Marshall Plan Scholarship Report, 31.1.2015, Raoul Vallon (vallon@stanford.edu)

Empirically Driven Design of the Agile Distributed Adaptable Process Toolkit (ADAPT)

Prof. Thomas Grechenig
Vienna University of Technology

Prof. Larry Leifer
Stanford University

Abstract

This final report describes the findings from the research visit at the Center for Design Research, Stanford University, for designing the ADAPT framework towards the completion of the dissertation work. The design theory behind the ADAPT framework, as presented in this report, has been further improved in private discussions with Prof. Larry Leifer, as well as in a workshop and several one-on-one interviews with members of the DesignX Lab, Center for Design Research, at Stanford.

Contents

CHAPTER 1

Acknowledgement

First and foremost, I would like to thank Professor Larry Leifer for providing the opportunity to join the Center for Design Research at Stanford University as a visiting researcher and my PhD supervisor Professor Thomas Grechenig for making it possible to pursue it. I deeply appreciate the financial support I received from the Austrian Marshall Plan Foundation, allowing this research visit to be a major contribution to my dissertation work. I also want to thank Anneliese Tunison (Stanford) for guiding me through the administrative process.

I am also grateful having been able to meet many wonderful people in Stanford, which have been a great enrichment both on a research-related and personal level (no particular order): Srinath Sibi, Becky Currano, David Sirkin, Wendy Ju (all Stanford), Alexander Gottwald (Otto-von-Guericke University Magdeburg), Mikko Sääskilahti (Aalto University), Ville Taajamaa (University of Turku), Alexander Neff (University of St. Gallen), Dirk Rothenbücher (University of Mannheim), Andreas Kunze (Technische Universität München) and Chris Turner (University of Oxford).

Last but not least, I want to thank my fiancée Christiane in supporting me to take that step in my PhD journey.

CHAPTER 2

The ADAPT Framework's Design Theory

Contents

Before diving into the design theory for the ADAPT (Agile Distributed Adaptable Process Toolkit) framework, the status quo of frameworks in the field is presented. Damian and Zowghi (2003) developed an issue-based model focusing on requirements engineering in distributed environments. Ågerfalk et al. (2005) worked on a framework of distributed development issues (which also appeared in (Ågerfalk et al., 2006)). Hossain et al. (2011) presented a research framework which maps GSD (global software development) challenges and mitigation strategies and discusses how scrum practices could be implemented in practice based on a systematic literature review. However, past frameworks are described in a high level of abstraction, giving only exemplary advise on how to implement the practices. The authors also conclude that "there is a substantial need for research to "catch up" and support the needs of practice" (Hossain et al., 2011, p. 100). The ADAPT framework aims to provide detailed advice on how to implement agile practices successfully based on empirical evidence gathered from a multiple-case study.
Section 2.1 investigates suitable design research theory to achieve that goal. Section 2.2 looks at how related papers have dealt with framework design and development and derives essential aspects for the creation of the ADAPT

framework. The process for implmenting the ADAPT framework is drafted in Section 2.3. Section 2.4 concludes the report by presenting the design components based on the former investigation to tackle the research question *What are suitable design components for building a distributed agile process framework?*.

2.1 Design Research

Developing the ADAPT framework is a constructivist approach in the sense that the developed artifact is the chief output to the research (Gregor and Jones, 2007). According to Gregg et al. (2001), the ADAPT framework would be rated as *medium* (conceptual), *low* (formal) and *none* (developmental), describing an incremental extension and/or generalization of an existing concept (applying agile to GSD), based on descriptive details (practices extracted from multiple-case study) and without implementation at this stage (future work). The framework can thus be rated as "a new/innovative concept with limited formal and development research effort" (Gregg et al., 2001, p. 177). The design theory is described with the components developed by Gregor and Jones (2007), which builds on top of Aristotle's writing on the four explanations of any *thing* explanation: (Falcon, 2014) (literal translation from Greek, see (Hooker, 1996)):

- The material cause: that out of which, e.g., the bronze of a statue.
- The formal cause: the form, the account of what-it-is-to-be, e.g., the shape of a statue.
- The efficient cause: the primary source of the change or rest, e.g., the artisan, the art of bronze-casting the statue, the man who gives advice, the father of the child.
- The final cause: the end, that for the sake of which a thing is done, e.g., health is the end of walking, losing weight, purging, drugs, and surgical tools.

