnson, 2004), to track the implementation progress and the challenges of the students. This approach as presented earlier (Stettina et al., 2012), enables to capture the development of selected perceptions of individuals and team throughout the entire project and it allows the comparison of these perceptions to the project outcomes.

To address our second research question embedded into the course, the 30 attending students formed 6 teams and were divided into two main groups: *SUnited* and *SIndividual*. While the 3 teams belonging to group *SUnited* would take part in the weekly stand-up meetings altogether, the 3 other teams belonging to the group *SIndividual* would take part in individual team stand-up meetings. This setup provides a suitable environment allowing the comparison of several teams working on the same project. It allows the analysis of patterns of action emerging in course of the experiment and their implications on the project results. The rationale for choosing intra and inter-team meetings for the experiment was that we wanted to understand the implications of the two different meeting routines on the information exchange and the emerging ideas among the participating teams. To do so we applied two types of questionnaires: First, a short questionnaire for a longitudinal collection of the individual team member's satisfaction with the project, the amount of work and the documentation was used. This part of the questionnaire was collected every week during the project. The approach has been applied by us in a similar study setting based upon an undergraduate course (Stettina et al., 2012) and is similar to the project satisfaction graph as presented by Moe (Moe et al., 2010). The questions were: *How satisfied are you with the project?*, *How satisfied are you with the teamwork in your team?* and *How satisfied are you with the information exchange in this project?* Second, longer questionnaires were applied in the course of the development phase, they were used to collect perceptions on the usefulness of specific artifacts, the distribution of roles within the groups. The questionnaires also gave students the opportunity to anonymously comment on their projects. Sample questions can be found here below: *How useful did you find the stand-up meetings? How useful did you find writing meeting minutes for the weekly stand-ups? Would you prefer to have stand-up meetings in bigger or in smaller groups?*

We chose to administer the questionnaires during the coaching sessions on paper as the response rate was expected to be higher if compared to questionnaires administered online after the class. A consistent response rate was important for the validity of our data. The forms were anonymous and we re-iterated to the students that the collected data was part of the research project and would by no means affect their grades.

10.5.2 Bias and limitations

Several factors limit the generalizability of this study. Although our experience is based upon multiple iterations of capstone courses the here discussed data stems from a single course undertaken in spring 2012, within one university setting with a limited number of teams and students. To counter this we collected the participant's perceptions over time and calculated

the statistical significance. The extent to which graduate, undergraduate and doctoral students are representative of professional software developers and the threat of interaction of selection and treatment respectively, have been addressed in various other studies (e.g., (Höst et al., 2000)).

10.6 Results

In this section we will proceed to describe our findings according to the two research subquestions.

10.6.1 Course Execution

The course project started with an introductory session on February 2, 2012 and lasted until March 15. During the six week of the course, the students attended the class consisting of lectures on two consecutive days each week. The practical sessions in which the stakeholders were introduced and the teams were coached, took place once a week right after the lecture. The lectures were given by the third author and covered topics on software project management and the related technical, economic and organizational issues. At the end of the course a trade fair event represented the finish of the project with about 50 internal and external participants.

During the six week of the course, the students attended the class consisting of lectures on two consecutive days each week. The practical sessions in which the stakeholders were introduced and the teams were coached, took place once a week right after the lecture. The lectures were given by the last author and covered topics on software project management and the related technical, economic and organizational issues. The process of the practical sessions is shown in Table 9.1. The later also depicts the produced artifacts alongside the feedback loops between the involved actors. The main participants of the process were: (1) the development teams, (2) the product owners and (3) trade fair jury and audience.

The introduction of the theme took place during the first session on February 2, and was given by the course lecturer, the ScrumMaster coaching the teams and the Product Owner. This introduction included a presentation on the project environment, stakeholders, examples of possible results and the broadly defined assignment (*"Develop a demonstrator within the OpenLearningLab that improves learning and student collaboration"*). The purpose of this broad task definition was to give the students as much space for their ideas as possible. To implement the project six development teams were assembled from 30 international master-level students with a background in ICT and Business. For the second session the students prepared a project bid, an initial project proposal. The proposal document was read and commented by the product owners and the lecturer. The product owners consisted of a committed stakeholder working on the educational IT strategy of the university and the first author who was a mentor of the original project. While the feedback of the lecturer was

concentrated on the academic part, the product owners gave feedback related to the content in the project context. The procedure was repeated with the third session and the project plan, which presented a more concrete proposal of the students on how to execute the project. The fourth and fifth sessions were dedicated to the development of a minimal implementation of the ideas in the form of a demonstrator. During these two sessions the coach acted as a ScrumMaster, coaching the teams, trying to remove possible impediments and providing feedback on the content.

10.6.2 Research Experiment: The stand-up meetings

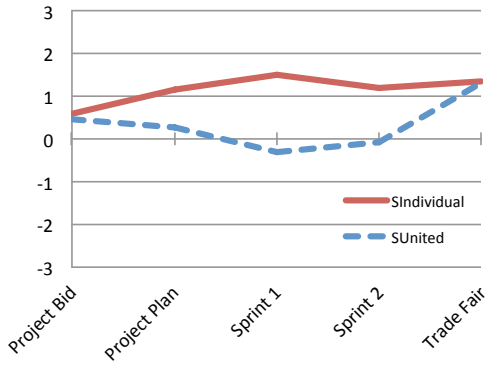
The theme of the embedded research project has been discussed with fellow colleagues and chosen according to state-of-the-art, applicability and contribution to research and practice. Questionnaire collection for the embedded experiment began together with the stand-up sessions during the second week and lasted throughout the entire project. The meetings were timeboxed to 15 minutes for the separated teams (*SIndividual*) and to 30 minutes within the big group (*SUnited*). The session times were adjusted in course of the project (originally 10 minutes and 40 minutes).

The initial observation was that it always took a little longer for the bigger group to gather. This remained constant till the end of the project and while the stand-ups for the separated groups seemed to become sharper in time, with more team members being on time as the time was scarce, the larger group was shrinking. At the last meeting while team A was complete, teams B and C of the bigger group was only partially present. This development is further confirmed by the satisfaction graph in Fig. 10.1a: While the teams belonging to *SUnited* are less satisfied the teams of *SIndividual* are more satisfied with their information exchange. In line with the observations we can see in our survey results that the teams prefer stand-up meetings in smaller groups and that the teams that participated in separated stand-ups, found the meetings more useful (Fig.11.2b).

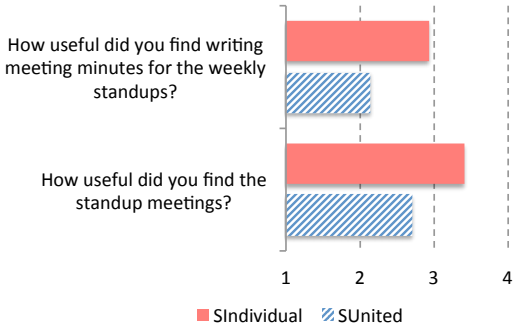
To test the statistical significance of our data we conducted an independent-samples t-test to compare the satisfaction levels in the *SIndividual* and the *SUnited* groups. A P-value below 0.05 is commonly considered statistically significant, while a value of 0.05 or bigger indicates no difference between the groups. There was a significant difference in the satisfaction with information exchange of the *SIndividual* (M=4.42, SD=1.52) and *SUnited* (M=5.15, SD=1.28) groups ($t(112)=2.9644$, $p=0.00371$). There was further a significant difference in the satisfaction with the project ($t(112)=2.7365$, $p=0.0036$) and innovativeness perceived ($t(112)=3.3109$, $p=0.00125$) and workload ($t(112)=1.8397$, $p=0.0685$). We did not find a significant difference in the satisfaction with teamwork ($t(112)=2.0939$, $p=0.0385$). These results suggest that the applied stand-up routine does have an effect on the perceived satisfaction with information exchange and innovativeness inside the teams. Specifically, our results suggest that in the individual stand-ups the team members are more satisfied with the

exchanged information.

Most of the teams had developed (consciously or not) a member acting as s spokesman. This became visible early on and while the smaller groups seemed to invite more team members to participate within the discussion, in the bigger setting it was rather single and interested team members taking part in the discussion while remainder stayed quite or was distracted. This seemed partially caused by the impatience of the team members caused by the similar project backgrounds and the same repeating issues and progress in course of the project. The teams provided short answers as they almost seemed pressured waiting for the next group follow in the stand-up.



(a) Satisfaction with information exchange



(b) Perceptions on the usefulness of stand-up meetings

Figure 10.1: Satisfaction with information exchange and stand-up meetings.[Scale: -3=Completely dissatisfied, -2=Mostly dissatisfied, -1=Somewhat dissatisfied, 0=Neither satisfied or dissatisfied, 1=Somewhat satisfied, 2=Mostly satisfied, 3=Completely satisfied]

10.7 Discussion

In this section, we discuss our findings according to the two guiding research questions. We will first begin discussing the embedded research question and the experiment, and then proceed to discuss our broader objective and the experiences with planning and improving in our course.

10.7.1 The implications of stand-up meetings: intra-team vs. inter-team

In this subsection we will answer the subquestion RQ2h: *"What are the implications of individual intra-team stand-up meetings on coaching success and team satisfaction compared to bigger inter-team stand-up meetings?"*

The advantage of possible knowledge gain among the *SUnited* teams compared to the individual meetings of the *SIndividual* teams was overridden by the decreasing satisfaction in *SUnited*. Information had to be repeated by the coach during the individual sessions, however, this was justified by the advantages such as the higher attention level in the more personal individual meetings. An interesting observation was that during the 3rd stand-up session members of team B, while hearing of a specific need of another team, offered to implement a specific plugin. "if you pay us..", said a team member jokingly. There certainly has been an exchange of students outside the team boundaries, this however, remained the only visible example among the student teams.

During the stand-up meetings we found that the teams in *SUnited* provided shorter answers with little discussion if compared to the teams in *SIndividual*. This could be caused by the fact that the teams with similar goals, issues and challenges the teams felt under pressure knowing that in there is a next group is waiting. This emphasizes that the teams should feel comfortable for a good exchange and discussion. While applying the routine to coach student teams the coach thus needs to establish a relationship of trust emphasizing on his facilitating role. Establishing such a connection is much easier in the individual stand-ups, and allows deeper discussions. Further, the short meeting notes proved to be very efficient to follow up on each team's progress, especially while coaching the six teams in parallel. However, by applying the described routine coaches by all means should avoid mechanically asking the three stand-up questions. Rather the team members should be encouraged to reflect on their planning, coordination and teamwork practices.

As inter-team stand-ups do not work in our experiment, how can we stimulate direct communication among teams on a frequent basis? In large organizations applying agile methods the distribution of inter-team information is often implemented through *"Scrum of Scrums"* (Laanti, 2008), a type of stand-up meetings among ScrumMasters or other designated team members. Although a team lead, a committed spokesman, naturally emerged in all of the teams, it should be further researched how such a meeting could be introduced in a class setting. Maintaining an optimal learning experience for all students while omitting information asymmetries is important to consider.

Similarly to earlier reports (Sharp & Robinson, 2004) we have observed that development impediments could be identified and addressed early on during the project. Especially as the projects building on existing infrastructure, the stand-ups improved the communication in terms of communicating technical specifications and known limitations of the system. They proved very efficient to intercept communication gaps between the project and the stakeholders.

The iteration length of one week proved to be appropriate. According to our experience an iteration length of one week seems appropriate for capstone courses lasting up to three months, especially at the beginning and on bachelor level. In our eyes this is not so much related to the amount of work, which would be also influenced by the amount of courses taken by the students in parallel, but rather to the repetition rate necessary to absorb the process in a

specific timeframe.

10.7.2 Software engineering courses improving education and contributing to research at the same time

In this subsection we will answer the subquestion RQ3a: *"How can we plan software engineering courses so that using agile process improvement techniques we can improve education and contribute to research at the same time?"*

The outcome of this particular embedded research experiment is that the next iteration of the class will be held in separate coaching sessions for each group with an iteration length of one week. The outcome of this particular course iteration is that the teaching staff can draw upon these findings improving the course during the next iteration and contributing to research. While coaching of 6 teams is education, it is a research setting through careful design of a study about the processes applied by the teams.

Fig. 10.2 depicts the feedback cycle based upon the course design. The initial research backlog, the accumulation of possible research items waiting to be done over time, is compiled from current state-of-the art (e.g., from a literature study on team routines in SE). During course preparation the university staff or coaching team agrees upon a piece of research, a particular experiment, to be executed within the course. Possible items for such a research backlog would be: application of new technology in teams (e.g., new collaborative solutions such as Google Docs or Dropbox) or testing new team based technologies in context (e.g., pair programming or pomodoro techniques). During the course the students teams are coached and data for the experiment is collected. After course completion a retrospective is being held together with the students, data is evaluated and a report being written. Then the circle begins anew moving to the next research backlog item with the next iteration of the course.

The attention on intensive coaching in our study is driven by the need of students with little experience to acquire professional knowledge from multiple domains which can only be trained on the process (H. Taylor, 1999). Literature suggests that the understanding of such everyday action is necessary to understand capabilities of teams and organizations and that their study is particularly difficult (Salvato, 2009). Without explicit practice it is almost impossible to study agile routines. How are they established or learned? And how do they evolve in teams? Our quasi-experiment provides a good example on how software engineering routines as patterns of human action evolve in teams over time.

Our quasi-experiment provides a good example on how software engineering routines as patterns of human action evolve in teams over time. In that sense we present our case for software engineering education based upon observable and concrete patterns of action performed by the participants. Agile routines such as stand-up meetings are observable micro-activities and produce measurable outcomes to enable process improvement. Our course design is driven by its consideration of use in educational environments as well as it aides the

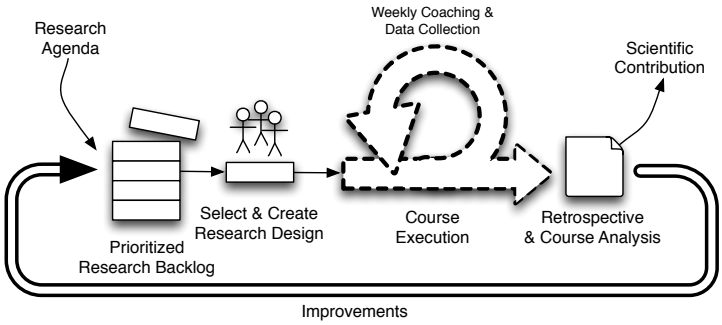


Figure 10.2: Course planning, execution and improvement cycle

fundamental understanding of routines in practice. Following the use-inspired basic science of the Pasteurs Quadrant (Stokes, 1997), the workload of the training routine in our class is thus not only justified by its contribution to the maturity of the students, but also by its contribution to research, thus to the maturity of the tutors.

Crucial for such a course design is the transparency towards the students and sharing of learning outcomes while staying consistent with academic principles to avoid doubts among students. It is important to reiterate that such continuous learning is important in their everyday work life and that they should be aware of teamwork aspects. It is necessary is to pay attention to sufficient training of the coaches in this case, especially if new coaches are added to the team. The course design requires a careful preparation and discussion of the chosen research theme, however, contributes significantly to practical experience to research and coaching of the involved coaching staff.

10.7.3 Recommendations for Future Research and Practice

The perspective on coaching based on team routines raises a number of questions for future research. With the increased demand for coaching, the collaboration amongst coaches becomes increasingly important. On the bachelor level we are currently experimenting on the application of multiple coaches in one course. Therefore, future research is encouraged to explore our results in different contexts. For example, in different in different courses with a professional background and group sizes. Another direction is regarding the applied research method. With the increasing importance of routines in creating knowledge, how can we improve the techniques to study routines and create experiments in-class?

10.8 Chapter Conclusions

In this chapter, we discuss our experiences with academic education of professional processes in software engineering, especially addressing the balance of practical coaching activities to academic reflection. We believe that students' learning experience can be enhanced by frequent feedback loops of agile coaching routines with processes close to practice and theoretical reflection.

As a partial answer to RQ2: "*What are the components of governance necessary to understand knowledge worker project organizations?*", we discuss the relation of routines and intensive coaching as components of governance. As a partial answer to RQ3: "*How can governance address the dynamics of knowledge work across multiple knowledge worker teams?*", we may conclude that governance in knowledge worker team organizations can be enabled through intensive coaching of team routines.

First, we advance the understanding of teaching in software engineering by discussing our approach to planning and improvement of software engineering capstone courses based upon intensive coaching and the notion of team routines. We present a graduate course design based on agile practices and discuss its contribution to the students practical and academic maturity through an iterative coaching routine. The intensive coaching is shorter in nature, and thus more more appealing to the students as our data shows. Process research and improvement is difficult, the education of professional practices takes time and needs to be taught on the process. By making academic courses more realistic it is possible to enable a better student experience and allows access for researchers to analyze routines. The application of a generic questionnaire throughout the course allows furthering the knowledge of educators and students by means of process improvement.

Second, we discuss a concrete example of our approach by conducting a quasi-experiment on the impact of two different stand-up meeting settings. Larger setting with three separate teams in one meeting and a smaller setting where each team holds a separate stand-up. Our results show that coaching per team costs little additional effort while providing much better information exchange and student satisfaction. Instead we found that the additional effort is not only justified by its contribution to maturity of the students but also by furthering the knowledge of educators and students by means of process improvement. This contributes to continuos improvement of the course design, as well as to improvement of agile methods in practice.

Part V

Conclusions

11

Knowledge Worker Governance Revisited

"Project X is part of a large IT project portfolio. The portfolio is distributed across different sub-organizations. For each project a plan is made, budget approved and defined deliverables expected in accordance to the plan. Project managers generally report on budget, achievement of a specific milestone using the common red, green and orange status indicators.

Project X, similarly to the remaining projects in the portfolio, usually reports a green status. While management concentrates on correct budget numbers and assumes a successful project progress, through informal channels rumors start slowly making their way to the PPM office - indicating that the status of the project reality is rather concerning.. "

Within the empirical parts of this dissertation I have drawn a view on knowledge worker organizations across the three layers of the individual, the team and the organization. In this chapter I will elaborate on the clash of knowledge worker teams with structures in existing project-based organizations. I will point out why knowledge worker teams need to be governed on knowledge rather than on information and what implications this has for the role of management. Meanwhile we discuss the research questions RQ1 ("*How can governing bodies in organizations of knowledge worker teams understand and steer multiple knowledge worker project teams in practice?*"), RQ2 ("*What are the components of governance necessary to understand knowledge worker project organizations?*"), RQ3 ("*How can governance address the dynamics of knowledge work across multiple knowledge worker teams?*"), and even the problem statement ("*Is governance of innovation project management necessary or neglected?*").

11.1 Governing Knowledge Worker Teams

The scenario outlined above portrays many large knowledge work organizations today. In particular it points at two challenges knowledge worker teams face today: (1) they are assumed to follow a concrete plan, and (2) they are governed on information rather than knowledge on the project in context. It points at a existing governance problem in an project-based organization executing multiple projects across multiple knowledge worker teams.

As suggested by some authors large organizations have been largely designed following the needs of mass production (Piore & Sabel, 1984). They are bureaucracies optimized for the control of unskilled workers (Mintzberg, 1979), with their work being managed on input and output variables (F. W. Taylor, 1911). It is what Mintzberg (1979) calls *Machine Bureaucracies* - organizations with a large degree of predictable tasks in environments that are simple and stable. Even though these organizations do not engage in production anymore, I argue that the underlying assumptions are still visible in the design of many project-based organizations today.

While managed just like predictable manual labor, the work accomplished in the scenario above is a complex and creative task. Similarly to that of medical teams, law firms and design offices they are knowledge workers. As laid out by Drucker (1994) and in Part II of this thesis, the rise of knowledge workers (Drucker, 1994) inherently challenges the Tayloristic perspective on organizations optimized towards the mass production of goods. As opposed to the blue-collar workers of the industrial age, knowledge workers are contributing through the application of theoretical knowledge and analytical skills. As such they need to be intelligent and well educated. Knowledge workers and the knowledge they carry are an asset to the organization (Drucker, 1999).

Existing project management frameworks do not prove suitable for knowledge worker projects and have been attributed for a lack of understanding in context (Cicmil et al., 2006). Existing frameworks have been criticized to be too plan driven, normative and not appropriately covering the complex nature of knowledge work projects, their development, often missing to deliver the appropriate value to its beneficiaries (Nerur & Balijepally, 2007). In knowledge work projects delivering new products and services, the benefits are discovered over time as ideas emerge.

The creation of knowledge and the collection of ideas is less predictable compared to the output of machines or manual workforce. The amount of ideas to be collected to complete a project is not comparable to the amount of bricks necessary to build a house. As the participants, project teams, beneficiaries and users learn the possibilities and the actual value of a project (Larman, 2004), such project team cannot be steered in a mechanical Tayloristic way. Projects developing new products and services can take an unpredicted course due to the difficulty to product knowledge creation and due to internal and external influences (e.g., markets, new developments). It ultimately leads to a lack of understanding of the project

environment, its development and the value it aims to deliver to its beneficiaries.

In this research I follow the case of agile teams, I study their characteristics, how they manage knowledge work projects and what implications it has on the role of management in project-based organizations. I observed that while successful in individual separated teams, agile methods need a different approach to governance and a different role of management as a coordinating body. As discussed in Chapter 2 (Section 2.3.2) agile methods address the uncertainty of knowledge work through two integrated planning and execution routines (Carlile, 2004) enabling a transfer, translation and transformation of knowledge rather than information only.

11.1.1 Putting Agile Teams into Context

While working well for individual and separated projects Scrum is an organizational blueprint, a template that works optimally in very specific project settings - Its configuration of roles, routines and artifacts is optimized for new product development projects of software products for a team size of 5-9 team members. However, when the size and complexity of a product or service-to-be grows outside the scope of an individual team, agile teams need to be embedded into an organization of teams. This challenges governance in existing organizations.

When agile teams are embedded in a project-based organization a number of challenges becomes visible as elaborated in Chapter 7. First, being that existing large organizations, their routines are aligned towards a plan-based delivery of projects. Second, a lack of commitment and involvement of senior management staff in the process. Third, missing autonomy of teams when tasks are allocated to individuals rather then teams. The existing structures (e.g., multiple projects across diverse functional departments), are challenged by the increased frequency of interactions as required by Scrum and discussed in Chapter 7.

Agile teams are knowledge worker teams. According to (Drucker, 1999) knowledge workers need a level of autonomy regarding their team and their ways of working. However, knowledge also needs to be facilitated across the teams. It needs to be put into perspective with business, marketing, production, strategy, tactics, and operations. Management involvement is considered as one of the major challenges as staff is not used to give so close feedback such as required by the role of the product owner in Scrum. It does not pay justice to the asset knowledge and knowledge workers bring into an organization. According to my findings the application of agile methods thus has implications on the routines in practice, organizational routines and culture (compare Chapter 7).

Role descriptions in traditional project management frameworks as described in Section 2.2.3 and to be elaborated in Section 11.1.3, are still positioned towards a Tayloristic management style on information based in the roots of Scientific Management (F. W. Taylor, 1911). They are Tayloristic in the sense that they are dependent on a concrete plan and focused on the delivery of the originally envisioned project output and control of the project staff.

However, the prediction of such a product-to-be turns out to be difficult as laid out in the literature study in Chapter 2. Following my case studies the encountered knowledge organizations follow a governance model of a machine bureaucracy where managing is a rather content free task relying on information rather than knowledge.

The findings presented in this thesis are complementary to research presented in literature on agile project management. In multi-project organizations the constellation of Product Owner providing direct feedback on product and project priorities in short intervals is challenged. This inter-team review and coordination has been reported troublesome for an individual product owner. Hoda, Noble and Marshall (2011) report how a lack of product owner availability, thus lack of feedback of a customer representative impacts the project outcomes in agile projects. Lehto and Rautiainen (2009) report a case where this led to the division of the role into a commercial, technical and resource responsibility, which resulted in coordination and communication chaos. Talby and Dubinsky (2009) present a study of a large-scale critical system development within the Israeli Air Force. They describe different governance stages and mechanisms including their roles and responsibilities. They found governance to be effective when performed at the level of a development iteration, during the iteration summary meetings. Bass (2014) describes what functions teams of product owners in agile multi-team organizations must cover. He distinguishes information gathering functions: groom, prioritiser, release master and risk assessor, and information disseminating functions: communicator, traveller, intermediary, technical architect and governor. Vlietland and van Vliet (2014) identify issues in interdependent Scrum teams: (1) Coordination: a lack of coordination among the teams, (2) Prioritization: mismatches in backlog priorities, (3) Alignment: alignment issues, (4) Automation: lack of automation of the IT chain process, (5) Predictability: difficultly to predict a product across several teams and the dependencies arising, (6) Visibility: lack of information visibility across the teams.

Thus, on the one hand, the multi-team multi-project methods known in traditional project management literature are challenged by knowledge work and the dynamic reevaluation of plans. On the other hand, agile methods describe mechanisms to organize knowledge work projects only on the level of individual project teams. This points at implications on the governance process and structure, which need to make sure that knowledge worker projects are appropriately guided.

11.1.2 Governance of Projects: Knowledge vs. Information

Following my findings in Chapter 7 I argue that we need to revise our view on the role of management in project-based organizations developing new products and services. Governance defines the roles of project members, management and their mandates in project organizations, as such it defines the "rules of the game" (Ahola et al., 2013) in project work. Following the view on the nature of knowledge workers as provided by Drucker (1994) versus the manual

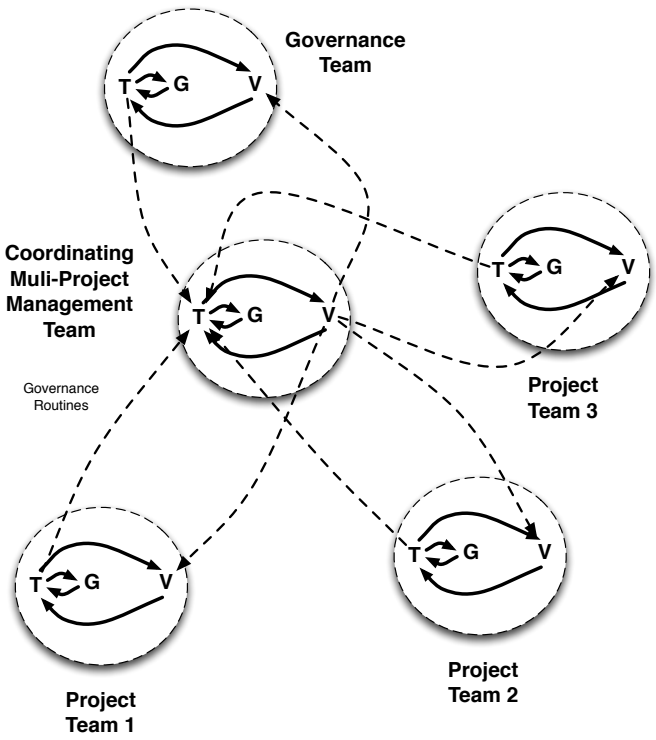


Figure 11.1: Model of a multi-project organization maintaining feedback loops across daily tasks (T), goals (G) and vision (V) (analogously to strategy, tactics and operations)

worker as discussed by F. W. Taylor (1911) I argue we need to specifically reconsider the way management interacts with knowledge worker project teams.

The knowledge worker organization is not an information processing machine with the throughput from input to output being the single most important metric. In a system optimized for mass production, quantity, the throughput from input to output, is the single most important metric. Acknowledging the knowledge and its tacit component (Polanyi, 1967; Nonaka & Takeuchi, 1995), the intangible nature of knowledge creation makes a different approach necessary.

In order to understand organizations suitable for knowledge workers we need to understand knowledge workers and the factors which influence their productivity. Drucker (1999) identifies six major factors determining knowledge worker productivity, such are: (1) The knowledge of what the task is, (2) autonomy, as knowledge workers are responsible for their own productivity, (3) continuing innovation, (4) engage in continuous learning, (5) quantity is not a primary measure of productivity, as quality of knowledge worker output is as least as important, and 6) acknowledging that the knowledge worker is an asset to the organization rather than a cost factor.

Static information is not sufficient to cover the tacit components of knowledge work and the resulting intangible dynamics. While the explicit nature of manual work is generally tangible and thus rather easy to communicate, teach and measure, the nature of knowledge work can be very intangible. We can easily grasp how to transport bricks from A to Z or even solder complex electric components, however, concepts envisioned, theories or the inner workings of software systems can be very difficult to communicate across team members. Neither exact course of the project nor the routines can be fully envisioned all up-front as discussed in Chapter 8. Teams developing new products and services rely on a constant understanding, translation and transformation of knowledge in context. As such agile teams cannot be governed on information only.

As sketched in Figure 11.1, the vision (V) of each team, the respective goals (G) as well as the resulting tasks (T) need to be coordinated, translated and transformed across the teams which might have different educational history and functional backgrounds within an organization (e.g., marketing, production).

Knowledge worker teams need to be managed and governed on knowledge, on *understanding* of the project in context. In order to establish a shared understanding a so called *shared mental model* as discussed in Chapter 4 and 8, it needs to be established, constantly negotiated and maintained in frequent discussions across the involved domains. Carlile (2004) argues this must be embedded in an iterative routine which supports the translation and alteration of knowledge across multiple knowledge domains in and across multi-disciplinary teams.

Agile methods establish such a shared understanding by promoting direct communication in ongoing feedback loop routines (e.g., daily stand ups, iteration reviews). However, embedded in a wider scope of a multi-project organization, this turns out to be difficult as the number of

communication lines to be maintained rises rapidly. Project work in interdependent teams for example, requires collaboration, coordination and communication (Sharp & Robinson, 2010). To support this the notion of boundary objects as as a guiding artifact as elaborated in Chapter 8 becomes particularly useful.

In the scenario presented at the beginning of this chapter, portfolio management staff follows a routine which does not support creation of adequate understanding. While the applied boundary object, the portfolio dashboard, might contain the budget numbers in accordance to the original plan, the dashboard can cover only a subset of information regarding the project reality.

11.1.3 The Role of Management and the Knowledge Worker Teams

I argue that role definitions in current project management frameworks as maintained in many organizations are still Tayloristic and plan driven. They are dependent on a concrete plan and focused on the delivery of the originally envisioned project output. These role definitions emphasize the existence of a concrete plan, which a project team only needs to implement. Following the traditional definition the role of program and portfolio managers is perceived as: *"The portfolio manager will receive component performance information and convey to the Portfolio Review Board how the components as a whole are aligned with the strategic goals."* (PMI, 2008). The role of the program manager is defined as *"The program manager must help ensure that the components in his or her program perform according to plan and achieve the strategic goals associated with the program."* (PMI, 2008). According to Kraut, Pedigo, McKenna and Dunnette (1989) the role of management and the underlying routines are associated with managing individual performance, instructing subordinates, planning and allocating resources, coordinating interdependent groups, managing group performance, monitoring the business environment, and representing ones staff (Kraut et al., 1989).

In contrast to this other authors closer to the knowledge work domain discuss a closer involvement of management in operational activities. Nonaka and Takeuchi (1995), for example, call middle managers as knowledge engineers. Drucker (1994) says *"A manager is responsible for the application and performance of knowledge"*. In accordance to this Bass (2014) describes what functions teams of product owners in agile multi-team organizations must cover. He distinguishes information gathering functions: groom, prioritiser, release master and risk assessor, and information disseminating functions: communicator, traveller, intermediary, technical architect and governor.

Following my observations and existing literature I argue that managers play a central role in the knowledge creation process in project-based organizations. Agile teams take over many traditional management tasks such as estimating and planning. Autonomy and self-management play an important role in agile teams as elaborated in Chapter 4, however, strategy and vision need to be coordinated across the teams and translated into operations and daily

tasks.

In our multiple case study in Chapter 8 discussing the application of agile project management methods across 14 large organizations, our participants demand for more interaction across the different teams and domains of practice. In Chapter 8 I discuss how results are transferred across project teams. I argue that it is the task of coordinating management bodies to maintain this understanding across the teams.

The vision of an organization needs to be translated across all its teams and layers, to resources, goals and tasks. Analogously, strategy needs to be translated to tactics, and operations.

Hanssen and Fægri (2008) discuss how software development and product line engineering are integrated to support the strategic and tactical goals by combining three interacting customer-centric processes in their case company: strategic, tactical and operational. In agile literature further Vähäniitty (2012) and Leffingwell (2007) discuss how interaction is enacted across "top management", "strategic release management" and "development management". Senior management support crucial by project management literature (Young & Jordan, 2008). Our case participants repeatedly pointed at the wish for a closer collaboration across the layers. Nonaka and Takeuchi (1995) describes the example of how the interplay of young front-line engineers, middle and senior management lead to the creation of the highly successful Honda City product line. They draw out how management helped to direct the young team by transforming abundant but ambiguous information from the marketplace towards meaningful knowledge in context.

Blomquist and Müller (2006) draw out that organizations should adapt their governance structure according to the needs of their context and project types. They state that middle managers in program and portfolio management should be personally involved in (1) handling project related issues, (2) conducting steering group meetings, (3) identification of bad projects, and (4) reviewing projects.

I argue that it is the task of coordinating bodies (e.g., management) to recognize and facilitate an appropriate balance of these two forms of knowledge. Carefully chosen Boundary Objects (Carlile, 2004) (e.g., prototypes, documents, diagrams, user stories) are used as guiding artifacts to enable an effective knowledge transfer, not only information transfer. In order to be able to recognize such an appropriate balance in context, management staff must have experienced such an appropriate balance in practice. The individual must have been exposed to appropriate boundary objectives and respective routines in practice.

As the importance of knowledge work in our society has increased steadily, we have to ask ourselves why this challenge has not been addressed earlier? Management support is considered one of the factors most important to the success of individual projects (Young & Jordan, 2008), but is also difficult to implement. As reported in Chapter 7, this has been similarly observed in our case organizations applying agile methods in multi-project organizations. In course of this project I have encountered resistance to change of management staff in a similar manner as

presented in practitioners literature (cf. Eskelinen (2012)). Eskelinen (2012) discusses seven reasons why managers might resist an agile adoption: (1) fear of losing their job, (2) new role hasnt been clearly defined, (3) loss of status & power, (4) the way the managers are led hasnt changed, (5) fear of not being capable of working in the new way, (6) no time, and (7) incentives and other policies are not aligned.

Management bases its assumptions on successful experiences made in the past, often build on classic management theory and the roots of project management in civil engineering. But in particular middle-management, important to establish the infrastructure across teams, is often not actively encouraged to actively be involved in project content and not measured against the supervision of these so important routines.

I argue that this lack of management support is caused by a missing awareness of the nature of knowledge worker tasks, a lack of adequate project management frameworks and inadequate governance in the respective multi-project organization. This points at the necessity to adjust governance structures and reevaluate job profiles, performance appraisal, and reward systems of management roles accordingly.[1]

11.2 Routines and Boundary Objects as Carriers of Governance

(a) Backlog grooming with User Stories

(b) Discussing a change initiative using a change canvas

Figure 11.2: Two examples of team routines to support managing on knowledge

In the previous section I elaborated why knowledge worker teams imply a different role and closer involvement of management in the continuous translation of strategy, tactics and operations alongside of vision, goals and tasks. In this section I will discuss (1) how

[1]Further suggestions, as presented by Eskelinen (2012) are: (1) organize training oriented to managers, (2) let managers participate in defining their new role, (3) arrange coaching for managers, (4) apply principles of self-organizing teams to managers, (5) support peer learning networks for managers, and (6) align policies.

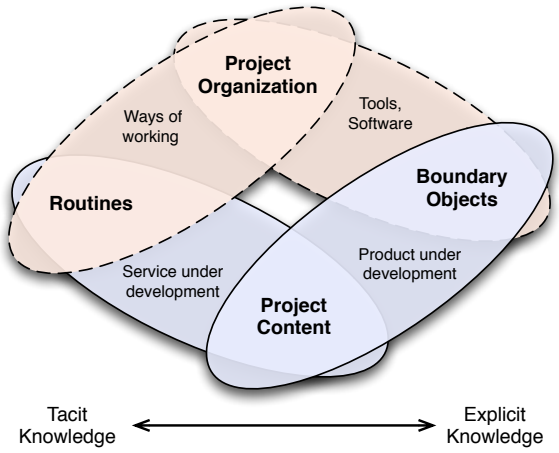


Figure 11.3: Routines, boundary objects and their relation to project content and organization

routines supported by boundary objects facilitate this interaction in organizations applying agile project management methods, and (2) how companies applying agile methods educate and continuously improve their team routines.

In order to understand governance of knowledge worker project organizations one needs to understand how knowledge is transformed across strategy, tactics and operations (project organization) alongside of vision, goals and tasks (project content). Further one needs to understand knowledge creation across these layers and possible boundaries to knowledge creation in organizations. Here I put forward the theory of knowledge creation as presented by Nonaka and Takeuchi (1995) and the concept of boundary objects as discussed by Carlile (2004). Following the seminal theory of Nonaka and Takeuchi (1995) knowledge creation is resting in conversions across tactic and explicit forms as depicted in Figure 11.3 (compare also Figure 9.1 on Page 150). Further, as Carlile (2004) explains different boundaries exist to knowledge creation across different functional domains in organizations: syntactic (transfer), semantic (translation), and pragmatic/political (transformation).