The four causes apply "to everything that requires an explanation, including artistic production and human action" (Falcon, 2014). Gregor and Jones (2007) extend on the four causes and define eight components as essential to the anatomy of design theory (six core and two additional ones, cf. Table 2.1).

2.2 Framework Design

This section investigates related design research on building frameworks in general to deduct knowledge for building the ADAPT framework.

Component	**Description**
Core components	
1) Purpose and scope (the causa finalis)	What the system is for, the set of meta-requirements or goals that specifies the type of artifact to which the theory applies and in conjunction also defines the scope, or boundaries, of the theory.
2) Constructs (the causa materialis)	Representations of the entities of interest in the theory.
3) Principle of form and function (the causa formalis)	The abstract blueprint or architecture that describes an IS artifact, either product or method/intervention.
4) Artifact mutability	The changes in state of the artifact anticipated in the theory, that is, what degree of artifact change is encompassed by the theory.
5) Testable propositions	Truth statements about the design theory.
6) Justificatory knowledge	The underlying knowledge or theory from the natural or social or design sciences that gives a basis and explanation for the design(kernel theories).
Additional components	
7) Principles of implementation (the causa efficiens)	A description of processes for implementing the theory (either product or method) in specific contexts.
8) Expository instantiation	A physical implementation of the artifact that can assist in representing the theory both as an expository device and for purposes of testing.

Table 2.1 – The eight components of design theory as defined by Gregor and Jones (2007)

The term *framework* within this report is understood as defined by Wild et al. (2009, p. 147):

> "A framework can be seen to be a general set of concepts for understanding a research area. It is not tightly organised enough to be a predictive theory. It aims to sketch out the general concepts of a field of enquiry & the possible relationships between them."

More specifically, the ADAPT framework is a *process framework*, which is defined by Sorathia et al. (2010, p. 297) as:

> "it integrates various elements involved in different phases of the software development life-cycle. Once the process is well defined, the individual teams can utilize required process subsets or the entire process and also may customize these to meet individual requirements."

The framework is based on empirical evidence only. As such it provides an overview of what worked in which distribution scenarios in a description-oriented fashion (in contrast to being prescription-oriented) (Van Aken, 2005). Table 2.2 illustrates that the framework is designed to support a process instantiation (Gregor and Jones, 2007), i.e. to derive a concrete process implemenation based on the information provided within the ADAPT framework.

	ADAPT Framework	**Process Instantiation**
Artifact type	Abstract artifact	Material artifact (instantiation)
Description	A framework including challenges, guidelines, practices and distribution scenarios	The concrete instantiated process implementation, guided by ADAPT's design guidelines and based on a subset of the ADAPT's practices

Table 2.2 – ADAPT framework vs a concrete process instantiation (inspired by (Gregor and Jones, 2007))

The framework consists of challenges (cf. Section 2.2.2), guidelines (cf. Section 2.2.3) and effective practices (cf. Section 2.2.4). The ADAPT framework is by design similar to (Soundararajan et al., 2012) in the way that it uses a three-layered setup and links principles (ADAPT uses guidelines) to practices. Figure 2.1 shows the schematic outline of the ADAPT framework.

Grounded theory is used for theory building from the case study research in this thesis (Eisenhardt, 1989). Practices and guideline candidates evolve from coding three case studies to ensure empirical grounding. Additionally further support is sought from related empirical studies to strengthen the emerging theory. The framework will thus be designed iteratively. Preliminary *concepts*, i.e. guidelines and practices, are extracted after each individual case and assigned to the pre-set categories of coordination, control and communication (cf. 2.2.2). The framework will thus be built in three iterations, one after each case and then discussed against related studies and evaluated in expert interviews.

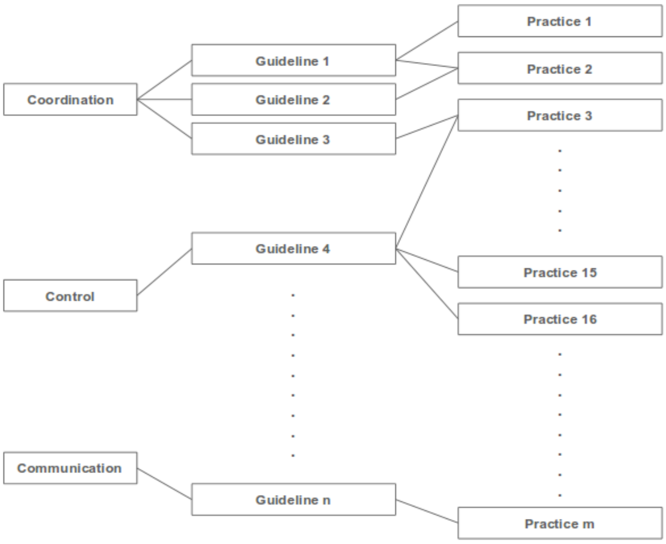


Figure 2.1 – Schematic outline of the ADAPT framework: Challenges, Guidelines and Practices