Specifically I would like to point at the role of routines and boundary objects and their distinct features to carry the tacit and explicit components of project organization and content. Routines play an important role in the creation of capabilities as long acknowledged in organization science literature (Cohen et al., 1996; Feldman & Pentland, 2003; Becker, 2004; Salvato, 2009). In their seminal work Levitt and March (1988) discuss organizational learning as based on routines, being dependent on previous experiences (history) and being target oriented. They argue that experiences and lessons learned are lost during personnel turnover

unless the experiences made and their implications can be transferred from those who did these experiences to those who did not (Levitt & March, 1988). Following earlier reports in product development (Salvato, 2009), we found organizational routines playing a particularly important role in carrying governance in organizations applying agile methods. In Table 11.1 I present examples of routines and boundary objects applied across software project management (compare also Table 8.1 in Chapter 8).

But why is it so important to continuously (re)consider ways of working and the knowledge carried by routines? Considering both project content (vision, goals and tasks) as well as organization (strategy, tactics and operations) is important for governance in the sense as following the view of Drucker (1994) knowledge workers cannot be controlled on input-output only Mintzberg (1979). Rather, they can be governed on standardization of work routines and their execution. For example, it is unrealistic to plan all steps of a medical intervention ahead, however, the governance board of a hospital can strive for the best education of their staff and routines (protocols) applied. This has implications on the way we look at project management frameworks as well as the way project content is treated (e.g. the product or service to be).

Following the ideas of Mintzberg (1979) on professional organizations, such as those described in this study, knowledge workers (and the teams they are part of) need decision autonomy on their ways of working. This autonomy is considered as one of the main characteristics of agile knowledge worker teams as discussed in Chapter 4. Governing bodies, however, can make sure knowledge workers are enabled with appropriate knowledge through equipping them with the right routines. For example, a meeting of the steering committee not only addresses how the ongoing activities relate to the goals and vision, but also how these activities are executed. If shortcomings are noticed, feedback is provided and measures can be taken. This is part of a mentoring/coaching routine frequently observed in coaching which is frequently applied in organizations using agile project management methods (Hanly, Wai, Meadows & Leaton, 2006; Fraser et al., 2003; Benefield, 2008; Silva & Doss, 2007; Padula, 2009; Paasivaara & Lassenius, 2011; OConnor & Duchonova, 2014).

Routine knowledge is tacit (Becker, 2004). This has implications on how routine knowledge can be transferred and how managing or coordinating bodies can facilitate this transfer. According to the four modes of knowledge conversion (Nonaka & Takeuchi, 1995) tacit knowledge can be acquired through socialization (e.g. coaching) or internalization (e.g. reading and following protocols, books). I argue that middle managers play an important role in the maintenance, transfer and improvement of team routines and that governing bodies can ensure that the right routines are in place. I argue that management and coordinating bodies need to be able to understand the routines in context and steer accordingly.

Further, although one might be inclined to consider routines as ways of working and the developed products as boundary objects the concepts are more interwoven. Supportive software, tools, protocols are boundary objects which influence the ways of working and which need to be accessible across the involved domains. Further, routines are part of each

Table 11.1: Example routines and boundary objects in project-based organizations (∗Boundary objects supporting transformation of knowledge across the pragmatic boundary (Carlile, 2004))

	Routines	**Boundary Objects**
Capability Development	Continuous Improvement	
	• Retrospective	• Ishikawa diagram∗
	• ScrumMaster	
	• Improvement	• Change Canvas∗
	Initial Training	
	• Workshops	• Games
	• Coaching	• Slides
Project Operations	Analyzing	
	• Backlog Grooming	• User Stories∗
	• Iteration reviews	• Iteration Backlog∗
	Planning	
	• Estimating	• Planning Poker∗
	• Sprint planning	• Iteration Backlog∗
	Implementing	
	• Daily stand ups	
	• Documenting	• Documents, Illustrations
	• Time management	• Kanban boards
Management and cross team Coordination	Coordination	
	• Resource Allocation	• PPM dashboard
	• Prioritization	• Product Backlog∗
	Collaboration	
	• Pricing	
	• Contracting	• Contracts
	• Quality control	• Acceptance criteria∗
	Communication	

service and product to be delivered. For example, how is that product or service to be applied in context. All these components want to be considered to deliver a successful project. In Figure 11.3 I elaborate how the concepts relate to each other.

In this subsection I would like to present a number of examples of such routines to illustrate this. Across our cases we found routines applied in knowledge worker teams to organize their work. I would like to emphasize that the examples here are not aiming to be a complete list of routines applied across knowledge worker teams. Such are very contextual and as explained in Chapter 6 it is impractical or even impossible to document all their variations. Rather, the table aims to provide an initial structure to governing and managing bodies.

1. *Capability development and improvement*: I have observed education as well as improvement of routines as part of (1) the initial training as well as (2) continuous learning activities.

2. *Project operations*: Project operations is the realm of project management frameworks such as Scrum or PRINCE2. The most applied routines in software product development teams, for example, are according to Williams (2012): continuous integration of results, development in short iterations, automated testing, iteration reviews and retrospectives. I have to emphasize that Scrum does not cover all aspects of traditional project management methods such as finance on one hand, while on the other hand, the teams take over estimating and planning, traditionally done by management only. Compared to the competence baseline provided by the International Project Management Association (IPMA, 2006), for example, Scrum, covers project management routines only partially. Cost and finance, program and portfolio management are not described. Resource allocation is described only on task level inside an existing team. A more complete list of specifically agile practices has been surveyed by Williams (2012).

3. *Management and coordination*: Scrum teams take over many management tasks such as estimating and planning. Bass (2014) describes what functions teams of product owners in agile multi-team organizations must cover. He distinguishes information gathering functions: groom, prioritiser, release master and risk assessor, and information disseminating functions: communicator, traveller, intermediary, technical architect and governor.

11.2.1 Project Organization: Routines and Boundary Objects as Tacit and Explicit Components of Organizing Project Work

In course of this study I have frequently observed that in practice project managers think in terms of concrete routines applied rather than a preselected set of practices prescribed by project management frameworks frameworks. When asked how they execute projects, project managers might refer to high-level frameworks such as PRINCE2 or Scrum, however, when

asked to specify their work they generally only refer to a few concrete routines such us as definition of the project plan or the usage of some of the templates. Out of the entire framework description consisting of perhaps a few hundred pages, only a few activities are applied. This has implications on how project management capabilities should be established and shared in an organization.

Across my agile case organizations building of capabilities was frequently supported by defined team routines coached in context. Research on routines comparing traditional and agile projects show that agile projects are executed in a more orderly, while projects using traditional methods in fact showed much more deviation and range of activities executed ad-hoc (Thummadi et al., 2011). I argue that this is caused by the definition of team level routines as well as the iterative delivery of the project as the frequent and consistent iterations provide structure to the project.

Scrum helps enabling project operations by concrete team routines as presented in Table 11.1. Such are ceremonies as *Backlog Grooming* (the initial collection of project requirements with the client, see Figure 11.2a), *Iteration Reviews* (presentation of iteration results), or *Retrospectives* (workshop to enable continuous improvement at the end of each sprint). The case organizations applying Scrum often apply coaching in order to transfer and contextualize routine knowledge through intensive coaching (socialization of knowledge). Further, continuous improvement of routines through project retrospective, and project post-mortem (Dingsøyr & Hanssen, 2002) sessions are considered an integral aspect of Scrum (Schwaber & Beedle, 2001).

In Chapter 7 I elaborate how agile team routines have impact existing project-based organizations. In Chapters 10 and 9 I draw out how even small changes to a routine can have a large impact on the effectiveness and the sustainability of project work in practice. In Chapter 6 I study how medical teams, which by definition need to be agile, adapt their routines to unforeseen local variations. I study how they partially document their routine knowledge by making them explicit in process models. These findings emphasize the necessary decision authority for knowledge worker teams as it is most impractical and at times impossible to predict and model all steps of a live intervention. The findings emphasize how routines as dynamic patterns of team action influence the success of project work and how routines as part of collective action embody the tacit components of knowledge.

Next to direct coaching (*socialization*), boundary objects and artifacts in general are used to delegate intentions to non-human actors (D'Adderio, 2011) in a twofold manner: Firstly, as boundary objects supporting a reflective team process in implementing and improving routines. Examples of such boundary objects are the Change Canvas as depicted in Figure 11.2 or the usage of the *Ishikawa diagram* as discussed by Dingsøyr and Hanssen (2002). Secondly, boundary objects being a direct part of a routine, such as *User Stories* or *Backlogs*. Sharp et al. (2009) discuss the use of user stories embedded in backlogs. In Chapters 5 and 8 I discuss the use of project management software to support agile teams. Project management supportive

software as well as repositories are widely applied across our case organizations, inside as well as across teams as elaborated in Chapters 5 and 8.

In both cases there is evidence that the applied software tools need to be embedded in an appropriate work routine. How strongly routines and boundary objects are interrelated is exemplified in Chapter 9 and the following quote: *"[W]e have a wiki that we are supposed to use.."*, as one ScrumMaster comments (see Chapter 5, p.88). If the envisioned routine as the boundary object is not in line the routine will not work.

Following the observations made in course of this study I argue that we need to better understand project work as a set of routines executed in and across the teams. Knowledge workers receive university-level education covering formal aspects of functions like sales, software engineering and research, however not how to execute projects and work in teams.

Agile teams take over many traditional management routines such as estimating and planning. Across the cases I see a shift of management towards coordination and collaboration across the individual teams, resource allocation, acceptance of delivered (partial) results. As the interface across the teams, management requires easy access to knowledge in order to make sense of the projects and their results in context. Agile methods promote direct communication embedded in iterative feedback routines, however, such are quite often supported by boundary objects as presented in Table 11.1.

11.2.2 Project Content: Routines and Boundary Objects as Tacit and Explicit Components of the Product or Service under Development

Analogously to project organization, routines and boundary objects are useful concepts to understand project content. Understanding the value to be delivered by a product or service-to-be is difficult and needs constant feedback loops across multi-disciplinary stakeholders to translate product vision to goals and daily tasks. Further, knowledge needs to be created and maintained throughout respective parts of an organization.

There are different knowledge boundaries which need to be taken into account to enable transfer, translation and transformation of knowledge across these layers and different functional domains Carlile (2004). Translating a product vision to concrete goals and tasks requires the crossing of the political (pragmatic) boundary (Carlile, 2004), in which participants across different sub-organizations (sales, marketing, development) need to transform their own knowledge. They will likely need to make compromises in oder to have a well adjusted product or service, suiting customer needs, with costs fitting market needs, with safety regulations being met etc. It requires appropriate boundary objects to facilitate such a transformation of knowledge.

One example of such boundary objects as applied by agile methods are user stories (Sharp et al., 2009). The photograph in Figure 11.2a depicts a *Backlog Grooming* session in which

requirements are written by non-technical participants, supported by a coach. Such enable the transfer, translation and transformation (Carlile, 2004) of user requirements which are accessible for and can be transformed across a team of multi-disciplinary participants. While traditional requirement documents provide carefully written out information about a product or service-to-be, such artifacts are difficult to access by non-technical participants, making the requirements collection process potentially ineffective and error prone.

The example of the portfolio dashboard presented at the beginning of this chapter contains only quantitative indicators on financial performance of the respective projects. In oder to cross the political boundary of a portfolio management process such an artifact needs to include all important components of governance such as project content, people, routines Waterman Jr et al. (1980); McLeod and MacDonell (2011). Such an understanding can ultimately only be established and maintained when management is exposed to and can make sense of environment. Following Carlile (2004) this must be embedded in an iterative routine. He can thus closely follow the project in frequent reviews to make sense of the project context.

The usage of boundary objects always involves some externalization and/or internalization of knowledge in some form as boundary objects are by nature explicit knowledge. Only though interaction with these artifacts the embodied explicit knowledge can become tacit (Carlile, 2002).

In Chapter 8 I study in depth which routines teams apply to transfer knowledge to other knowledge worker teams.

While traditional software development methods follow a tradition relying mostly on information captured in documents (codified, explicit knowledge) to transfer project results, agile teams emphasize in the effectiveness of direct communication. While the former tried to capture as much as possible in large textual descriptions which lead to the creation of large documents which were difficult to read, the latter experienced knowledge losses when team members left the project as described in Chapter 5. As elaborated in Chapter 8, for an appropriate sustainable and effective knowledge transfer in and across knowledge worker teams developing new products and services direct communication as well as boundary objects (Carlile, 2004) play an important role.

11.2.3 Enabling Governance through Team Routines

In this section, I will elaborate how in my case organizations routine capabilities are established through initial training of standardized routines adapted in context and continuously improved.

A: Initial Routine Education

Existing project management frameworks have been criticized as being too normative, assuming a similar applicability of methods in all environments (Cicmil et al., 2006). Ahlemann,

Teuteberg and Vogelsang (2009) argue that the existing project management standards are generally accepted in practice, however, the major problems are administrative overhead, high costs and a lack of acceptance among participants. At the same time frameworks do not provide sufficient guidance on the specific micro-activities to be carried out in teams.

I found very limited education of project work routines across traditional organizations. Team members if at all, receive limited project management training, such as a two day PRINCE2 course and a handover which cannot adequately cover the tacit nature of the routines. This leads to the fact that every individual and team often reinvents their ways of working. This in turn leads to huge differences in capabilities across teams and departments.

Agile methods address this by defining team level routines and their contextualization with the help of coaches and ScrumMasters (Hanly et al., 2006). It is enabled through the definition of team level routines and application of coaches when implementing agile methods (Hanly et al., 2006).

Agile methods put an emphasis on the education of routines in context. Project management and project work in general requires a lot of experience, tacit knowledge stored in our heads which we cannot always make explicit. Routines are processual knowledge, routines that want to be acquired through practice. As discussed in Chapter 10 such routines must be acquired through execution.

The importance of coaching of routines has been further recognized by the SME instrument of the European Commission (EuropeanCommission, 2014). To stimulate entrepreneurship and the creation of new startups evidence shows that coaching and mentoring leads to a better survival of these ventures independently of the funding source, compared to startups with no coaching (Cull, 2006). Entrepreneurs are activity driven and don't have the patience, time and experience to find out which knowledge they need to acquire. In contrary, coaches can deliver very contextualized knowledge on the spot (Padula, 2009; Paasivaara & Lassenius, 2011; OConnor & Duchonova, 2014).

B: Continuous Improvement

Continuous learning is considered an integral aspect of knowledge workers (Drucker, 1999). Traditional project management frameworks recommend documentation of lessons learned from past projects (Kerzner, 2013, p.790). However, in many organizations this is challenged in two ways. First, documentation of lessons learned is often left to project leaders. By the end of the project, however, project teams are already shifting focus and dismantling towards new assignments and the lessons learned often not captured at all. Second, even if lessons learned are documented they are often transferred inefficiently as there is little reflection.

I argue that continuous learning needs to be a central component of governance in knowledge worker organizations. In agile methods continuous learning is enabled through retrospectives as a reflective part of each iteration and the focus point of the ScrumMaster role

on facilitating routines and removing impediments. I reflect upon this in Chapter 4 and 10. In Chapter 4 I evaluate a self-management tool for teams to reflect on the team routines. In Chapter 10 I discuss an approach to planning and improving software engineering project based coursed based on intensive coaching and team routines.

Just like in sports agility is about exercise, it is about possessing the right routines allowing to react spontaneously reflexively to a changing situation in context. Such routines need to be acquired and exercised up-front as the acquisition of routines can be long some. When measuring the brain activity of Neymar, (Naito & Hirose, 2014) found that the Brazilian football star uses 10 percent less cerebral function compared to amateur players when performing routine activities. This reduced brain activity allows him to perform more complex tasks at once. *"Champions don't do extraordinary things, they do extraordinary things, but they do them without thinking.."*, says Tony Dungy an American football coach who made routine change an integral aspect of his coaching strategy (Duhigg, 2012).

The governing body must ensure acquisition of the respective routines each time an individual takes over a new function. Management must invest time (e.g., a contingent of hours) in the development of routines. Management must be able to recognize effective routines working well and steer accordingly. This requires well educated and experienced management staff being able to make sense of the situation in context and coaching. As exemplified by the two experiments presented in Chapter 9 and Chapter 10, even small changes to routines and their interplay with artifacts can have strong impact on effectiveness and sustainability of routine.

11.3 Substantial Management and its Governance

Based on the notion of knowledge captured in routines and boundary objects as carriers of governance in this section I will discuss four components of governance in multi-project organizations and how they help understand the challenges knowledge worker teams face in organizations in practice.

In order to understand multi-project governance in knowledge worker organizations we need to be able to understand the roles, responsibilities and rules in (1) individual project teams, and (2) the multi-project organization as such. I address this by building the framework upon my findings on three levels of analysis: the individual knowledge worker, the knowledge worker team, and the knowledge worker multi-team multi-project perspective.

Based on the findings and the frameworks of Waterman Jr et al. (1980), Kerzner (2013), and McLeod and MacDonell (2011), in Table 11.2 I present the model of *Substantial Governance* in four components embedded in a recurring governance routine through (1) Sense making, (2) Knowledge creation, and (3) Decision making.

Four components of governance:

1. *Goals and their relationship to intermediate results*: Respective goals on each level can be complimentary, conflicting, or duplicate. Conflicts need to be resolved, synergies removed and doublets removed. In knowledge worker environments it is important to continuously exchange ideas facilitated across strategy, tactics and operations.

2. *Routines and Boundary Objects*: In Machine Bureaucracies (Mintzberg, 1979) routines exercise control through rules and formal communication lines. However, such bureaucracies are counterproductive in knowledge work organizations as ideas do not emerge along formal lines within a hierarchy only. Following the Professional Bureaucracies (Mintzberg, 1979), i.e., ensuring the right routines, processes are in place together with people who have appropriate skills helps to understand how the product will most likely evolve.

3. *People, their knowledge, their individual goals and organizational structure*: People, their educational background, power, role and position within the network greatly influence governance. Morale and leadership are two topics to be more deeply investigated.

4. *Organizational culture*: Values, norms, beliefs, and governmentality are crucial drivers of agile organizations.

Further, as identified in the literature study in Chapter 2 there are two gaps in knowledge hindering our understanding of governance across multiple knowledge worker teams: (1) understanding the dynamics of knowledge work in knowledge worker project teams, (2) understanding the dynamics across different teams. I address this gap by understanding governance as a recurring sense making routine emphasizing on knowledge, its dynamics captured in iterations and contextually of actors continuously reshaping governance in a networked organization.

The components discussed enable an understanding of knowledge worker teams in context through:

1. *Emphasis on knowledge rather than information:* (1) Closer collaboration and involvement of management (seen as part of coordinating bodies) enables transformation of knowledge across organizational subunits. It addresses the emerging and not predictable nature of knowledge creation. (2) Drawing attention to the tacit nature of projects in their context and the necessity to balance explicit and tacit knowledge through direct communication and boundary objects. (3) Necessity for continuous and appropriate education of project staff.