2.2.1 Software Process Tailoring

Research efforts in software process tailoring go back to the 1980s (Akbar et al., 2011) but it is still a relatively new topic that has not been extensively researched (Martínez-Ruiz et al., 2012). It is a necessity in both traditional processes, e.g. Rational Unified Process (Hanssen et al., 2005), or agile ones, e.g. XP (Mirakhorli et al., 2008) or scrum (Kniberg, 2007; Kniberg and Skarin, 2010). Pedreira et al. (2007) differentiate between formal and informal approaches and argue that formal approaches may be better for large organizations with a planned and strictly managed process, while small and medium-sized organizations may benefit from a simple and pragmatic process. The informal process tailoring approach suits lightweight agile thinking better and will be picked up for process tailoring with the ADAPT framework. Software process tailoring can be done at different levels, e.g. organizational and project level (Pedreira et al., 2007). Although it is acknowledged that context consists of both organizational and project-based parts (Xu and Ramesh, 2003), the ADAPT framework will focus on project-based tailoring due to the argument that each project is unique even within the same organization. Applying agile practices to distributed software development is no silver bullet solution, the process has to be tailored correctly to the individual project's needs. Failure to do so will not produce better results, as Alqahtani et al. (2013) showed: 75% of the the studies report a lack of communication and collaboration in agile DSD. Dumitriu et al. (2011) argue that GSD and agile software development are two extremes (distribution and collocation) that are not easy to integrate, so tailoring must be seen as finding an optimal compromise between the two in order to allow agile practices to reduce the consequences of geographical, temporal and socio-cultural distance. It also very important that the rationale behind the practice is understood for a succesful process tailoring (Šmite et al., 2010).

2.2.2 GSD Challenge Categories

There have been different categorizations of challenges in distributed software development such as (Kajko-Mattsson et al., 2010; Mudumba and Lee, 2010; Sriram and Mathew, 2012). The ADAPT framework follows the most established approach in the field of GSD to classify challenges in categories *communication*, *coordination* and *control* (Carmel, 1999; Carmel and Agarwal, 2001; Ågerfalk et al., 2005, 2006; Holmström et al., 2006; Pries-Heje and Pries-Heje, 2011; Hossain et al., 2011). The categories are described as follows.

> "*Coordination* is the act of integrating each task with each organizational unit, so the unit contributes to the overall objective. Orchestrating the integration often requires intense and ongoing communication.
>
> *Control* is the process of adhering to goals, policies, standards, or quality levels. Controls can be formal (such as budgets and explicit guidelines) or informal (such as peer pressure). We recognize today that, for knowledge workers, coordination and control have in many ways blended together.
>
> *Communication* is a mediating factor affecting both coordination and control. It is the exchange of complete and unambiguous informationthat is, the sender and receiver can reach a common understanding." (Carmel and Agarwal, 2001, p. 23)

Carmel and Agarwal (2001, p. 23) states that "coordination and control have in many ways blended together". For disambiguation of the two terms, further elaboration is required: Coordination and control can be seen as two sides of the same coin (Nurmi et al., 2005), which is the process of managing dependencies among activities (Malone and Crowston, 1994). The extreme of each side would be *organic coordination* (cooperative, informal, and decentralized) and *mechanistic control* (controlling, formal, and centralized) (McCann and Galbraith, 1981). Another important distinction is that coordination is the work of dependent parts towards a common goal (Nurmi et al., 2005), while control is needed when the goals of individual stakeholders differ from those of the larger overall entity (Sabherwal, 2003). Figure 2.2 shows the original draft of the categories in (Carmel and Agarwal, 2001). It illustrates well that communication is a mediating factor for both coordination and control.

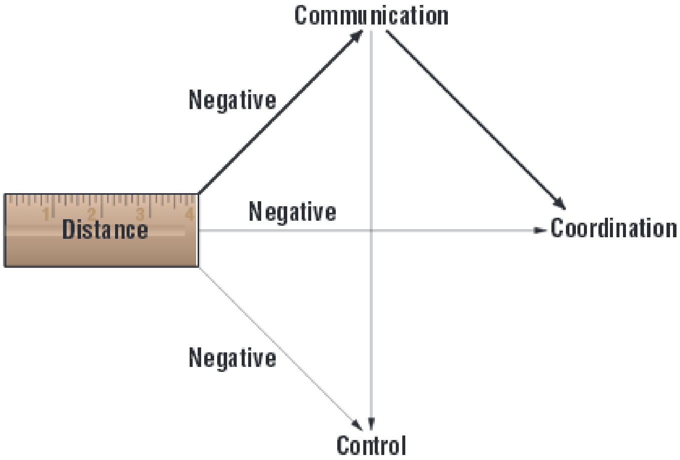


Figure 2.2 – Challenge categories by Carmel and Agarwal (2001): Impacts of distance in distributed software development

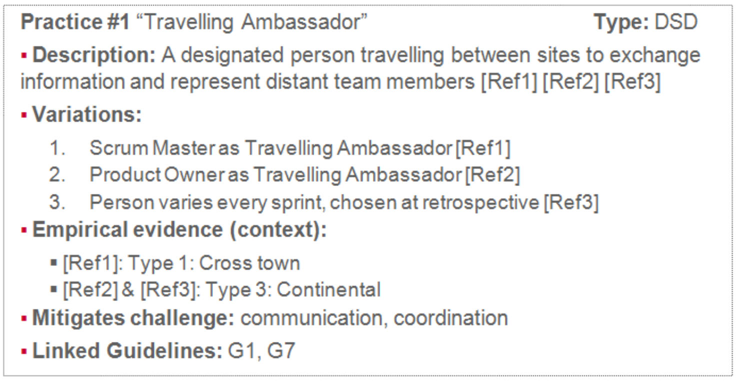
These three categories are the starting point to group guidelines (cf. Section 2.2.3) and *effective practices* (cf. Section 2.2.4) of the ADAPT framework. Pries-Heje and Pries-Heje (2011) and Hossain et al. (2011) have both worked with the CCC model (coordination, control, communication) in distributed agile environments and thus underlined its applicability to distributed agile environments.

2.2.3 Design Guidelines

Design guidelines are the second layer in the ADAPT framework (cf. Figure 2.1) and overarch the practices. Similar to (Soundararajan et al., 2012), the relationship between guidelines and practices is *N to M*, which means that a guideline is linked to several practices and a practice can be linked to one or more guidelines. The guidelines are treated as *constructive heuristics* (Heeager and Rose, 2014) and emerge from case study research. The design guidelines specifically aim at guiding the practitioner to build his individual process instantiation.

2.2.4 Effective Practices based on Context

The goal of this framework is to provide effective practices that have a successful empirical grounding. The identified practices will not be called *best practices* as no practice can be "best" in every context (Ambler, 2011). Inspired by (Ambler, 2002), the practices to be identified are called *effective practices*. They are regarded as *effective* because they rely on succesful empirical implementation evidence and are thus seen to meet a goal with higher



Practice #1 “Travelling Ambassador” **Type:** DSD

- **Description:** A designated person travelling between sites to exchange information and represent distant team members [Ref1] [Ref2] [Ref3]
- **Variations:**
  1. Scrum Master as Travelling Ambassador [Ref1]
  2. Product Owner as Travelling Ambassador [Ref2]
  3. Person varies every sprint, chosen at retrospective [Ref3]
- **Empirical evidence (context):**
  - [Ref1]: Type 1: Cross town
  - [Ref2] & [Ref3]: Type 3: Continental
- **Mitigates challenge:** communication, coordination
- **Linked Guidelines:** G1, G7