2. *Dynamics of Actor Networks:* Addressing the dynamics of actor networks, their (1) views and (2) motives are integral.

3. *Contextuality of Actor Networks:* Views and motives are grounded in their personalities, educational backgrounds, and roles and responsibilities within the organization. Each

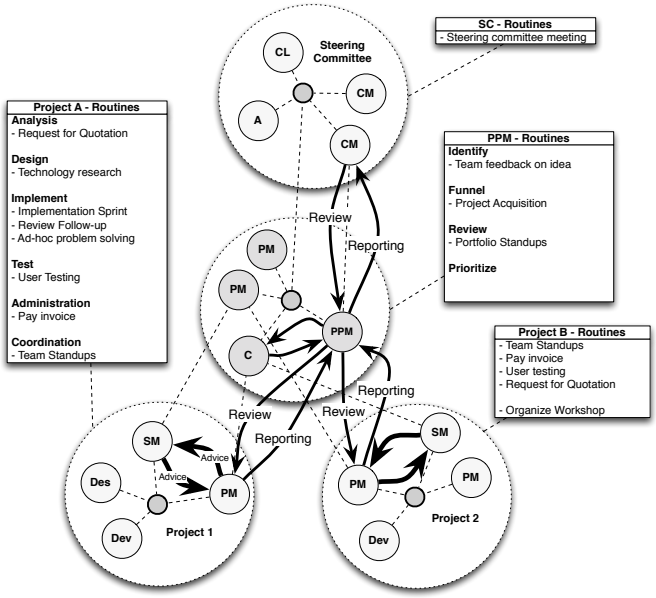


Figure 11.4: Network of Organization A, the underlying teams, including two exemplary projects out of the portfolio and involved team routines

individual is considered an asset to the knowledge worker organization and is responsible for his own productivity (Drucker, 1999). We need to acknowledge that there are different governance components to be considered in each layer and that these components interact and influence each other, thus continuously reshaping the face of governance.

Assuming human resources and their skills as given, and the organizational culture being outside the scope of this thesis I will concentrate on the product content and the routines applied.

In the following subsection I will elaborate these four components more in depth. I argue that due to the uncertainties of knowledge work the manager, in fact each knowledge worker as part of the network organization needs to be aware of their existence and the role they play in the sense making process.

11.3.1 Governance: Sense Making, Knowledge Creation and Decision Making

Too and Weaver (2013) describe governance as the "mirror of management". It specifies: (1) Definition the roles and responsibilities, as well as (2) oversight and assurance on project content and organization.

According to Drucker the knowledge worker needs to manage him/herself. In a network organization every individual is a part of that governance routine. A such governance in knowledge worker organizations rather resembles what Ahola et al. (2013) call 'project governance being internal to a specific project', thus a view on governance as continuously recreated across the actors within the project network rather than hierarchically and unidirectionally defined.

Through the continuous development of ideas management must accept that the original vision is likely to change and must be prepared to steer dynamically. When during a medical operation a dangerous heart condition is found during a lung examination, the doctor needs to be able to provide help although a heart operation was not originally envisioned. He or she then also learns the status of the project and its context which allows him to make better decisions. When a multi-project manager observes a knowledge worker team does have a lack of experience (routines) to execute a specific project effectively he must make sure to transfer the experience (e.g., by coaching the team) or appoint a subordinate/team member with that particular knowledge.

Governance becomes visible in mechanisms such as steering meetings and resulting interventions such as reassurance of iteration lengths (Talby & Dubinsky, 2009) or assigning additional teams (Cheng, Jansen & Remmers, 2009). During such steering meetings governance bodies review projects and their results, and compare those to previously set goals and the organization's strategic directions, applied routines, people involved and the organizational embedding (culture). Then decisions are made.

Table 11.2: Components of governance across layers of a project-based organization

	Project Team	**Coordinating Team**	**Governance Team**	
Goals and intermediate results	• Deliver product or service on time, on budget and on quality (Kerzner, 2013)	• Link strategy to projects • Balance projects and resources • Maximize a portfolio's value (Martinsuo & Lehtonen, 2007)	• Deciding which projects pursue • Oversight and assurance (Too & Weaver, 2013)	**Governance routine: Sense making, Knowledge creation and decision making**
Routines and Boundary Objects (Operations, Management and coordination, capability development)	• Daily stand ups	• Iteration review and Project backlogs	• Portfolio project reviews and dashboards	
People and Org. Structure (Power, Competencies, Educational background, Motivation, personality type)	• Cross-functional team facilitated by a servant leader	• multi-team org. facilitated by a coordinating program or portfolio team, led by a servant leader		
Org. Culture (Norms, Values, Beliefs)	• Shared leadership • Team Orientation • Redundancy • Learning • Autonomy	• Transparency • Collaboration • Commitment • Team Orientation		

Following Choo's concept of the *Knowing Organization* (Choo, 2006) I argue that governance in project-based organizations is exercised in a recurring routine of sense making, knowledge creation, and decision making across the components and actors as outlined in Table 11.2. This happens in a recurring routine during which governing bodies inspect the components of organization and content as outlined in Table 11.2. Following such an understanding, management must understand work on micro-level of routines. The manager must have sufficient experience to coach and supervise subordinates, and he should be accountable for constant organizational learning (e.g., appoint personnel to conduct improvements).

In the following subsection I will discuss the components of governance presented in Table 11.2 more in depth.

11.3.2 Goals and Intermediate Results

As commonly discussed in literature, project governance deals with project content as well as project organization (Too & Weaver, 2013; Ahola et al., 2013). Vision, goals and tasks want to be translated alongside strategy, tactics and operations in an organization (Hanssen & Fægri, 2008). Across my case organizations this translation generally happens across the three layers of project teams, coordination management bodies (e.g., portfolio management team) and governing bodies (e.g., top management team) as discussed in Chapter 7.

The development of new products and services cannot be planned all up front, as acknowledged across the fields of new product development (Takeuchi & Nonaka, 1986), computer science (Nerur & Balijepally, 2007), entrepreneurship and new business development (Sykes & Dunham, 1995). According to literature (Fægri, 2010) it is easier to foresee intermediate goals and the derived tasks while the longer-term goals remain more difficult to predict. New business development literature (Sykes & Dunham, 1995) recognizes that the strict adherence to a plan can lead to the failure of a business venture. Sykes and Dunham (1995) argue for more adaptable planning approaches based on cyclical reevaluation of critical assumptions, rather than plans based on these assumptions.

This is addressed in Scrum through iteration and close customer collaboration. However, in multi-project organizations more actors come into play when results, resources and knowledge needs to be coordinated across different teams. For example, the goals of each project team can be complimentary, conflicting, or duplicate. The goal of the coordinating management team is then to leverage and communicate synergies, resolve conflicts and remove duplicates - thus balancing resources, knowledge and linking projects to strategy (Martinsuo & Lehtonen, 2007). The respective goal of the governance team is to make sure that the individual goals of the coordinating team are in balance with the good of the entire organization.

Individual teams in a project organizations have different goals which can be conflicting. As visualized in the model in Figure 11.1 and in the more elaborated in the example in Figure 11.5 our case multi-project organizations embodied several encapsulated governance

layers. While the governance team provides steering to the coordinating management team, the coordinating multi-project management team provides steering to the individual project teams. As laid out in Table 11.2 each of these layers (governance, management and project teams) have different goals which want to be considered in context.

Due to potentially changing goals and understanding of such goals, governance across individual teams needs to be embedded in an iterative routine. This resembles a political (pragmatic) boundary (Carlile, 2004) as the goals of teams and individuals can be conflicting. Due to the tacit aspect of knowledge, knowledge worker teams cannot be managed on information only - Management must have an understanding of the project in context to make adequate decisions on project direction and resources. According to Carlile (2004) transformation of knowledge must be negotiated within an iterative routine in oder to cover value to be delivered, work to be scheduled, within a given capacity (resources) and backed up by sustainable finances.

Existing agile literature focuses on iterative reviews of results. Initial research points at the importance of so called "cycles of control" (cf. Vähäniitty (2012)), iterative reviews of results across teams. However, following the findings discussed in this thesis, multi-project organizations of autonomous teams, routines need to be reviewed as well.

11.3.3 Routines and Boundary Objects

In the previous sections I discuss the role of routines and boundary objects in supporting the interaction for an effective knowledge work. In Part IV of this thesis, specifically in Chapter 9 and Chapter 10, I show based on two experiments how even small changes to routines in context can have a large impact on the effectiveness of a knowledge work practice in use. This emphasizes the contextuality of project work and that team members as well as the coordinating bodies need to consider routines as carriers of governance.

Table 11.1 provides a few examples of routines and respective boundary objects such as project operations, capability development. However, I would like to emphasize that these routines are only templates for 'live' routines, which need to be contextualized in practice (Pentland & Feldman, 2008). In my case organization this contextualization is often supported by coaches. This has implications on the way we look at project management frameworks.

The example scenario of the project portfolio being reviewed only on numbers in a spreadsheet presented in the beginning of this chapter emphasizes the importance of routines and their interplay with artifacts, more specifically boundary objects (Carlile, 2004). The information given, is not sufficient for the portfolio manager to enable an appropriate sense making in context. Here, management must be able to make sense on the situation, the project environment and the outcomes. The boundary object containing only numbers (raw data) on the project status is not sufficient to convey the knowledge necessary for the portfolio team to create an adequate understanding of the respective project in context. The routine of reviewing

the project portfolio using a table of projects states reporting only in numbers is an insufficient artifact to cover the complex nature and the true status of the project.

At any time in the project management must be able to make sense of the project context, the routines applied and the project content. In that sense it is important to acknowledge and consider these team routines and their characteristics (e.g., frequency) as an important aspect of project governance. This emphasizes the importance of routines as a mechanism of governance.

In multi-team and multi-project organizations then it becomes important that the routines are compatible across the individual teams.

11.3.4 People, their Individual Goals and Organizational Structure

People, their individual goals, power, role and position within the network greatly influence effectiveness of projects pursued by an organization. Actors, their positions within the network and their power influence project content as well as effectiveness of routines. Literature (cf. Drucker (1999)) argues that projects need to be carefully considered before being assigned to a particular knowledge worker due to own interest of the individual. In portfolio management, for example, metrics affect budget allocation. This can pose a threat to power of bodies holding the respective budget, as it influences the decisions where the specific actor previously used his preferences (Wang, 2000). Also, looking back at our reporting example, in bottom-up reporting, specially when things do go not as planned, operational staff can take advantage of the ignorance of management bodies, tweaking the report.

Knowledge workers need sufficient formal education in order to complete their tasks, their educational background as well as mental capabilities do influence their views on the world.

A: Considerations on Organizational Structure

Some scholars argue in favor of seeing projects as temporary organizations, such are (re)configured every time a new project is started (Turner & Müller, 2003). In many traditional matrix-like organizational forms an individual can be part of multiple projects across multiple teams at the same time. This is in contrast with our agile case organizations were projects are allocated to preferably stable teams - team members find this allocation to multiple projects and teams distracting.

O'leary, Mortensen and Woolley (2011) elaborate the matter of multiple team membership discussing how organizations are challenged trying to balance available resources and individual and team productivity and learning. There is little research on multi team membership (O'leary et al., 2011), however, the general consensus is that it should not be more than 2-3 projects in parallel. O'leary et al. (2011) point out that a careful balance of number and variety of assignments in, multiple teams memberships can enhance both productivity and learning.

However, these positive effects need to be put into perspective with high costs of fragmented attention and coordination overhead.

Some streams of literature (Kettunen & Laanti, 2008) observe a trend away from big organizations towards collaboration across small entrepreneurial more autonomous companies. Agile teams are much more autonomous compared to teams continuously recreated in traditional project matrix organizations. Due to the increased autonomy of such organizations, there is a redistribution of power. As such, seeing project governance as internal to each project is more appropriate. Command and control structures are less likely in such organizational form and governance is more likely to be under continuous (re)negotiation. Management needs to be aware of this shift.

This requires a balance of autonomy and the need for control across the coordinating bodies. Self-organization of the team to foster innovation and sensitivity to local conditions. Some authors recommend a co-creation of "what", while leaving the specification of the "how" to the team (Lindkvist, 2004). Weick, Sutcliffe and Obstfeld (2008) call this the "underspecification of structure". As the nature of the product or service is more uncertain in knowledge worker projects, management needs to balance the uncertainty by watching the factors influencing project outcomes more closely and steer accordingly. Management must make sense of the project environment and output in context.

While the project is perceived as something dynamic and temporary, our participants state their preference towards working in stable teams as such becomes less prominent. While the project is always a temporary assignment, a routine, the organizational structure is often team-based. Following the argument that organizations are moving towards networks of small entrepreneurial companies (Kettunen & Laanti, 2008), the teams can be relatively stable.

Especially for effective agile organizations team members need to develop the capabilities to work effectively as a team. It takes time to develop the capabilities as discussed in Chapter 4. As it requires considerable effort our participants prefer to keep a high-performing team stable, also for reasons of resource allocation. On one hand as it takes an investment to pass the stages of team development (Tuckman & Jensen, 1977), on the other hand to develop the routines and mental model of an effective team as described in Chapter 4. Every time a team works together for the first time these capabilities have to be developed anew. Uncertainties have to be moved aside and team members have to bring each other up to speed.

I argue that one must differentiate knowledge worker projects. Across our cases the project is perceived as an organizational form in the sense of a team based structure. I argue that the structural focus should be on teams rather than on projects, as according to our findings (compare Chapter 7) teams are more stable units while projects are unique and temporal assignments.

Further our participants have indicated that team orientation should not only be applied to the operational teams, but also on the level of portfolio management and top management teams as depicted in Figure 11.4.

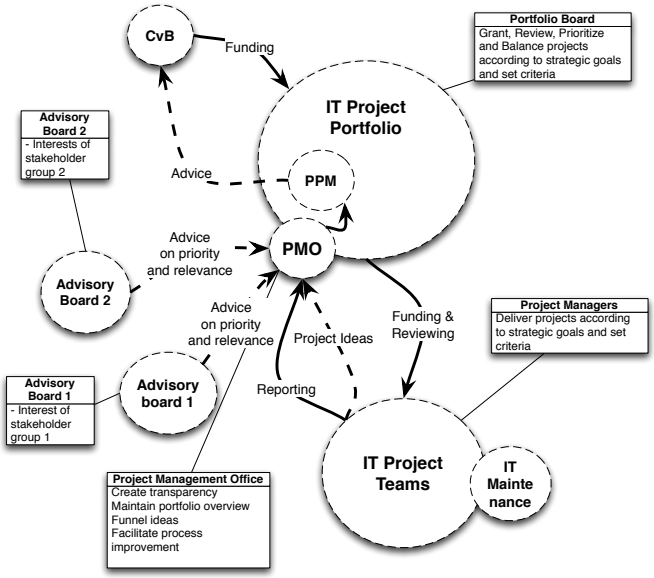


Figure 11.5: Example of a multi-project organization

11.3.5 Organizational Culture

As discussed by Robinson and Sharp (2003) routines (agile practices) and organizational values (agile principles) are complimentary and leverage one another. The use of user stories facilitates collaboration, but, if a collaborative mindset is not in place, this will most likely hinder if not prevent collaboration. The use of portfolio dashboards can promote transparency and improve decision making, but, if a mindset is not in place that promotes transparency, portfolio dashboards can hinder if not prevent justified decision making.

12 Conclusions

In this chapter I present the conclusions of my research. In Section 12.1 I address the research questions and problem statement. In Section 12.2 I highlight the contributions claimed. In Section 12.3 I summarize the concept of Substantial Management by eight propositions to be elaborated in the form of a quantitative study. In Section 12.4 I present the implications for society (e.g., governing bodies, managers, and policy makers). Further, in Section 12.5 I present two directions for future research.

The study is motivated by the aspiration to understand project governance in organizations pursuing the development of new products and services across multiple knowledge worker teams. In particular, the study is guided by the following problem statement: *Is governance of innovation project management necessary or neglected?*

In total, I have devised eight studies, each consisting of a rigorous methodology and contributing to the overarching problem statement, discussing knowledge workers and their embedding in organizations across three levels of analysis. The findings are based on data from a total of 53 knowledge worker teams across 44 organizations in 8 countries, about 3800 minutes of recorded material enriched by four quantitative surveys and many hours of observations. According to these observations the manager has been depicted and advised.

The empirical findings presented in this study show that knowledge worker teams and the complex nature of their tasks requires a different approach to governance in multi-team organizations. As argued by Piore and Sabel (1984) many of our organizations are aligned towards the mass production of goods and the control of manual workers. Moreover, by following the notion of Drucker (1994), we see that in our increasingly knowledge-based economy, however, knowledge workers contribute to the creation of economic value by applying their theoretical knowledge and analytical skills. Thus, the knowledge worker is

opposing *Scientific Management* (F. W. Taylor, 1911). Moreover, the manual worker is a by definition preferably ignorant workforce solely executing orders issued by management staff. I argue that the knowledge worker (see Drucker (1994)) as part of an intelligent, problem-solving workforce implies a new role for the management. The manager in innovation projects observes and senses the emerging product or service. He understands projects in context similarly to how Nonaka and Takeuchi (1995) described the role as that of a *knowledge engineer*.

The two essential conclusions that can be drawn from this study are: (1) we need a fundamentally different way to govern innovation projects developing new products and services, and (2) that governance especially in heavily top-down influenced fields such as public administration can benefit from advancements in agile methods.

12.1 Answering the Research Questions and the Problem Statement

In this section I present a summary of the answers to the research questions and the problem statement. The PS and RQs are defined in Section 1.2. For clarity of reading they are reproduced below.

- PS: *Is governance of innovation project management necessary or neglected?*
- RQ1: *How can governing bodies in organizations of knowledge worker teams understand and steer multiple knowledge worker project teams in practice?*
- RQ2: *What are the components of governance necessary to understand knowledge worker project organizations?*
- RQ3: *How can governance address the dynamics of knowledge work across multiple knowledge worker teams?*

12.1.1 Understanding Knowledge Worker Project Teams in Practice

- RQ1: *How can governing bodies in organizations of knowledge worker teams understand and steer multiple knowledge worker project teams in practice?*

Optimization of approaches to the work of knowledge workers has been mentioned as one of the biggest tasks for the 21st century (see, e.g., Drucker (1999); Davenport (2013)). According to Drucker (1994) knowledge workers differ fundamentally in their approach to work, in their mind-set, and their habit of continuous learning. They show fundamental differences to manual labor workers and assembly line workers.