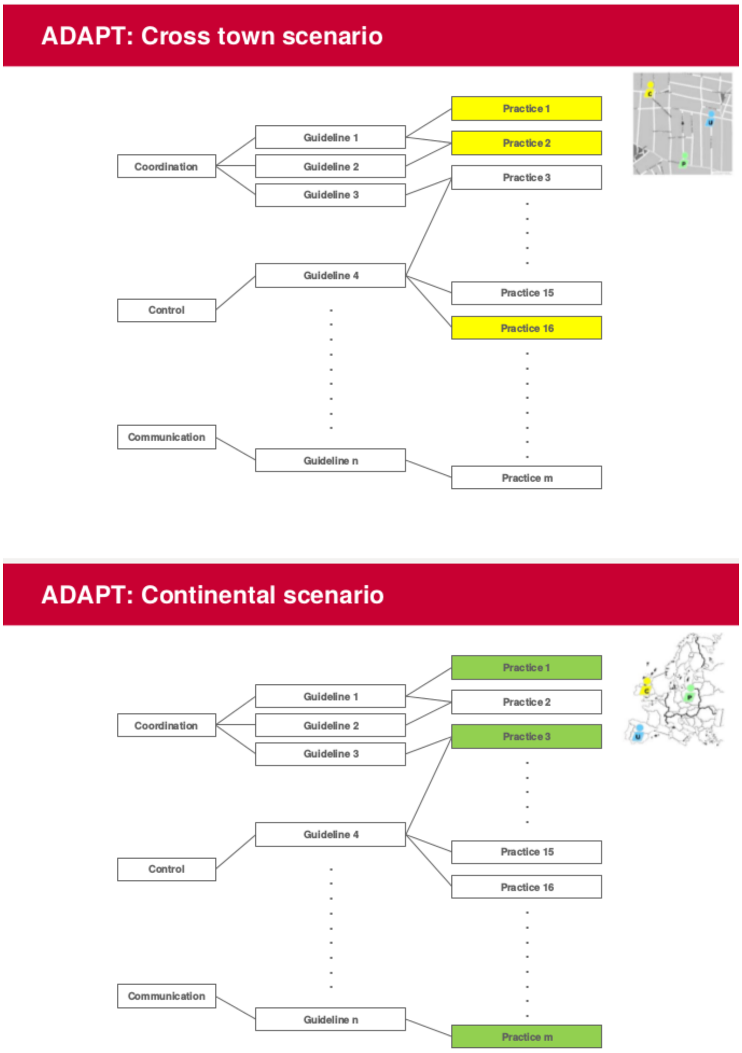
Figure 2.3 – Examplary practice of the ADAPT framework

probability and fewer risks involved (Schatten et al., 2010). The practices can be regarded as *method fragments* (Baskerville and Pries-Heje, 2013). It is a major concern that that the practices be detailed enough and not too simple in their description in order to be of practical usability (Baskerville and Pries-Heje, 2013).

It seems to be agreed by the research community that context is a major concern for any type case study research, although different ways of reporting context have been proposed (Kitchenham et al., 1995; Runeson and Höst, 2009; Petersen and Wohlin, 2009; Jalali and Wohlin, 2010, 2012). This dissertation follows and extends on the checklist for reporting context by Jalali and Wohlin (2010, 2012) as it has been applied in an extensive systematic review of empirical studies. Practices within the ADAPT framework will be reported including their context of application. This criterion leads to the constraint that no theoretical practices will be part of the ADAPT framework, a successful empirical application is the minimum requirement (sine qua non) for being considered for inclusion in the framework.

Figure 2.3 shows an exemplary *effective practice*. As the design process of the ADAPT framework is iterative, the final practice layout is subject to change. The practice is the smallest fragment of the ADAPT framework in the hierarchy of challenges, guidelines and practices. The focus is on providing effective and tangible practices to be used by the practitioner.
To that end, the practices are clustered based on their distribution scenarios (Prikladnicki et al., 2003), which are:

- Cross Town Scenario
- No Time Shift Scenario
- Continental Scenario



ADAPT: Cross town scenario

ADAPT: Continental scenario

Figure 2.4 – Distribution scenarios offer a different perspective on a subset of practices beside the challenge-guideline-practice hierarchy

- Global Scenario

The distribution scenarios offer a different perspective on the hierarchy of challenges, guidelines and practices, as illustrated in Figure 2.4. to the empirical background and variations, offering a view on a subset of practices.
The practices may evolve into a *pattern language* (Alexander et al., 1977) at a later stage based on the ADAPT framework, once a significant amount of good empirical research (including a rich description of the study context and background) has been done on the subject.