Table 12.1: Answers to the research questions

	Research question	**Finding**	**In this thesis**
RQ1	*How can governing bodies in organizations of knowledge worker teams understand and steer multiple knowledge worker project teams in practice?*	Through the perspective of Organizational Routines with emphasis an on (1) knowledge, (2) dynamics, and (3) context	Chapter 3, 4, 7, and 11
RQ2	*What are the components of governance necessary to understand knowledge worker project organizations?*	The components are: (1) goals and their relationship to intermediate results, (2) routines and boundary objects, (3) people, their knowledge, their individual goals, and organizational structure, and (4) institutional context	Chapter 4, 5, 7, 8, 9, 10, and 11
RQ3	*How can governance address the dynamics of knowledge work across multiple knowledge worker teams?*	Due to the evolving nature of knowledge, governance needs to be embedded in an iterative governance routine of (1) sense making, (2) knowledge creation, and (3) decision making	Chapter 6, 8, 10, and 11

My study contributes to the understanding of the properties of knowledge worker project organizations. In Chapter 3 I put forward the recommendations provided by executive staff in knowledge worker organizations. In Chapter 4 I discussed five dimensions of agile teamwork. In Chapter 7 I discussed the characteristics of agile knowledge project worker organizations.

It should be noted that this list does not aim to be complete, but is intended to show the main properties. In particular, I would like to highlight three properties, e.g., knowledge, dynamics, and context.

1. *Emphasis on knowledge rather than information:* (1) Closer collaboration and involvement of management (seen as part of coordinating bodies) enables transformation of knowledge across organizational subunits. It addresses the emerging and not predictable nature of knowledge creation. (2) Drawing attention to the tacit nature of projects in their context and the necessity to balance explicit and tacit knowledge through direct communication and boundary objects. (3) Necessity for continuous and appropriate education of project staff.

2. *Dynamics of Actor Networks:* Addressing the dynamics of actor networks, their (1) views and (2) motives are integral.

3. *Contextuality of Actor Networks:* Views and motives are grounded in their personalities, educational backgrounds, and roles and responsibilities within the organization. Each individual is considered an asset to the knowledge worker organization and is responsible for his own productivity (Drucker, 1999). We need to acknowledge that there are different governance components to be considered in each layer and that these components interact and influence each other, thus continuously reshaping the face of governance.

12.1.2 Components of Governance

- RQ2: *What are the components of governance necessary to understand knowledge worker project organizations?*

In Section 11.3 I elaborated the four identified components of governance based on the findings and the frameworks of Waterman Jr et al. (1980), Kerzner (2013), and McLeod and MacDonell (2011), in Table 11.2.

In Chapter 7 I pointed out (1) routines, (2) organizational structure, and (3) values, as integral components of governance in knowledge worker project organizations. In Chapter 11, specifically in Section 11.1.2, I reflected on goals and their relationship to intermediate results as components of governance. In Chapter 8 I discussed the relation of routines and artifacts as components of governance. In Chapter 9 I discussed the impact that (1) routines, (2) boundary objects, as well as the (3) concrete formalisms that these artifacts have on the sustainability and effectiveness of a routine. In Chapter 10 I discussed the relation of routines and intensive

coaching as components of governance. In Chapter 4 and 7 I discussed the components of governance linked to organizational culture.

Based on the findings and existing literature I may conclude that the components are: (1) Goals and their relationship to intermediate results, (2) Routines and Boundary Objects, (3) People, their knowledge, their individual goals and organizational structure, and (4) Organizational culture.

1. *Goals and their relationship to intermediate results*: Respective goals on each level can be complimentary, conflicting, or duplicate. Conflicts need to be resolved, synergies removed and doublets removed. In knowledge worker environments it is important to continuously exchange ideas facilitated across strategy, tactics and operations.

2. *Routines and Boundary Objects*: In Machine Bureaucracies (Mintzberg, 1979) routines exercise control through rules and formal communication lines. However, such bureaucracies are counterproductive in knowledge work organizations as ideas do not emerge along formal lines within a hierarchy only. Following the Professional Bureaucracies (Mintzberg, 1979), i.e., ensuring the right routines, processes are in place together with people who have appropriate skills helps to understand how the product will most likely evolve.

3. *People, their knowledge, their individual goals and organizational structure*: People, their educational background, power, role and position within the network greatly influence governance. Morale and leadership are two topics to be more deeply investigated.

4. *Organizational culture*: Values, norms, beliefs, and governmentality are crucial drivers of agile organizations.

12.1.3 Addressing Dynamics of Knowledge Work

- RQ3: *How can governance address the dynamics of knowledge work across multiple knowledge worker teams?*

In Chapter 11, specifically in Section 11.1.3, I connected the perspectives on the role of management in knowledge intensive organizations as discussed by previous authors (cf. Drucker (1999) and Nonaka and Takeuchi (1995)) to the project management domain.

In Chapter 6 I employed the example of a highly agile medical procedure to contribute to understanding of collaboration and adaptation of unforeseen variations in enterprise process models. In Chapter 10 I put forward that governance in knowledge worker team organizations can be enabled through intensive coaching of team routines. In Chapter 8 I pointed out that the context sensitivity of projects and unpredictability of knowledge required an iterative governance routine.

Based on my research findings as well as on digested literature, I may conclude that role of the manager in team based project organizations is that of a coach on process and content. As he understands the project in context, governance is a sensemaking routine of: (1) sense making, (2) knowledge creation, and (3) decision making.

12.1.4 Addressing the Problem Statement

- PS: *Is governance of innovation project management necessary or neglected?*

With reference to the answers of the three research questions I may conclude that governance of innovation project management is necessary *and* neglected.

With the roots of the management function grounded in the mass production of goods (cf. Piore and Sabel (1984)) many large organizations found today have been, consciously or not, designed around the ideas developed by Taylorism and optimized towards the control of manual workers. As emerging ideas cannot be predicted in the same way as we usually predict the output of manual labour teams, knowledge worker teams developing new products and services require a fundamentally different approach to project governance. Such an approach needs to be based on operational autonomy of teams and dynamic governance based on routines.

12.2 Claimed Contributions

To address my problem statement I take the example of how agile teams deal with the higher degree of interaction necessary in knowledge work projects. Based on the research findings across these pre-published studies I develop the concept of *Substantial Management* grounded in empirical work presented across three levels of analysis.

1. The individual knowledge worker - presented in Part II
2. The knowledge worker project team - presented in Part III
3. The multi-project organization - presented in Part IV

In Part II I discussed how an increasing number of relationships to colleagues, co-workers, clients and supervisors as well as higher dynamics are perceived by board level executives. In Part III I discussed the characteristics of agile knowledge worker teams, the way they adjust their ways of working, and how they document their project knowledge internally. In Part IV I discussed how agile methods applied across knowledge worker teams accommodate a higher rate of interaction and what implications it has on routines, structures and values in existing multi-project organizations.

In summary, I claim to add to the body of knowledge on project governance through the following four contributions.

1. Based on the empirical findings across the three levels of analysis, I identified the characteristics and components of governance in knowledge worker team organizations

2. I showed the *impact of routines and boundary objects* on the sustainability and effectiveness of a routine.

3. I introduced an *agile governance framework* addressing the dynamics of knowledge work in organizations developing new products and services. The framework connects project governance to modern management theory. It acknowledges the different characteristics of knowledge workers opposing traditional project management frameworks which assume a predictability of manual work.

4. I introduced the concept of *Substantial Management*, combining different streams of management literature based on eight propositions to be tested in future research.

Moreover, I argue that my analysis contributes to project management literature and provides an understanding of project governance different from the traditional view originating in civil engineering, construction and the ideas of the Tayloristic age and mass production.

12.3 Substantial Management

In order to underpin my contributions, I will now reflect upon the findings and present them by means of propositions to be tested in future research.

The empirical findings presented in this study show that knowledge worker teams and the complex nature of their tasks requires a different approach to governance. Opposing *Scientific Management* (F. W. Taylor, 1911) and the manual workers, by definition preferably ignorant workers of the industrial age, I argue that *Substantial Management* is necessary to deal with uncertainties when developing new products and services.

The emerging nature of knowledge worker projects is subject to many external factors, since the projects cannot be governed on input and output only. As Drucker (1999) elaborates, knowledge workers are responsible for their own productivity. As such they have to manage themselves and need autonomy to do so. As the outcomes of knowledge worker projects are difficult to predict and to plan up-front, knowledge worker teams benefit from autonomy to enable decision making based on local knowledge.

This is rooted in the fact that the configuration of each team (Chapter 4), the ways the teams develop and transfer project results (Chapter 8), as well as the way they adapt their ways of working (Chapter 6) are very context dependent. Moreover, they are continuously reshaped, i.e., the teams need local decision authority to function effectively. Due to the emerging nature of the results, the teams across the analyzed cases largely profited from local information and the given degree of autonomy which allowed them to make better

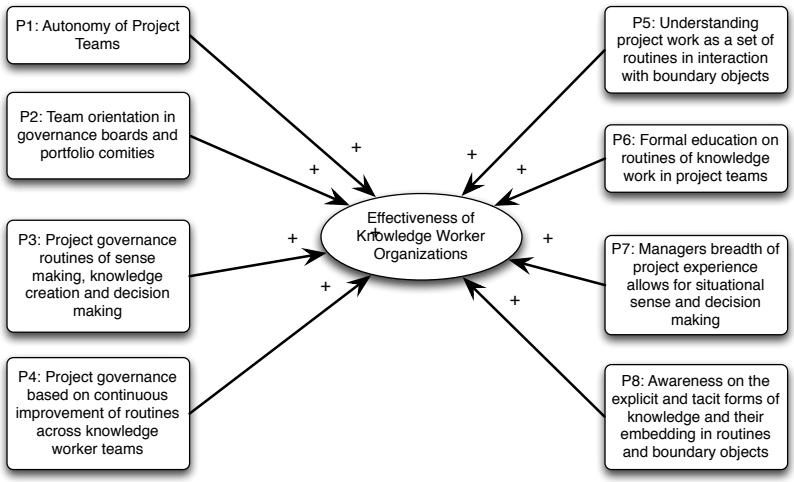


Figure 12.1: Propositions for Substantial Management

local decisions. As we learned in Chapters 4 and 7 team autonomy is reported to improve ownership of the project and enables better decision making based on local information. It is a characteristic that is acknowledged at team perspective as well as at the multi-team perspective.

Based on the rooted facts I develop the following eight propositions.

> *Proposition 1: Autonomy of agile knowledge worker project teams positively influences the effectiveness of knowledge worker organizations*

Team orientation is an integral aspect of agile knowledge worker organizations (compare Chapter 4). Obviously, project operations are often conducted in teams. However, in order to promote commitment towards common goals also governance boards and portfolio committees should be seen as teams.

> *Proposition 2: Team orientation on the level of project teams, governance boards, and portfolio committees positively influences the effectiveness of knowledge worker organizations.*

Across the cases I found that project governance is divided into (1) the governance of the project content, and (2) the project organization as elaborated in Chapter 11.

In order to balance the unpredictability of the results and the increased autonomy of the teams with the need for control, the manager (as a knowledge and task coordinating body

across project teams) watches the factors influencing project outcomes throughout the project, particularly the product and service under development and the routines applied.

As content and context are emerging continuously, management coordinates knowledge and facilitates its transformation across the domains of practice. Carlile (2004) argues that as such a translation happens across multiple domains where individuals need to be willing to change their own views (e.g., transforming strategic choices to concrete project ideas across business, marketing, sales, IT), it must be embedded in an iterative routine.

Studying project handovers in Chapter 8 I discussed how knowledge worker teams created and transfered project results across different teams using iterative routines. In Chapter 7 I discussed (1) the implications of such iterative routines in existing multi-project organizations and (2) how they increased the frequency of interactions across the domains of practice.

Based on this I developed the third proposition.

> *Proposition 3: Project governance routines of sense making, knowledge creation, and decision making across project content and project organization positively influences the effectiveness of knowledge worker organizations.*

Knowledge workers need autonomy regarding the execution of their tasks. However, following Mintzberg (1979) they can be governed through routines and their standardization. Agile methods support such a standardization through their definition of team routines which provide structure to projects as frequently observed across my cases. This is as confirmed in the sequencing study of Thummadi et al. (2011). Routines and boundary objects are carriers of governance.

> *Proposition 4: Project governance based on continuous improvement of routines across knowledge worker teams positively influences the effectiveness of knowledge worker organizations.*

Although project management is often understood in terms of high-level frameworks such as PRINCE2 or Scrum, the difference in the respective implementations in practice is so large that research calls for (1) a more dynamic elaboration of the actual project work and (2) the complex social processes it is based on (Cicmil et al., 2006). Following the contributions in the field of agile software development (Thummadi et al., 2011) I thus argue that we need to look at concrete ”as-is performative practices” and need to understand project management as a set of organizational routines (Feldman & Pentland, 2003). In order to understand project governance one needs to consider project management as a set of organizational routines.

In order to illustrate the impact of routines on the effectiveness of an organization and their interplay with boundary objects in Chapter 9 and Chapter 10 I discussed the examples of two

concrete routine experiments. I elaborated how slight adjustments in their execution can have a large impact on the effectiveness and sustainability of a practice.

In practice project managers think in terms of concrete routines that are applied rather than a preselected set of practices prescribed by project management frameworks. Project management frameworks need to support routines in interaction with boundary objects adapted by the team members in specific project context.

Moreover, I will demonstrate that the routines applied are very sensitive to context.

> *Proposition 5: Understanding project work as a set of routines in interaction with boundary objects positively influences the effectiveness of knowledge worker organizations.*

Knowledge workers need a high degree of formal education to perform and need to have the right skills to do so. However, across my case organizations I found little awareness of the characteristics of knowledge work and how to handle it. Knowledge workers need to increasingly manage themselves and need education to be able to do so. University programs, while providing formal education need to provide better academic reflection on the routines of knowledge work in project teams.

> *Proposition 6: Formal education on routines of knowledge work in project teams positively influences the effectiveness of knowledge worker organizations."*

Just like the product or service under development, the development practice itself, i.e., the routines we apply to create project outcomes, are part of our mental model. They are tactic and intangible knowledge, very contextual and following Nonaka and Takeuchi (1995) difficult to communicate. As routines are tactic knowledge which can only be acquired through direct interaction (e.g., with other individuals and artifacts), a manager in knowledge worker organizations needs to have sufficient experience to make sense of the project in context to give appropriate advice and steer. Based on this I developed the following proposition.

> *Proposition 7: In knowledge worker organizations the manager is a coach. As tacit knowledge is best transferred via direct interaction among the participants, managers need to have a broad experience in order to make sense of the situation in context and make adequate decisions. The breadth of a manager's project experience allows for situational sense and decision making, and positively influences the effectiveness of knowledge worker organizations.*

As routines are tactic knowledge which can only be acquired through direct interaction (e.g., with other individuals and boundary objects), a manager in knowledge worker organizations needs to have sufficient experience to make sense of the project in context to give appropriate advice and steering.

In bureaucracies documents have been often abused for control, but viewing them as boundary objects raises awareness that we need to understand project reality as something dynamic and continuously adapted in context. As discussed in Chapter 8 transfer of project knowledge across individual knowledge workers and teams using explicit documentation only can be uneconomic and impossible at times. Governing bodies in knowledge worker organizations thus should make explicit choices what information (explicit knowledge) can be stored in formal documents (e.g., for reasons of accountability like contracts) and what knowledge in its more volatile form can and should rather be transferred directly across knowledge workers with the help of boundary objects (e.g., volatile knowledge on product or service under development, routines that accompany a service). Based on this I developed the following proposition.

> *Proposition 8: Rising awareness on the explicit and tacit forms of knowledge and their embedding in routines and boundary objects positively influences the effectiveness of knowledge worker organizations*

12.4 Implications for Society

Practitioners such as governing and managing bodies, but in fact every knowledge worker can benefit from this study. Drucker (1999) calls the improvement of knowledge worker productivity a *"survival requirement"* for the developed countries.

To governing bodies which want to setup effective organizations developing new products and services I add a governance framework addressing the dynamics of knowledge work consisting of components to understand governance in contexts of Project-Based Organizations. As pointed out in my analysis Understanding projects in terms of high-level frameworks (PRINCE2, Agile) is not sufficient to cover the contextuality of projects. Small changes to routines and their interplay with artifacts can have strong impact on effectiveness and sustainability of routines. Management must understand work on micro-level of routines and make sure project staff is equipped with the right routines (skills and capabilities) to actually execute their work.

To managers it provides a perspective on their activities outside the content-free view discussed by classic management literature. In the knowledge society the manager plays an important role in the knowledge creation process. Following Polanyi (1967) and other contemporary authors (cf. (Nonaka & Takeuchi, 1995; Choo, 2006)) there are two types of knowledge:

explicit and tacit. Tacit knowledge, such as processual knowledge can only be acquired in the process. In order to enable this, management must have sufficient experience to coach and supervise subordinates. Management must be accountable for continuous organizational learning (e.g., appoint personnel to conduct improvements). Management must be able to recognize effective routines working well and steer accordingly

To policy makers my research provides the key thesis that the way we prepare knowledge workers for the labour market is not sufficient. While universities provide a lot of formal knowledge necessary, knowledge workers are not sufficiently prepared to deliver their tasks.

12.5 Future Research Directions

The developed governance framework and its anchorage in routines opens at least two directions for future research. First, the perspective on project work as a set of routines enables an in-depth view on project management as a practice-in-use. Second, accepting knowledge worker projects changes the way we see projects in context. This has implications on other domains such as public administration, law, economics and political science.

Researching project work as a set of routines: How to choose the right routines in practice? Medical decision support systems could provide initial advice, but how can such support be applied in project teams? Routines are notoriously difficult to capture and document as described in Chapter 6. What are effective and efficient means to capture those routines sufficiently?

Researching project governance: What is the right balance of tacit knowledge and explicitly documented knowledge in organizations? What is an appropriate number of teams, in which knowledge workers can be part of, while maintaining an effective work routine. An iterative approach to governance (based on value projects deliver in context) has implications for project tenders in public organizations. So, the question remains what are the precise implications for policy makers, economists, and public servants? Whatever the case, in my opinion, Governance of Innovation Project Management is necessary and may not be neglected.

References

Abrahamsson, P., Conboy, K. & Wang, X. (2009). lots done, more to do: the current state of agile systems development research.

Abrahamsson, P., Warsta, J., Siponen, M. T. & Ronkainen, J. (2003). New directions on agile methods: a comparative analysis. In *Proceedings of the 25th international conference on software engineering* (pp. 244–254). Washington, DC, USA: IEEE Computer Society.

Ågerfalk, J., Fitzgerald, B. & In, O. P. (2006). Flexible and distributed software processes: old petunias in new bowls. In *Communications of the acm.*

Aguiar, A. & David, G. (2005). Wikiwiki weaving heterogeneous software artifacts. In *Proceedings of the 2005 international symposium on wikis* (pp. 67–74). New York, NY, USA: ACM.