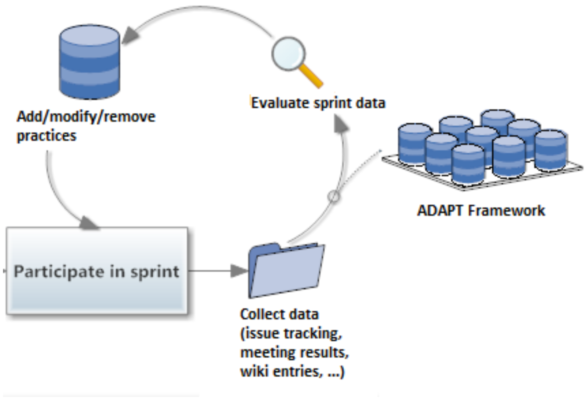

Add/modify/remove practices
Evaluate sprint data
ADAPT Framework
Participate in sprint
Collect data (issue tracking, meeting results, wiki entries, ...)

Figure 2.5 – Using the ADAPT framework for process design

2.3 Process Design

A *process design* (Aken, 2004) is necessary because ”professionals need to know how to apply the knowledge in their own unique and specific cases” (Gregor and Jones, 2007, p.322). The practices presented in the ADAPT framework can be consumed in a ”supermarket approach” (Baskerville and Pries-Heje, 2013), i.e. it is advised to implement a minimal set to satisfy all guidelines, but the framework users, i.e. the practitioners, decide which of the practices to select. The classification in distribution scenario helps as a starting point for experimentation and fosters the process of arriving from the general ADAPT framework to the concrete process instantiation. The process design is an iterative process, where practices should be evaluated and then modified or replaced in regular retrospective-type meetings. In order to maintain the self-organization of teams, the decision on what practices to select should be a majority vote (bottom up) rather than a (top down) management decision to achieve a better level of acceptance to change. Figure 2.5 illustrated the proposed iterative agile process design.

2.4 Design Components

This final section builds on top of the prior discussion in this report and defines the design components in Table 2.3 to complete the presentation of the design theory for the development of the ADAPT framework.

Component	**Description**
1) Purpose and scope	The aim is to develop a framework for applying agile practices effectively to (globally) distributed software development.

2) Constructs	The framework is representated by the following hierarchy of entities: challenge categories, guidelines and effective practices.
3) Principles of form and function	A process framework is provided to aid practitioners and researchers in tailoring agile practices to the respective unique distributed environment.
4) Artifact mutability	The design process supports continuous construction cycles, allowing the practitioners to add, modify or remove practices as the project (and thus empirical feedback) progresses. The framework is designed to be open for integrating additional effective practices and guidelines as the research field evolves over time, possibly introducing a more prescriptive nature in future work.
5) Testable propositions	The ADAPT framework should satisfy the four propositions P1 to P4, which are presented in Table 2.4
6) Justificatory knowledge	The framework is grounded in current research on agile software development and global software development. The design theory has been presented and improved during a four-month research visit at the Center for Design Research at Stanford University.
7) Principles of implementation	The process-design (how to arrive from the generic ADAPT framework at the concrete process instantiation) is an iterative process utilizing agile feedback loops at the process level.
8) Expository instantiation	Expository instantiations are provided based on varying distribution scenarios.

Table 2.3 – The eight design components of the ADAPT framework's design theory

Proposition	**Rationale**	**Metric**
P1. Each practice of the ADAPT framework is grounded in empirical evidence.	The ADAPT framework is not a silver bullet solution but it is a set of tools based on empirical evidence showing what worked in which context.	100% of practices must specify the context of origin.
P2. The ADAPT framework allows a simple, pragmatic and iterative process tailoring (rather than planned and strictly managed).	Process Tailoring should be part of any agile implementation.	Evaluation within study.
P3. The ADAPT framework supports project-based process tailoring (rather than organization-based).	Even within the same organization each project is unique.	Each distribution scenario has sufficient practices to satisfy the implementation of all guidelines.
P4. The ADAPT framework provides tangible and detailed advice to the practitioner.	In order to be of practical use the practices must provide enough detail.	Evaluation interviews with experts

Table 2.4 – Testable propositions for the design theory of the ADAPT framework

List of Figures

List of Tables

Bibliography

Ågerfalk, J, Fitzgerald, Brian and In, Old Petunias. Flexible and distributed software processes: old petunias in new bowls. In *Communications of the ACM*, 2006.

Ågerfalk, Pär J, Fitzgerald, Brian, Holmström, Helena, Lings, Brian, Lundell, Björn and Conchúi, Eoin Ó. A framework for considering opportunities and threats in distributed software development. 2005.

Akbar, Rehan, Hassan, Mohd Fadzil and Abdullah, Azrai. A Review of Prominent Work on Agile Processes Software Process Improvement and Process Tailoring Practices. In *Software Engineering and Computer Systems*, pp. 571–585. Springer, 2011.

Aken, Joan E van. Management research based on the paradigm of the design sciences: the quest for field-tested and grounded technological rules. *Journal of management studies*, 41(2):219–246, 2004.

Alexander, Christopher, Ishikawa, Sara and Silverstein, Murray. *A pattern language: towns, buildings, construction*, volume 2. Oxford University Press, 1977.

Alqahtani, Abdullah Saad, Moore, John David, Harrison, David K and Wood, Bruce M. The Challenges of Applaying Distributed Agile Software Development: A Systematic Review. *International Journal of Advances in Engineering & Technology*, 5(2):23–36, January 2013.

Ambler, Scott. *Agile modeling: effective practices for extreme programming and the unified process*. John Wiley & Sons, 2002.

Ambler, Scott W. Questioning "Best Practices" for Software Development: Practices are Contextual, Never Best, 2011. URL `http://www.ambysoft.com/essays/bestPractices.html`.

Baskerville, Richard and Pries-Heje, Jan. Discursive Co-development of Agile Systems and Agile Methods. In *Grand Successes and Failures in IT. Public and Private Sectors*, pp. 279–294. Springer, 2013.

Carmel, Erran. *Global software teams: collaborating across borders and time zones*. Prentice Hall PTR, 1999.

Carmel, Erran and Agarwal, Ritu. Tactical approaches for alleviating distance in global software development. *Software, IEEE*, 18(2):22–29, 2001.

Damian, Daniela E and Zowghi, Didar. RE challenges in multi-site software development organisations. *Requirements engineering*, 8(3):149–160, 2003.

Dumitriu, Florin, Oprea, Dumitru and Mesnita, Gabriela. Issues and strategy for agile global software development adoption. *Recent researchers in Applied Economics*, pp. 37–42, 2011.

Eisenhardt, Kathleen M. Building theories from case study research. *Academy of management review*, 14(4):532–550, 1989.

Falcon, Andrea. Aristotle on Causality. In Zalta, Edward N., editor, *The Stanford Encyclopedia of Philosophy*. Spring 2014 edition, 2014. URL `http://plato.stanford.edu/archives/spr2014/entries/aristotle-causality/`.

Gregg, Dawn G, Kulkarni, Uday R and Vinzé, Ajay S. Understanding the philosophical underpinnings of software engineering research in information systems. *Information Systems Frontiers*, 3(2):169–183, 2001.

Gregor, Shirley and Jones, David. The anatomy of a design theory. *Journal of the Association for Information Systems*, 8(5):312–335, 2007.

Hanssen, Geir K, Westerheim, Hans and Bjørnson, Finn Olav. Tailoring RUP to a defined project type: A case study. In *Product Focused Software Process Improvement*, pp. 314–327. Springer, 2005.

Heeager, Lise Tordrup and Rose, Jeremy. Optimising agile development practices for the maintenance operation: nine heuristics. *Empirical Software Engineering*, pp. 1–23, 2014.

Holmström, Helena, Fitzgerald, Brian, Ågerfalk, Pär J and Conchúir, Eoin Ó. Agile practices reduce distance in global software development. *Information Systems Management*, 23(3):7–18, 2006.

Hooker, R. Aristotle: The Four Causes-Physics II. 3, 1996. URL `http://richard-hooker.com/sites/worldcultures/GREECE/4CAUSES.HTM`.

Hossain, Emam, Bannerman, Paul L and Jeffery, D Ross. Scrum practices in global software development: a research framework. In *Product-focused software process improvement*, pp. 88–102. Springer, 2011.

Jalali, Samireh and Wohlin, Claes. Agile practices in global software engineering-A systematic map. In *Global Software Engineering (ICGSE), 2010 5th IEEE International Conference on*, pp. 45–54. IEEE, 2010.

Jalali, Samireh and Wohlin, Claes. Global software engineering and agile practices: a systematic review. *Journal of Software: Evolution and Process*, 24(6):643–659, 2012.

Kajko-Mattsson, Mira, Azizyan, Gayane and Magarian, Miganoush Katrin. Classes of distributed Agile development problems. In *Agile Conference (AGILE), 2010*, pp. 51–58. IEEE, 2010.

Kitchenham, Barbara, Pickard, Lesley and Pfleeger, Shari Lawrence. Case studies for method and tool evaluation. *IEEE software*, 12(4):52–62, 1995.

Kniberg, Henrik. Scrum and XP from the Trenches. *Lulu.com*, 2007.