Ahlemann, F., Teuteberg, F. & Vogelsang, K. (2009). Project management standards–diffusion and application in germany and switzerland. *International Journal of Project Management*, *27*(3), 292–303.

Ahola, T., Ruuska, I., Artto, K. & Kujala, J. (2013). What is project governance and what are its origins? *International Journal of Project Management*.

Allen, T. J. & Henn, G. (2006). *The Organization and Architecture of Innovation: Managing the Flow of Technology*. Butterworth-Heinemann.

Anderson, J. R. (1982). Acquisition of cognitive skill. *Psychological Review*, *89*, 369–406.

Andova, S., Groenewegen, L. & de Vink, E. P. (2011). Dynamic consistency in process algebra: From paradigm to acp. *Science of Computer Programming*, *76*(8), 711–735.

Andova, S., Groenewegen, L. P. J., Stafleu, J. & de Vink, E. P. (2009). Formalizing adaptation on-the-fly. *Electronic Notes in Theoretical Computer Science*, *255*, 23–44. doi: http://dx.doi.org/10.1016/j.entcs.2009.10.023

Arakawa, S. & Yukita, S. (2006). An effective agile teaching environment for java programming courses. *Frontiers in Education, Annual*, *0*, 13-18. doi: http://doi.ieeecomputersociety.org/10.1109/FIE.2006.322534

Aubry, M., Hobbs, B. & Thuillier, D. (2007). A new framework for understanding organisational project management through the PMO. *International Journal of Project Management*, *25*(4), 328–336.

Balogun, J. & Johnson, G. (2004). Organizational restructuring and middle manager sensemaking. *Academy of Management Journal*, *47*(4), 523–549.

Barzilai-Nahon, K. & Mason, R. M. (2010). How executives perceive the net generation. *Information, Communication & Society*, *13*(3), 396–418.

Bass, J. M. (2014). How product owner teams scale agile methods to large distributed enterprises. *Empirical Software Engineering*, 1–33.

Beck, K. & Andres, C. (2004). *Extreme programming explained: embrace change*. Addison-Wesley Professional.

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., ... others (2001). The agile manifesto. *http://www. agilemanifesto. org/principles. html. Acesso em*, *7*(08), 2009.

Becker, M. C. (2004, 1st August). Organizational routines: a review of the literature. *Ind Corp Change*, *13*(4), 643–678.

Benefield, G. (2008). Rolling out agile in a large enterprise. In *Hawaii international conference on system sciences, proceedings of the 41st annual* (pp. 461–461).

Benschop/ANP, L. (2014). *Overheid verspilt jaarlijks 1 tot 5 miljard door ICT-projecten.* `http://www.nu.nl/politiek/3903779/overheid-verspilt-jaarlijks-1-5-miljard-ict-projecten.html"`. ([Online; posted 15-October-2014])

Blichfeldt, B. S. & Eskerod, P. (2008). Project portfolio management–theres more to it than what management enacts. *International Journal of Project Management*, *26*(4), 357–365.

Blomquist, T. & Müller, R. (2006). Practices, roles, and responsibilities of middle managers in program and portfolio management. *Project Management Journal*, *37*(1), 52–66.

Boehm, B. & Turner, R. (2003). *Balancing agility and discipline: A guide for the perplexed*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

Boehm, B. & Turner, R. (2005). Management challenges to implementing agile processes in traditional development organizations. *Software, IEEE*, *22*(5), 30–39.

Booth, C. & Harmer, M. (1994). Agile manufacturing concepts and opportunities in ceramics. *Ceram. Trans.*, *50*, 67–76.

Boxwala, A. A., Peleg, M., Tu, S., Ogunyemi, O., Zeng, Q. T., Wang, D., ... Shortliffe, E. H. (2004). Glif3: a representation format for sharable computer-interpretable clinical practice guidelines. *Journal of biomedical informatics*, *37*(3), 147–161.

Briand, L. C., Labiche, Y., Penta, M. D. & Yan-Bondoc, H. D. (2005). An experimental investigation of formality in UML-based development. *IEEE Trans. Software Eng*, *31*(10), 833–849. doi: 10.1109/TSE.2005.105

Carlile, P. R. (2002, July). A pragmatic view of knowledge and boundaries: Boundary objects in new product development. *Organization Science*, *13*, 442–455.

Carlile, P. R. (2004, October). Transferring, translating, and transforming: An integrative framework for managing knowledge across boundaries. *Organization Science*, *15*(5), 555–568. Retrieved from `http://dx.doi.org/10.1287/orsc.1040.0094` doi: 10.1287/orsc.1040.0094

Chao, J. (2005). Balancing hands-on and research activities: A graduate level agile software development course. In *Proceedings of the agile development conference* (pp. 306–311). Washington, DC, USA: IEEE Computer Society.

Cheng, T.-H., Jansen, S. & Remmers, M. (2009). Controlling and monitoring agile software development in three dutch product software companies. In *Proceedings of the 2009*

icse workshop on software development governance (pp. 29–35).

Choo, C. W. (2006). *The knowing organization: How organizations use information to construct meaning, create knowledge, and make decisions*. Oxford university press New York.

Chow, T. & Cao, D.-B. (2008, June). A survey study of critical success factors in agile software projects. *J. Syst. Softw.*, *81*(6), 961–971.

Cicmil, S., Williams, T., Thomas, J. & Hodgson, D. (2006). Rethinking project management: researching the actuality of projects. *International Journal of Project Management*, *24*(8), 675–686.

Clear, T. (2003). Documentation and agile methods: striking a balance. *SIGCSE Bulletin*, *35*(2), 12-13.

Clegg, S. R., Pitsis, T. S., Rura-Polley, T. & Marosszeky, M. (2002). Governmentality matters: designing an alliance culture of inter-organizational collaboration for managing projects. *Organization Studies*, *23*(3), 317–337.

Cockburn, A. & Highsmith, J. (2001, November). Agile software development: The people factor. *Computer*, *34*, 131–133.

Cohen, M. D., Burkhart, R., Dosi, G., Egidi, M., Marengo, L., Warglien, M. & Winter, S. (1996, 1st January). Routines and Other Recurring Action Patterns of Organizations: Contemporary Research Issues. *Ind Corp Change*, *5*(3), 653–698. doi: 10.1093/icc/5.3.653

Conboy, K. (2009, September). Agility from first principles: Reconstructing the concept of agility in information systems development. *Info. Sys. Research*, *20*(3), 329–354.

Cooper, R. G., Edgett, S. J. & Kleinschmidt, E. J. (1999). New product portfolio management: Practices and performance. *Journal of Product Innovation Management*, *16*(4), 333–351.

Cull, J. (2006). Mentoring young entrepreneurs: What leads to success? *International journal of evidence based coaching and mentoring*, *4*(2), 8–18.

D'Adderio, L. (2011). Artifacts at the centre of routines: Performing the material turn in routines theory. *Journal of Institutional Economics*, *7*(02), 197–230.

Davenport, T. H. (2013). *Thinking for a living: how to get better performances and results from knowledge workers*. Harvard Business Press.

DeGrace, P. & Stahl, L. (1990). *Wicked problems, righteous solutions*. Yourdon Press Upper Saddle River, NJ, USA.

De Grazia, V. (2009). *Irresistible empire: America's advance through twentieth-century europe*. Harvard University Press.

de Souza, S. C. B., Anquetil, N. & de Oliveira, K. M. (2005). A study of the documentation essential to software maintenance. In *Proceedings of the 23rd annual international conference on design of communication* (pp. 68–75). New York, NY, USA: ACM.

Digman, J. (1990). Personality structure: Emergence of the five-factor model. *Annual Review*

of Psychology, *41*, 417–440.

Dingsøyr, T. & Hanssen, G. K. (2002). Extending agile methods: Postmortem reviews as extended feedback. In *4th international workshop on learning software organizations* (pp. 4–12).

Dixon-Woods, M., Agarwal, S., Jones, D., Young, B. & Sutton, A. (2005). Synthesising qualitative and quantitative evidence: a review of possible methods. *Journal of health services research & policy*, *10*(1), 45–53B.

Drucker, P. (1977). *People and performance*. Routledge.

Drucker, P. (1993). *Post-capitalist society*. Routledge.

Drucker, P. (1994). The age of social transformation. *Atlantic Monthly*, *274*(5), 53–70.

Drucker, P. (1999). Knowledge-worker productivity: The biggest challenge. *The knowledge management yearbook 2000-2001*.

Dugan, R. F. (2011). A survey of computer science capstone course literature. *Computer Science Education*, *21*(3), 201-267.

Duhigg, C. (2012). *The power of habit: why we do what we do in life and business*. Random House LLC.

Dutta, S. & van Wassenhove, L. N. (1996). *Report on the 1995/1996 software excellence survey* (Working Paper No. 96/52/TM). Boulevard de Constance, 77305 Fontainebleau, France: INSEAD.

Dybå, T. & Dingsøyr, T. (2008a). Empirical studies of agile software development: A systematic review. *Information Software Technology*, *50*(9-10), 833–859.

Dybå, T. & Dingsøyr, T. (2008b, August). Empirical studies of agile software development: A systematic review. *Information Software Technology*, *50*, 833–859. doi: 10.1016/j.infsof.2008.01.006

Dybå, T., Dingsøyr, T. & Moe, N. B. (2014). Agile project management. In *Software project management in a changing world* (pp. 277–300). Springer.

Dybå, T. & Moe, N. B. (1999). Rethinking the concept of software process assessment. In *Proceedings of european software process improvement conference (eurospi 1999).* Pori, Finland.

Dye, L. & Pennypacker, J. (2000). Project portfolio managing and managing multiple projects: Two sides of the same coin? In *Proceedings of the 2000 pmi seminars & symposium. pmi* (Vol. 321).

Emery, F. E., Thorsrud, E., Engelstad, P. H., Gulowsen, J. & Qale, T. (1976). *Democracy at work: The report of the norwegian industrial democracy program*. Martinus Nijhoff Social Sciences Division Leiden.

Engwall, M. & Jerbrant, A. (2003). The resource allocation syndrome: the prime challenge of multi-project management? *International journal of project management*, *21*(6), 403–409.

Eriksson, T. & Ortega, J. (2006). The adoption of job rotaton: Testing the theories. *Industrial and labor relations review*, 653–666.

Eskelinen, A. (2012). *Agile transformation: What to do with managers?* Retrieved from `http://conferences.agilealliance.org/sessions/13957`

EuropeanCommission. (2014). *Horizon 2020: Dedicated SME Instrument Work Programme 2014-2015.* `http://ec.europa.eu/research/sme-techweb/pdf/SME%20Instrument%20in%20WP%202014-2015.pdf`. ([Online; accessed 23-February-2015])

Fægri, T. E. (2010). Adoption of team estimation in a specialist organizational environment. In *Agile processes in software engineering and extreme programming* (pp. 28–42). Springer Berlin Heidelberg.

Fægri, T. E., Dybå, T. & Dingsøyr, T. (2010, October). Introducing knowledge redundancy practice in software development: Experiences with job rotation in support work. *Inf. Softw. Technol.*, *52*, 1118–1132.

Feldman, M. S. & Pentland, B. T. (2003). Reconceptualizing organizational routines as a source of flexibility and change. *Administrative Science Quarterly*, *48*(1), 94–118.

Fernandez, R. M. & Gould, R. V. (1994). A dilemma of state power: Brokerage and influence in the national health policy domain. *American Journal of Sociology*, 1455–1491.

Ferns, D. (1991). Developments in programme management. *International Journal of Project Management*, *9*(3), 148–156.

Flyvbjerg, B. & Budzier, A. (2011). Why your it project may be riskier than you think. *Harvard Business Review*, *89*(9), 601–603.

Forward, A. & Lethbridge, T. (2002). The relevance of software documentation, tools and technologies: a survey. In *Acm symposium on document engineering* (p. 26-33). doi: http://doi.acm.org/10.1145/585058.585065

Foucault, M., Burchell, G., Gordon, C. & Miller, P. (1991). *The foucault effect: Studies in governmentality*. University of Chicago Press.

Fox, J. & Das, S. K. (2000). *Safe and sound: artificial intelligence in hazardous applications* (Vol. 1). AAAI Press/MIT Press Menlo Park, CA/Cambridge.

Fraser, S., Reinitz, R., Eckstein, J., Kerievsky, J., Mee, R. & Poppendieck, M. (2003). Xtreme programming and agile coaching. In *Conference on object oriented programming systems languages and applications: Companion of the 18 th annual acm sigplan conference on object-oriented programming, systems, languages, and applications* (Vol. 26, pp. 265–267).

Ghauri, P. N. & Grønhaug, K. (2005). *Research methods in business studies: A practical guide*. Pearson Education.

Giordano, L., Terenziani, P., Bottrighi, A., Montani, S. & Donzella, L. (2006). Model checking for clinical guidelines: an agent-based approach. In *Amia annual symposium proceedings* (Vol. 2006, p. 289).

Grando, A., Peleg, M. & Glasspool, D. (2010). A goal-oriented framework for specifying clinical guidelines and handling medical errors. *Journal of biomedical informatics*, *43*(2), 287–299.

Granovetter, M. (1983). The strength of weak ties: A network theory revisited. *Sociological theory*, *1*(1), 201–233.

Granovetter, M. (2003). Ignorance, knowledge, and outcomes in a small world. *Science*, *301*(5634), 773–774.

Groenewegen, L., Kampenhout, N. v. & Vink, E. d. (2005). Delegation modeling with paradigm. In J.-M. Jacquet & G. Picco (Eds.), *Proceedings coordination 2005* (pp. 94–108). LNCS 3454.

Guzzo, R. A. & Dickson, M. W. (1996). Teams in organizations: Recent research on performance and effectiveness. *Annual Review of Psychology*, *47*(1), 307–338. doi: 10.1146/annurev.psych.47.1.307

Hanly, S., Wai, L., Meadows, L. & Leaton, R. (2006). Agile coaching in british telecom: Making strawberry jam. In *Agile conference, 2006* (pp. 9–pp).

Hansen, M. (2013). *Collaboration: How leaders avoid the traps, build common ground, and reap big results*. Harvard Business Press.

Hansen, M. T. (1999). The search-transfer problem: The role of weak ties in sharing knowledge across organization subunits. *Administrative science quarterly*, *44*(1), 82–111.

Hanssen, G. & Fægri, T. (2008). Process fusion: An industrial case study on agile software product line engineering. *Journal of Systems and Software*, *81*(6), 843–854.

Hatch, M. J. (2012). *Organization theory: modern, symbolic and postmodern perspectives*. Oxford university press.

Haythornthwaite, C. (2001). The strength and the impact of new media. In *System sciences, 2001. proceedings of the 34th annual hawaii international conference on* (pp. 10–pp).

Hazzan, O. & Dubinsky, Y. (2007, March). Why software engineering programs should teach agile software development. *SIGSOFT Softw. Eng. Notes*, *32*(2), 1–3.

Hewitt, B. & Walz, D. (2005). Using shared leadership to foster knowledge sharing in information systems development projects. *HICSS*, *8*, 256a. doi: http://doi.ieeecomputersociety.org/10.1109/HICSS.2005.666

Heydebrand, W. V. (1989). New organizational forms. *Work and occupations*, *16*(3), 323–357.

Highsmith, J. & Fowler, M. (2001). The agile manifesto. *Software Development Magazine*, *9*(8), 29–30.

Hobday, M. (2000). The project-based organisation: an ideal form for managing complex products and systems? *Research policy*, *29*(7), 871–893.

Hoda, R., Kruchten, P., Noble, J. & Marshall, S. (2010). Agility in context. In *Proceedings of the acm international conference on object oriented programming systems languages and applications* (pp. 74–88). NY, USA: ACM.

Hoda, R., Noble, J. & Marshall, S. (2011). The impact of inadequate customer collaboration on self-organizing agile teams. *Information and Software Technology*, *53*(5), 521–534.

Hodgkins, P. & Hohmann, L. (2007). Agile program management: Lessons learned from the verisign managed security services team. In *Proceedings of the agile 2007* (pp. 194–199). Washington, DC, USA: IEEE.

Holstein, J. A. & Gubrium, J. F. (1997). *Active interviewing*. Sage Publications.

Höst, M., Regnell, B. & Wohlin, C. (2000). Using students as subjectsa comparative study of students and professionals in lead-time impact assessment. *Empirical Software Engineering*, *5*(3), 201–214. doi: 10.1023/A:1026586415054

Igbaria, M. (1999). The driving forces in the virtual society. *Communications of the ACM*, *42*(12), 64–70.

Igbaria, M. & Tan, M. (1998). *The virtual workplace*. IGI Global.

IPMA. (2006). Icb-ipma competence baseline version 3.0. *Nijkerk, The Netherlands: International Project Management Association*.

Isern, D. & Moreno, A. (2008). Computer-based execution of clinical guidelines: a review. *International journal of medical informatics*, *77*(12), 787–808.

Kalliney, M. (2009). Transitioning from agile development to enterprise product management agility. In *Proceedings of the 2009 agile conference* (pp. 209–213). Washington, DC, USA: IEEE Computer Society.

Karau, S. & Williams, K. (1993). Social loafing: A meta-analytic review and theoretical integration. *Journal of Personality and Social Psychology*, *65*(4), 681-706.

Karlström, D. & Runeson, P. (2006, June). Integrating agile software development into stage-gate managed product development. *Empirical Softw. Eng.*, *11*, 203–225.

Katzy, B., Zhang, C. & Löh, H. (2005). Reference models for virtual organisations. In *Virtual organizations* (pp. 45–58). Springer.

Katzy, B. R. & Crowston, K. (2008). Competency rallying for technical innovation: The case of the Virtuelle Fabrik. *Technovation*, *28*(10), 679–692.

Kerzner, H. R. (2013). *Project management: a systems approach to planning, scheduling, and controlling*. John Wiley & Sons.

Kettunen, P. & Laanti, M. (2008). Combining agile software projects and large-scale organizational agility. *Softw. Process*, *13*, 183–193.

Khan, A. S. & Kajko-Mattsson, M. (2010). Demarcating the scope of a handover process. In *Software engineering advances (icsea), 2010 fifth international conference on* (pp. 244–251).

Kraut, A. I., Pedigo, P. R., McKenna, D. D. & Dunnette, M. D. (1989). The role of the manager: What's really important in different management jobs. *The Academy of Management Executive*, *3*(4), 286–293.

Krebs, J. (2008). *Agile portfolio management*. Microsoft Press.

Küster, J. M. & Engels, G. (2004). Consistency management within model-based object-

oriented development of components. In *Formal methods for components and objects* (pp. 157–176).

Laanti, M. (2008). Implementing program model with agile principles in a large software development organization. In (pp. 1383–1391). Washington, DC, USA: IEEE Computer Society.