Kniberg, Henrik and Skarin, Mattias. *Kanban and Scrum-making the most of both.* Lulu.com, 2010.

Malone, Thomas W and Crowston, Kevin. The interdisciplinary study of coordination. *ACM Computing Surveys (CSUR)*, 26(1):87–119, 1994.

Martínez-Ruiz, Tomás, Münch, Jürgen, García, Félix and Piattini, Mario. Requirements and constructors for tailoring software processes: a systematic literature review. *Software Quality Journal*, 20(1):229–260, 2012.

McCann, Joseph and Galbraith, Jay R. Interdepartmental relations. *Handbook of organizational design*, 2:60–84, 1981.

Mirakhorli, Mehdi, Khanipour Rad, Abdorrahman, Shams, Fereidoon, Pazoki, Maryam and Mirakhorli, Abbas. RDP technique: A practice to customize XP. In *Proceedings of the 2008 international workshop on Scrutinizing agile practices or shoot-out at the agile corral*, pp. 23–32. ACM, 2008.

Mudumba, Venkateshwara and Lee, One-Ki. A new perspective on GDSD risk management: agile risk management. In *Global Software Engineering (ICGSE), 2010 5th IEEE International Conference on*, pp. 219–227. IEEE, 2010.

Nurmi, Antti, Hallikainen, Petri and Rossi, Matti. Coordination of Outsourced Information System Development in Multiple Customer Environment-A Case Study of a Joint Information System Development Project. In *System Sciences, 2005. HICSS'05. Proceedings of the 38th Annual Hawaii International Conference on*, pp. 260a–260a. IEEE, 2005.

Pedreira, Oscar, Piattini, Mario, Luaces, Miguel R and Brisaboa, Nieves R. A systematic review of software process tailoring. *ACM SIGSOFT Software Engineering Notes*, 32(3):1–6, 2007.

Petersen, Kai and Wohlin, Claes. Context in industrial software engineering research. In *Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement*, pp. 401–404. IEEE Computer Society, 2009.

Pries-Heje, Lene and Pries-Heje, Jan. Why Scrum works: A case study from an agile distributed project in Denmark and India. In *Agile Conference (AGILE), 2011*, pp. 20–28. IEEE, 2011.

Prikladnicki, Rafael, Audy, Jorge Luis Nicolas and Evaristo, J Roberto. Distributed Software Development: Toward an Understanding of the Relationship Between Project Team, Users and Customers. In *ICEIS (3)*, pp. 417–423. Citeseer, 2003.

Runeson, Per and Höst, Martin. Guidelines for conducting and reporting case study research in software engineering. *Empirical software engineering*, 14 (2):131–164, 2009.

Sabherwal, Rajiv. The evolution of coordination in outsourced software development projects: a comparison of client and vendor perspectives. *Information and organization*, 13(3):153–202, 2003.

Schatten, Alexander, Biffl, Stefan, Demolsky, M, Gostischa-Franta, E, Östreicher, T and Winkler, D. *Best Practice Software-Engineering.* 2010.

Šmite, Darja, Moe, Nils Brede and Ågerfalk, Pär J. Fundamentals of agile distributed software development. In *Agility Across Time and Space*, pp. 3–7. Springer, 2010.

Sorathia, Vikram, van Sinderen, Marten and Pires, Luís Ferreira. Towards a Unifying Process Framework for Services Knowledge Management. In *Exploring Services Science*, pp. 295–299. Springer, 2010.

Soundararajan, Shvetha, Arthur, James D and Balci, Osman. A Methodology for Assessing Agile Software Development Methods. In *Agile Conference (AGILE), 2012*, pp. 51–54. IEEE, 2012.

Sriram, R and Mathew, SK. Global software development using agile methodologies: A review of literature. In *Management of Innovation and Technology (ICMIT), 2012 IEEE International Conference on*, pp. 389–393. IEEE, 2012.

Van Aken, Joan Ernst. Management research as a design science: articulating the research products of mode 2 knowledge production in management. *British journal of management*, 16(1):19–36, 2005.

Wild, PJ, Clarkson, PJ and McFarlane, DC. A framework for cross disciplinary efforts in services research. In *Proceedings of the 19th CIRP Design Conference–Competitive Design*. Cranfield University Press, 2009.

Xu, Peng and Ramesh, Balasubramaniam. A tool for the capture and use of process knowledge in process tailoring. In *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on*. IEEE, 2003.