Laanti, M., Salo, O. & Abrahamsson, P. (2011). Agile methods rapidly replacing traditional methods at nokia: A survey of opinions on agile transformation. *Information and Software Technology*, *53*(3), 276–290.

Laanti, M., Similä, J. & Abrahamsson, P. (2013). Definitions of agile software development and agility. In *Systems, software and services process improvement* (pp. 247–258). Springer.

Langley, A. (1999). Strategies for Theorizing from Process Data. *The Academy of Management Review*, *24*(4), 691–710.

Larman, C. (2004). *Agile and iterative development: a manager's guide*. Addison-Wesley Professional.

Latour, B. (2005). Reassembling the social: an introduction to actor-network-theory. *Oxford: Clarendon, 2005*.

Layman, L., Cornwell, T. & Williams, L. (2006). Personality types, learning styles, and an agile approach to software engineering education. In *Proceedings of the 37th sigcse technical symposium on computer science education* (pp. 428–432). New York, NY, USA: ACM.

Leffingwell, D. (2007). *Scaling software agility: best practices for large enterprises*. Addison-Wesley Professional.

Leffingwell, D. (2010). *Agile software requirements: lean requirements practices for teams, programs, and the enterprise*. Addison-Wesley Professional.

Lehto, I. & Rautiainen, K. (2009). Software development governance challenges of a middle-sized company in agile transition. In *Proceedings of the 2009 icse workshop on software development governance* (pp. 36–39).

Lethbridge, T. (2000). What knowledge is important to a software professional? *Computer*, *33*, 44-50. doi: http://doi.ieeecomputersociety.org/10.1109/2.841783

Lethbridge, T., Singer, J. & Forward, A. (2003). How software engineers use documentation: The state of the practice. *IEEE Software*, *20*(6), 35–39.

Levi-Faur, D. (2012). *The oxford handbook of governance*. Oxford University Press.

Levitt, B. & March, J. G. (1988). Organizational learning. *Annual review of sociology*, 319–340.

Lévy, P. & Bonomo, R. (1999). *Collective intelligence: Mankind's emerging world in cyberspace*. Perseus Publishing.

Lincoln, Y. S., Lynham, S. A. & Guba, E. G. (2011). Paradigmatic controversies, contradictions, and emerging confluences, revisited. *The Sage handbook of qualitative research*,

4, 97–128.

Lindkvist, L. (2004). Governing project-based firms: promoting market-like processes within hierarchies. *Journal of Management and Governance*, *8*(1), 3–25.

Lipnack, J. & Stamps, J. (1997). *Virtual teams: Reaching across space, time, and organizations with technology*. John Wiley & Sons, Inc.

Lycett, M., Rassau, A. & Danson, J. (2004). Programme management: a critical review. *International Journal of Project Management*, *22*(4), 289–299.

March, J. G. & Simon, H. A. (1993). *Organizations (2nd ed.)* [Book]. Cambridge, Massachusetts: Blackwell Publishers.

Maric, J. (2014). *Web communities, immigration, and social capital*. Unpublished doctoral dissertation, Tilburg.

Martinsuo, M. & Lehtonen, P. (2007). Role of single-project management in achieving portfolio management efficiency. *International Journal of Project Management*, *25*(1), 56–65.

Matta, N. F. & Ashkenas, R. N. (2003). Why good projects fail anyway. *Harvard Business Review*, *81*(9), 109–116.

McBreen, P. (2002). *Questioning extreme programming*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

McLeod, L. & MacDonell, S. G. (2011). Factors that affect software systems development project outcomes: A survey of research. *ACM Computing Surveys (CSUR)*, *43*(4), 24.

Melnik, G. & Maurer, F. (2004). Direct verbal communication as a catalyst of agile knowledge sharing. In *Proceedings of the agile development conference* (pp. 21–31). Washington, DC, USA: IEEE Computer Society.

Melnik, G. & Maurer, F. (2005). A cross-program investigation of students' perceptions of agile methods. In *Proceedings of the 27th international conference on software engineering* (pp. 481–488). New York, NY, USA: ACM.

Memmel, T., Gundelsweiler, F. & Reiterer, H. (2007). Agile human-centered software engineering. In *Proceedings of the 21st british hci group annual conference on people and computers: Hci...but not as we know it - volume 1* (pp. 167–175). Swinton, UK, UK.

Microsoft. (2005). *Digital Workstyle: The New World Of Work*. `https://www.microsoft.com/mscorp/execmail/2005/05-19newworldofwork.mspx`. ([Online; accessed 25-August-2014])

Miles, M. & Huberman, A. (1994). *Qualitative data analysis : An expanded sourcebook* (2. ed.). Thousand Oaks: Sage.

Miller, R. & Hobbs, J. B. (2005). Governance regimes for large complex projects. *International Journal of Project Management*, *36*(3), 42–51.

Mintzberg, H. (1979). The structuring of organizations: A synthesis of the research. *University of Illinois at Urbana-Champaign's Academy for Entrepreneurial Leadership Historical*

Research Reference in Entrepreneurship.

Mintzberg, H. (1993). *Structure in fives: Designing effective organizations.* Prentice-Hall, Inc.

Mockus, A. & Weiss, D. M. (2001). Globalization by chunking: A quantitative approach. *IEEE Software*, *18*(2).

Moe, N. B. & Dingsøyr, T. (2008). Scrum and Team Effectiveness: Theory and Practice. *Agile Processes in Software Engineering and Extreme Programming*, 11–20. doi: 10.1007/978-3-540-68255-4_2

Moe, N. B., Dingsøyr, T. & Dybå, T. (2010). A teamwork model for understanding an agile team: A case study of a scrum project. *Inf. Softw. Technol.*, *52*, 480–491.

Moe, N. B., Dingsøyr, T. & Dybå, T. (2009). Overcoming barriers to self-management in software teams. *IEEE Software*, *26*, 20–26.

Moe, N. B., Dingsøyr, T. & Røyrvik, E. A. (2009). Putting agile teamwork to the test– an preliminary instrument for empirically assessing and improving agile software development. In *Agile processes in software engineering and extreme programming* (pp. 114–123). Springer.

Morgan, G. (2007). *Images of organization*. Thousand Oaks: SAGE Publications.

Mowshowitz, A. (2002). *Virtual organization: Toward a theory of societal transformation stimulated by information technology*. Greenwood Publishing Group.

Müller, R., Pemsel, S. & Shao, J. (2014). Organizational enablers for governance and governmentality of projects: A literature review. *International Journal of Project Management*.

Mulyar, N., Pesic, M., Van Der Aalst, W. M. & Peleg, M. (2008). Declarative and procedural approaches for modelling clinical guidelines: addressing flexibility issues. In *Business process management workshops* (pp. 335–346).

Murray, A. et al. (2009). Managing successful projects with prince2.

Naito, E. & Hirose, S. (2014). Efficient foot motor control by neymars brain. *Frontiers in Human Neuroscience*, *8*, 594.

Nerur, S. & Balijepally, V. (2007). Theoretical reflections on agile development methodologies. *Communications of the ACM*, *50*, 79–83.

Nerur, S., Mahapatra, R. & Mangalaraj, G. (2005). Challenges of migrating to agile methodologies. *Commun. ACM*, *48*, 72–78.

Neus, A. (2001). Managing information quality in virtual communities of practice. In *Iq* (pp. 119–131).

Nonaka, I. & Takeuchi, H. (1995). *The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation*. Oxford University Press, USA. Hardcover.

OECD. (2004). *Oecd principles of corporate governance 2004*. OECD Publishing.

O'leary, M. B., Mortensen, M. & Woolley, A. W. (2011). Multiple team membership: A theoretical model of its effects on productivity and learning for individuals and teams.

Academy of Management Review, *36*(3), 461–478.

Ortega, J. (2001). Job rotation as a learning mechanism. *Management Science*, *47*(10), 1361–1370.

OConnor, R. V. & Duchonova, N. (2014). Assessing the value of an agile coach in agile method adoption. In *Systems, software and services process improvement* (pp. 135–146). Springer.

Paasivaara, M. & Lassenius, C. (2011). How does an agile coaching team work? a case study. In *Proceedings of the 2011 international conference on software and systems process* (pp. 101–109). New York, NY, USA: ACM.

Padula, A. (2009). Organically growing internal coaches. In *Proceedings of the 2009 agile conference* (pp. 237–242). Washington, DC, USA: IEEE Computer Society.

Paulhus, D. L. (2002). The role of constructs in psychological and educational measurement. In (pp. 46–69). Mahwah NJ: Lawrence Erlbaum.

Pearce, C. L. (2004). The Future of Leadership: Combining Vertical and Shared Leadership to Transform Knowledge Work. *Academy of Management Executive*, *18*(1), 47–57.

Peleg, M., Tu, S., Bury, J., Ciccarese, P., Fox, J., Greenes, R. A., ... others (2003). Comparing computer-interpretable guideline models: a case-study approach. *Journal of the American Medical Informatics Association*, *10*(1), 52–68.

Peleg, M. & Tu, S. W. (2009). Design patterns for clinical guidelines. *Artificial intelligence in medicine*, *47*(1), 1–24.

Pemsel, S. & Müller, R. (2012). The governance of knowledge in project-based organizations. *International Journal of Project Management*, *30*(8), 865–876.

Pentland, B. T. & Feldman, M. (2008). Designing routines: On the folly of designing artifacts, while hoping for patterns of action. *Information and Organization*, *18*(4), 235–250.

Pentland, B. T. & Feldman, M. S. (2007). Narrative networks: Patterns of technology and organization. *Organization Science*, *18*(5), 781–795.

Pentland, B. T. & Rueter, H. H. (1994). Organizational Routines as Grammars of Action. *Administrative Science Quarterly*, *39*(3), 484–510.

Petersen, K. & Wohlin, C. (2009). A comparison of issues and advantages in agile and incremental development between state of the art and an industrial case. *J. Syst. Softw.*, *82*(9), 1479–1490.

Piore, M. & Sabel, C. (1984). *The second industrial divide: Possibilities for prosperity*. New York, Basic Book.

PMI. (2008). *The standard for portfolio management*. Project Management Institute.

Polanyi, M. (1967). *The tacit dimension*.

Prencipe, A. & Tell, F. (2001). Inter-project learning: processes and outcomes of knowledge codification in project-based firms. *Research policy*, *30*(9), 1373–1394.

Preston, C. (2000). Optimal number of response categories in rating scales: reliability, validity, discriminating power, and respondent preferences. *Acta Psychologica*, *104*(1), 1–15.

Prinz, W., Jeners, N., Ruland, R. & Villa, M. (2009). Supporting the change of cooperation patterns by integrated collaboration tools. In *Leveraging knowledge for innovation in collaborative networks* (pp. 651–658). Springer.

Ramesh, B., Cao, L. & Baskerville, R. (2007). Agile requirements engineering practices and challenges: an empirical study. *Information Systems Journal*, *20*(5), 449–480.

Rautiainen, K., von Schantz, J. & Vahaniitty, J. (2011). Supporting scaling agile with portfolio management: Case paf.com. In (pp. 1–10). Washington, DC, USA: IEEE Computer Society.

Reijers, H. A. & Mendling, J. (2011). A study into the factors that influence the understandability of business process models. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, *41*(3), 449–462.

Reyck, B., Grushka-Cockayne, Y., Lockett, M., Calderini, S., Moura, M. & Sloper, A. (2005). The impact of project portfolio management on information technology projects. *International Journal of Project Management*, *23*(7), 524–537.

Richards, D. (2009). Designing project-based courses with a focus on group formation and assessment. *Trans. Comput. Educ.*, *9*, 2:1–2:40.

Robinson, H. & Sharp, H. (2003). Xp culture: Why the twelve practices both are and are not the most significant thing. In *Agile development conference, 2003. adc 2003. proceedings of the* (pp. 12–21).

Robson, C. (2002). *Real word research*. Oxford: Blackwell.

Rosenberg, D. & Scott, K. (1999). *Use case driven object modeling with uml: a practical approach*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

Rubin, E. & Rubin, H. (2011). Supporting agile software development through active documentation. *Requirements Engineering*, *16*, 117-132.

Rundle, P. J. & Dewar, R. G. (2006). Using return on investment to compare agile and plan-driven practices in undergraduate group projects. In *Proceedings of the 28th international conference on software engineering* (pp. 649–654). New York, NY, USA: ACM.

Salas, E., Sims, D. & Burke, C. (2005). Is there a big five in teamwork? *Small Group Research*, *36*(5), 555-599.

Salvato, C. (2009). Capabilities unveiled: The role of ordinary activities in the evolution of product development processes. *Org. Science*, *20*(2), 384-409.

Sari, B., Loeh, H. & Katzy, B. R. (2010). Emerging collaboration routines in knowledge-intensive work processes: Insights from three case studies. *International Journal of e-Collaboration (IJeC)*, *6*(1), 33–52.

Schaffers, H., Brodt, T., Pallot, M. & Prinz, W. (2006). The future workplace-perspectives on mobile and collaborative working. *Telematica Instituut, The Netherlands*.

Schurink, E. & Schurink, E. (1998). Deciding to use a qualitative research approach. *Research at grass roots*, 239–251.

Schwaber, K. & Beedle, M. (2001). *Agile software development with scrum* (1st ed.). Upper Saddle River, NJ, USA: Prentice Hall PTR.

Senapathi, M. & Srinivasan, A. (2014). An empirical investigation of the factors affecting agile usage. In *Proceedings of the 18th international conference on evaluation and assessment in software engineering* (p. 10).

Shahar, Y., Miksch, S. & Johnson, P. (1998). The asgaard project: a task-specific framework for the application and critiquing of time-oriented clinical guidelines. *Artificial intelligence in medicine*, *14*(1), 29–51.

Sharp, H. & Robinson, H. (2004, December). An ethnographic study of xp practice. *Empirical Softw. Engg.*, *9*(4), 353–375.

Sharp, H. & Robinson, H. (2010). Three cs of agile practice: collaboration, co-ordination and communication. In *Agile software development* (pp. 61–85). Springer.

Sharp, H., Robinson, H. & Petre, M. (2009, January). The role of physical artefacts in agile software development: Two complementary perspectives. *Interacting with Computers*, *21*, 108–116.

Shea, G. & Guzzo, R. (1987). Group effectiveness: What really matters? *Sloan Management Review*, *28*, 25–31.

Shea, S., DuMouchel, W. & Bahamonde, L. (1996). A meta-analysis of 16 randomized controlled trials to evaluate computer-based clinical reminder systems for preventive care in the ambulatory setting. *Journal of the American Medical Informatics Association*, *3*(6), 399–409.

Sheppard, J. & Young, W. (2006). Agility literature review: classifications, training and testing. *Journal of sports sciences*, *24*(9), 919–932.

Sillitti, A., Ceschi, M., Russo, B. & Succi, G. (2005). Managing uncertainty in requirements: A survey in documentation-driven and agile companies. In *Proceedings of the 11th ieee international software metrics symposium.* Washington, DC, USA: IEEE Computer Society.

Silva, K. & Doss, C. (2007). The growth of an agile coach community at a fortune 200 company. In *Proceedings of the agile 2007* (pp. 225–228). Washington, DC, USA: IEEE Computer Society.

Smith, L., Mann, S. & Buissink-Smith, N. (2001). Crashing a bus full of empowered software engineering students. *New Zealand Journal of Applied Computing and Information Technology*, *5*, 69–74.

Snowden, D. J. (2005). Multi-ontology sense making: a new simplicity in decision making. *Informatics in Primary Care*, *13*(1), 45–54.

Sonnenberg, F. & Hagerty, C. (2006). Computer-interpretable clinical practice guidelines. *Where are we and where are we going*, 145–158.

SPS. (2010). *Stiftung Produktive Schweiz: The future world of work.* `http://www.produktive-schweiz.ch/Publikationen/NWOW.aspx?lang=en-US`". ([Online;

accessed 25-August-2014])

Stettina, C. J. & Heijstek, W. (2011a). Five agile factors: Helping self-management to self-reflect. In *Proceedings of european software process improvement conference (eurospi 2011)*. Roskilde, Denmark.

Stettina, C. J. & Heijstek, W. (2011b). Necessary and neglected? An empirical study of internal documentation in agile software development teams. In *Proceedings of the 29th acm international conference on design of communication* (pp. 159–166). New York, NY, USA: ACM.

Stettina, C. J., Heijstek, W. & Fægri, T. E. (2012). Documentation work in agile teams: The role of documentation formalism in achieving a sustainable practice. In (pp. 31–40). Washington, DC, USA: IEEE.

Stokes, D. (1997). *Pasteur's quadrant: Basic science and technological innovation*. Brookings Institution Press.

Strauss, A. & Corbin, J. M. (1990). *Basics of qualitative research: Grounded theory procedures and techniques.* Sage Publications, Inc.

Stray, V. G., Moe, N. B. & Aurum, A. (2012). Investigating daily team meetings in agile software projects. *2010 36th EUROMICRO Conference on Software Engineering and Advanced Applications*, *0*, 274-281. doi: http://doi.ieeecomputersociety.org/10.1109/SEAA.2012.16

Sykes, H. B. & Dunham, D. (1995). Critical assumption planning: A practical tool for managing business development risk. *Journal of Business Venturing*, *10*(6), 413–424.

Takeuchi, H. & Nonaka, I. (1986). The new new product development game. *Harvard Business Review*.

Talby, D. & Dubinsky, Y. (2009). Governance of an agile software project. In *Software development governance, 2009. sdg'09. icse workshop on* (pp. 40–45).

Tata, J. & Prasad, S. (2004). Team Self-Management, Organizational Structure and Judgments of Team Effectiveness. *Journal of Managerial Issues*, *16*(2), 248+.

Taylor, F. W. (1911). *The principles of scientific management*. New York: Harper and Brothers.

Taylor, H. (1999). Role-play cases for teaching interviewing skills in information systems analysis. In *Proceedings of herdsa annual international conference* (pp. 1–9).

Thomas, J. C. & Baker, S. W. (2008). Establishing an agile portfolio to align it investments with business needs. In *Proceedings of the agile 2008* (pp. 252–258). Washington, DC, USA: IEEE Computer Society.

Thompson, E. R. & Phua, F. T. T. (2005). Reliability among senior managers of the marlowe-crowne short-form social desirability scale. *Journal of Business and Psychology*, *19*(4), 541–554.

Thompson, L. (2002). *Making the team. ch. 2 and ch. 4*. New York: Prentice-Hall.

Thummadi, B. V., Shiv, O. & Lyytinen, K. (2011). Enacted routines in agile and waterfall

processes. In *Agile* (p. 67-76). IEEE Computer Society.

Tierney, W. & Holley, K. (2008). Inside pasteur's quadrant: knowledge production in a profession. *Educational Studies*, *34*(4), 289-297.

Tiwana, A. & Keil, M. (2004). The one-minute risk assessment tool. *Communications of the ACM*, *47*(11), 73–77.

Too, E. G. & Weaver, P. (2013). The management of project management: a conceptual framework for project governance. *International Journal of Project Management*.

Tu, S. W., Campbell, J. & Musen, M. A. (2004). The sage guideline modeling: motivation and methodology. *Studies in health technology and informatics*, 167–171.

Tuckman, B. W. & Jensen, M. A. C. (1977). Stages of small-group development revisited. *Group & Organization Management*, *2*(4), 419–427.

Turner, J. R. & Keegan, A. (1999). The versatile project-based organization: governance and operational control. *European Management Journal*, *17*(3), 296–309.

Turner, J. R. & Müller, R. (2003). On the nature of the project as a temporary organization. *International Journal of Project Management*, *21*(1), 1–8.

Vähäniitty, J. (2012). *Towards agile product and portfolio management*. Unpublished doctoral dissertation, Aalto University.

Van Den Bosch, F. A., Volberda, H. W. & De Boer, M. (1999). Coevolution of firm absorptive capacity and knowledge environment: Organizational forms and combinative capabilities. *Organization Science*, *10*(5), 551–568.

van Oosterhout, M. P. A. (2010). *Business agility and information technology in service organizations* (No. EPS-2010-198-LIS). Erasmus Research Institute of Management (ERIM).

Venkatagiri, S. (2011). Teach project management, pack an agile punch. In *Proceedings of the 2011 24th ieee-cs conference on software engineering education and training* (pp. 351–360). Washington, DC, USA: IEEE Computer Society.

Vlietland, J. & van Vliet, H. (2014). Towards a governance framework for chains of scrum teams. *Information and Software Technology*.

von Nordenflycht, A. (2010). What is a professional service firm? Toward a theory and taxonomy of knowledge-intensive firms. *Academy of Management Review*, *35*(1), 155–174.

Walton, R. E. & Hackman, J. R. (1986). *Designing effective work groups*. San Francisco: Jossey-Bass Publishers.

Wang, X. (2000). Performance measurement in budgeting: a study of county governments. *Public Budgeting & Finance*, *20*(3), 102–118.

Waterman Jr, R. H., Peters, T. J. & Phillips, J. R. (1980). Structure is not organization. *Business Horizons*, *23*(3), 14–26.

Weick, K. E., Sutcliffe, K. M. & Obstfeld, D. (2008). Organizing for high reliability: Processes of collective mindfulness. *Crisis management*, *3*, 81–123.

Weiss, P. (1960). Knowledge: a growth process. *Proceedings of the American Philosophical Society*, 242–247.

Wenger, E. (1998). *Communities of practice: Learning, meaning, and identity*. Cambridge university press.

Whittington, R., Pettigrew, A., Peck, S., Fenton, E. & Conyon, M. (1999). Change and complementarities in the new competitive landscape: A european panel study, 1992–1996. *Organization Science*, *10*(5), 583–600.

Williams, L. (2012). What agile teams think of agile principles. *Commun. ACM*, *55*(4), 71–76.

Wilson, J. Q. (2000). *Bureaucracy: What government agencies do and why they do it*. Basic Books.

Woolley, A. W., Chabris, C. F., Pentland, A., Hashmi, N. & Malone, T. W. (2010). Evidence for a collective intelligence factor in the performance of human groups. *science*, *330*(6004), 686–688.

Yin, R. K. (2009). *Case study research: Design and methods (applied social research methods)* (Fourth Edition. ed.). Sage Publications.

Young, R. & Jordan, E. (2008). Top management support: Mantra or necessity? *International Journal of Project Management*, *26*(7), 713–725.

Summary

Why is governance of knowledge workers in project organizations necessary and yet so difficult? Governance defines the roles, rules and responsibilities of project work (cf. Ahola et al. (2013)). While knowledge workers resemble the majority of our society today, I argue that the governance models applied today have been designed to govern manual workers. Knowledge workers challenge traditional bureaucracy-based project governance found in organizations today. They see that their organization is deeply rooted in the ideas of Scientific Management and geared toward the control of manual workers. My desire to investigate these concerns leads to the following problem statement (PS) as presented in Chapter 1. PS: *Is governance of innovation project management necessary or neglected?*

A knowledge worker is someone who contributes to a knowledge-based economy through the application of theoretical and analytical knowledge. His aim is to solve complex problems in context. Examples of knowledge workers are project managers, software engineers, doctors, scientists, and lawyers. As knowledge emerges in context over time, knowledge workers discover their tasks only while executing the actual assignment. This procedure challenges existing frameworks of project governance that follow a bureaucratic approach and rely on the prediction of outcomes by management in a top-down manner. Because outcomes cannot be predicted up-front and planning cannot be done in a traditional manner, the traditionally top-down planned approach to project governance is notoriously difficult to follow in knowledge worker projects that develop new products and services. Strictly viewed, traditional methods hinder innovation, as they require a new contract to be signed each time when new ideas emerge (e.g., through the availability of new technology, and through market changes).

To answer the PS I have formulated three research questions (RQs) which guide the research. They read as follows:

- RQ1: *How can governing bodies in organizations of knowledge worker teams understand and steer multiple knowledge worker project teams in practice?*
- RQ2: *What are the components of governance necessary to understand knowledge worker project organizations?*
- RQ3: *How can governance address the dynamics of knowledge work across multiple knowledge worker teams?*

The study aims to identify effective governance models able to cope with challenges in knowledge worker organizations creating new products and services.

To address this aim I follow the advice of contemporary authors by rebuilding the organization, starting with the individual knowledge worker. There are many books on governance, most of which discuss a top-down approach (see, e.g., Levi-Faur (2012)). Others argue that we need to go back to the work floor in order to (re)develop governance for a knowledge worker

society (e.g., Latour, 2005). Analogously to the time-and-motion studies by Taylor (1911), while omitting the mechanistic views, I study knowledge workers, their work in teams, and their embedding in organizations. In order to address this challenge I have divided the research into a number of subsequent smaller studies. I have designed eight studies, each of which has a rigorous methodology and a concrete result. Together they should bring us to a better understanding of the subject in context.

The thesis divided into five main parts: (I) Introduction, (II-IV) Empirical Investigation, and (V) Conclusions. Within the empirical part I have divided the research questions into a number of subsequent studies. To improve readability the empirical part is further sub-divided into the three levels of analysis consisting of: (II) the knowledge worker, (III) the knowledge worker teams, and (IV) the multi-project organization. The empirical part consists of eight self-contained studies. Each of the studies follows a rigorous methodology and provides concrete research results answering RQ1, RQ2, and RQ3, and contributes to understanding the overall problem statement.

The thesis contributes to theory building in five ways. First, based on empirical findings across three levels of analysis I rebuild the organization, beginning with the individual knowledge worker, the knowledge worker project teams, and their embedding in the scope of an organization. Second, based on my analysis and in relation to the existing literature, I develop the concept of *Substantial Management*, advocating a new role and responsibilities of management as coordinating body across knowledge worker teams. Third, based on my case studies of agile teams in product development, software engineering, medicine and academia, I contribute to a better understanding of the role of routines and boundary objects as carriers of governance. Fourth, based on the reestablished view on the organization, I develop a framework for governance in knowledge worker organizations. Fifth, I further contribute to theory building by developing a number of propositions to be evaluated in future research.

The thesis also contributes to practice. Following the ideas of use-inspired basic research as pursued by Louis Pasteur (see Stokes (1997)), this study does not only contribute to theory but has also practical implications. My contribution to public administration is to look at governance approaches in agile project management as an alternative to top-down governance based on power. First, I provide a list of characteristics of agile knowledge worker teams and how to improve those using a reflective questionnaire. Second, I provide understanding and recommendations for knowledge transfer across project teams using routines and documentation as boundary objects. Third, I provide a model consisting of governance components across three layers for participants to be considered in a knowledge worker project organization - as part of a reoccurring governance routine.

Samenvatting

Waarom is het aansturen van kenniswerkers in projectorganisaties noodzakelijk, maar tegelijkertijd ook zo moeilijk? Dit komt door *Governance*. Governance bepaalt de rollen, regels en verantwoordelijkheden van projectwerk (cf. Ahola et al. (2013)). Ondanks dat kenniswerkers de meerderheid van de huidige samenleving vertegenwoordigen, beweer ik dat de governance-modellen die vandaag de dag gebruikt worden, ontworpen zijn om werknemers die fysieke arbeid verrichten aan te sturen. Organisaties van nu zijn gestoeld op ideeën van Wetenschappelijke Bedrijfsvoering, ook voor het aansturen van fysieke arbeid. Daarom betwisten kenniswerkers in onze huidige organisaties de traditionele bureaucratische manier van projectmanagement. Mijn behoefte om deze uitdagingen te onderzoeken leidt tot de volgende probleemstelling (PS) zoals gepresenteerd in Hoofdstuk 1. PS: *Is het aansturen van innovatief projectmanagement noodzakelijk of verwaarloosd?*

Een kenniswerker is iemand die bijdraagt aan een kenniseconomie door het toepassen van theoretische en analytische kennis. Het doel is om complexe problemen op te lossen binnen de context. Voorbeelden van kenniswerkers zijn projectmanagers, software ontwikkelaars, artsen, wetenschappers en advocaten. Kenniswerkers ontdekken hun precieze taak juist tijdens de uitvoering, naarmate kennis zich in een bepaalde context ontwikkelt. Dit daagt de bestaande kaders van projectmatig werken uit. Ze zijn gebaseerd op een bureaucratische benadering en ze vertrouwen op voorspelling van de uitkomsten die door het management op een top-down manier verricht zijn. Omdat uitkomsten van te voren niet echt voorspeld kunnen worden, kan men ze derhalve niet volledig plannen. Daarom is de traditionele en top-down benadering van projectmanagement bijzonder moeilijk bij het aansturen van kenniswerkers die nieuwe producten en diensten ontwerpen. Strikt genomen verhinderen traditionele methoden iedere innovatie omdat zij een nieuw contract behoeven elke keer dat nieuwe ideeën opkomen (bijv. door de beschikbaarheid van nieuwe technologieën en door veranderingen in de markt).

Om de PS te beantwoorden heb ik drie onderzoeksvragen (OVs) geformuleerd om het onderzoek richting te geven. Ze luiden als volgt:

- OV1: *Hoe kunnen governing bodies in organisaties van kenniswerkerteams meerdere kenniswerker projectteams in de praktijk begrijpen en aansturen?*
- OV2: *Welke componenten van governance zijn noodzakelijk om kenniswerker projectorganisaties te begrijpen?*
- OV3: *Hoe kan governance de dynamiek van kennis werk over meerdere kenniswerkerteams adresseren?*

Dit onderzoek heeft het doel om effectieve modellen voor *governance* te identificeren die in staat zijn om te gaan met de uitdagingen binnen de kenniswerkorganisaties.

Om dit doel te bereiken volg ik het advies van recente auteurs door de organisatie te herstructureren, beginnende bij de individuele kenniswerker. Er is veel gepubliceerd over *governance*, vooral over de top-down stijl (zie bijv., Levi-Faur (2012)). Anderen beweren dat we terug moeten naar de werkvloer om het leidinggeven in een kenniswerkmaatschappij te kunnen (her)ontwikkelen (bijv., Latour, 2005). Vergelijkbaar met het tijd-en-bewegingsonderzoek van Taylor (1911) (daarbij de mechanische zienswijze weglatend) bestudeer ik (1) kenniswerkers, (2) hun manier van werken in teamverband, en (3) hun inbedding in de organisatie. Om deze uitdaging te adresseren heb ik het onderzoek in een aantal opvolgende studies verdeeld. Hiervoor heb ik acht onderzoeken ontworpen, met elk een gedegen methodologie en concrete resultaten. Deze studies gezamenlijk zullen ons meer inzicht in het onderwerp en haar context geven.

Het proefschrift is onderverdeeld in vijf delen: (I) Introductie, (II-IV) Empirisch Onderzoek en (V) Conclusies. De onderzoeksvragen zijn in het Empirisch gedeelte verder onderverdeeld in een aantal deelonderzoeken. Om de leesbaarheid te vergroten is het empirische gedeelte verder geordend in drie analyse-niveaus bestaande uit: (II) de kenniswerker, (III) kenniswerkerteams en (IV) de multi-projectorganisaties. Het empirische deel bestaat zoals genoemd uit acht autonome onderzoeken. Elk van deze studies is uitgevoerd vanuit een strikte methodologie en geven ons concrete onderzoeksresultaten voor OV1, OV2 en OV3, en dragen bij aan het algehele begrip van de probleemstelling. Het proefschrift draagt op vijf manieren bij aan theorievorming. Ten eerste, gebaseerd op de empirische bevindingen verkregen op drie analyseniveaus, heb ik de organisatie geherstructureerd; beginnende bij de individuele kenniswerker, de projectteams van kenniswerkers en hun inbedding op het terrein van de organisatie. Ten tweede, gebaseerd op mijn analyse en relatie met de bestaande literatuur ontwikkel ik het concept *Substantial Management*, waarbij ik pleit voor een nieuwe rol en nieuwe verantwoordelijkheden voor management als een coördinerende eenheid over teams van kenniswerkers. Ten derde, gebaseerd op mijn case studies van agile teams in productontwikkeling, software engineering, de gezondheidszorg en academici, draag ik bij aan het begrijpen van de rol van routines en *boundary object* als overbrengers van governance. Ten vierde, gebaseerd op de opnieuw vastgestelde inzichten over de organisatie, ontwikkel ik kaders voor leidinggeven in kenniswerkorganisaties. Ten vijfde, draag ik bij aan de theorievorming door aanbevelingen te formuleren voor vervolgonderzoek.

Dit proefschrift draagt ook bij aan de dagelijkse praktijk. In navolging van de ideeën van Louis Pasteur (zie Stokes (1997)) over *use-inspired* onderzoek, draagt dit onderzoek niet alleen bij aan de theorievorming, maar heeft het ook praktische implicaties. Zo wil ik de overheid laten zien dat de wijze van aansturing in agile projectmanagement een alternatief is voor het hiërarchische top-down management. Daarvoor heb ik ten eerste een lijst ontwikkeld met eigenschappen van agile kenniswerkteams en hoe die verbeterd kunnen worden middels een reflectieve vragenlijst. Ten tweede verschaf ik inzicht en geef ik aanbevelingen voor kennisoverdracht tussen projectteams die gebruik maken van routines en documentatie als

boundary objects. Ten derde lever ik een model aan dat bestaat uit governance componenten over drie deelnemerslagen voor organisaties die nieuwe producten en diensten ontwerpen in projectteams. Dit kan beschouwd worden als onderdeel van een hernieuwde governance routine.

Acknowledgments

Exploring - *Science is learning in collaboration, as such in this thesis many ideas have been influenced by or have emerged in discussions with others. Hereby I would like to thank:*

Within the LIACS Technology and Innovation Management group my thanks go to Zhao Zhou for discussions on entrepreneurship, Wouter Mensink for his insights into philosophy, and Claudia Bücker for feedback and mental support. I would like to thank Werner Heijstek for the great out-of-group collaboration and introducing me to conference publications. Luuk Groenewegen for a refreshing perspective on computer science and his pleasant nature. Vivienne Medik for collaboration on New Ways of Working, inspiring discussions and her always helpful manner. Tor Erlend Fægri for collaboration and insightful discussions on interaction of routines and artifacts. Torgeir Dingsøyr for inspiration and introducing me to the world of scientific research on agile methods. Mateusz Tkaczen for discussions on connecting to interior architecture.

The large data set discussed in this thesis would have not been possible without the students I had the honor to supervise. Here I would like to in particular thank Jeannette Hörz for the work on portfolio management, Egbert Kroon for project handovers, Shi Hao Zijdemans for contracting, and providing a helping hand at workhops.

Engaging - *Project management and governance is a practical research topic. As a social complex phenomenon it needs to be studied in context. I would like to thank:*

A big thank you goes to the Centre for Innovation team. Gideon Shimshon for critical eye, guidance and opportunities given. Ulrich Mans for strategy. Sjoerd Louwaars for being an engaging spirit. Janne Polman for making the Agile for Excellence launch possible with Vivienne. Marja Verstelle for online learning and discussions on portfolio management. Jouke de Vries for a campus of opportunities.

Our Lynks R&D team, Uli, Gideon, Eelke Heemskerk, Wouter Eekhout, Eugene Tjoa, Jan Kalmeijer for making our network visualization platform the first fully agile R&D project at the centre. Jan-Willem Brock, Kim Zunderdorp, Kavish Sewnandan and Patrick Klaassen for the collaboration on portfolio management and governance at Leiden University. Daniel van den Hoven, Kim Bosman and others at Rally Software for the collaboration, making the launch event *Agile for Excellence* and the portfolio management game possible. Marloes Plomp and Jorn Poldermans for the Dutch translation. Gijsbert Dijker, and his students Pascal Schilp, Allisan Salazar, Charlotte Gramberg, Fay Asselbergs, and Yasheng Zhang at the Royal Academy of Art (KABK).

Connecting - *Due to the empirical nature of the thesis, access to projects in context was crucial. I would like to thank the following individuals for their trust.*

The NITIM graduate school for a network of friends. Agile Consortium and the Dutch National Research Group in Project Management (DNRG) for their workshop series. Michiel De Vries from Deloitte. Berend van Duijvenvoorde from Agile Overheid. Berend van der Eijk from Bird&Bird. Everyone who generously contributed to this study, you know who you are.

Grounding - *Space for ideas and motivation are necessary to cope with the uncertainty of research.*

My grandfather, Rafael Swienty, for support, his reflective nature, and making my studies abroad possible. My family. Nikita Swikker and her parents Ellen and Koos for a great time and inspiring trips. Thomas Holzmann and Ozgur Dedehayir for mental support. Timo Sandmeier, Chris Nowitzki and Markus Herbst for fun.

About the Author

Christoph Johann Stettina was born in 1980 in Peiskretscham. Christoph earned a master's degree (German: Diplom-Informatiker) in Computer Engineering at FH Dortmund, as well as a master's degree (MA) in Project Management completed at the Norwegian University of Science and Technology (NTNU).

Christoph began his professional career in 2004 at Nokia where he worked on mobile phone and automotive connectivity related software systems in Bochum, Germany. After finishing his masters degree in Trondheim, Norway he moved to The Hague, Netherlands where he worked as a project officer and research associate at the Center for Technology and Innovation Management.

Christoph currently works as a consultant, project leader and researcher at the Centre for Innovation at Leiden University. He is the founder of the Agile for Excellence initiative, helping organizations to excel in research and innovation through agile team based project management methods. With education in computer engineering and project management Christoph has a wide range of experience in management of research and innovation projects with a passion for challenges and technology.