TEAMS IN AGILE SOFTWARE DEVELOPMENT:

DESIGN PRINCIPLES AND EXAMINATION OF HUMAN FACTORS

Zur Erlangung des akademischen Grades eines

Doktors der Wirtschaftswissenschaften

Dr. rer. pol.

bei der KIT-Fakultät für Wirtschaftswissenschaften

des Karlsruher Instituts für Technologie (KIT)

genehmigte

DISSERTATION

von

Karl Werder, M. Sc.

Tag der mündlichen Prüfung: 19.05.2017

Referent: Prof. Dr. Alexander Mädche

Korreferent: Prof. Dr. Sjaak Brinkkemper (Universität Utrecht)

Karlsruhe, 2017

Table of Content

List of Figures

List of Tables

Abbreviations List

ASD	agile software development
CAS	complex adaptive system
CF	collaboration function
CMMI	capability maturity model integration
CS	collaboration structure
EASI	emotions-as-social-information
HCI	human-computer interaction
IMOI	input-mediator-output-input
IPO	input-process-outcome
IS	information system(s)
ISF	information systems frontiers
ISM	information systems management
ISO	international organization for standardization
ISR	information systems research
IST	information and software technology
IDE	integrated development environment
IPSO	input-process-state-output
JSS	journal of systems and software
LDUF	little design upfront
ME	method engineering
MISQ	management of information systems quarterly
OSS	open source software
PDD	process deliverable diagram
PDISC	software product discovery
RQ	research question
SDT	software development team
SE	software engineering
SRQ	subresearch question
TED	team emotional display

TLE	team leader emotional display
UCASD	user-centered agile software development
UCD	user-centered design
UI	user involvement
XP	eXtreme programming

Abstract

In response to new customer requirements, market dynamics, mergers, and technological innovation, modern software development organizations are adopting agile software development (ASD). Yet, the simple adoption of agile methods such as Scrum or eXtreme programming does not automatically result in a very agile team. While we understand the introduction and adoption of ASD from a methodical perspective, we have yet to explore design principles that guide methodical extensions of ASD, and we need to learn more about the human factors that influence software development teams. This thesis presents four studies. Studies 1 and 2 investigate the methodical extension of ASD by identifying design principles from secondary data. Study 1 extends ASD with processes and practices from user-centered design. Study 2 investigates early activities that precede development activities. The thesis also investigates human factors of agile software development in studies 3 and 4. Study 3 compares teams along their extents of agility in order to identify influential factors using a multicase study design. Study 4 tests the effects of emotional contagion in virtual software development teams using a large dataset from an open source software repository. Thus, this thesis makes two primary contributions. First, it develops design principles for methodical extensions of ASD; second, it contributes to the human factors that influence software development teams. Managers also receive guidance on the improvement of ASD in their organization.

1. Introduction

Software development organizations need to respond to multiple challenges, including new customer requirements, market dynamics, mergers, and technological innovation (Börjesson & Mathiassen, 2005). Users' increased expectations require that software development organizations provide additional value, for instance by increasing user experiences or flexible pricing (Gruman, Morrison, & Retter, 2007). Thus, software development organizations must improve their reactions to changes generally and changing customer requirements in particular. Many organizations are increasingly adopting agile software development (ASD) as their development approach (Serrador & Pinto, 2015; West & Grant, 2010). ASD is the result of a joint effort by 17 practitioners who suggested a shift in software development (Beck et al., 2001). Their ideas have been documented in 4 values and 12 principles. Since then, the idea of ASD has received widespread attention and has been adopted by practitioners and academics beyond its original domain (Bazigos, Smet, & Gagnon, 2015; Conboy, 2009; R. G. Cooper & Sommer, 2016).

1.1. Problem Statement

Simply adopting agile methods such as Scrum or eXtreme programming (XP) will not automatically lead to an agile organization (Conboy, 2009; Gregory et al., 2016). Practitioners face different challenges when becoming agile. One example is the development organization for Microsoft's Visual Studio Online, which took four years to move from a waterfall-oriented organization to becoming a truly agile one that releases new features into the cloud-based product in a three-week cycle (Bisson, 2015). There are different states of agility, but we lack a sound understanding of how they emerge. The thesis sheds light on this dilemma by identifying different dynamics that explain different emergent states of agility in software development teams; it also explains the influences of emotional processes in development teams.

While scholars have made tremendous advancements researching ASD and its core concept, agility, the majority of contributions have been in the same vein. Much of the research relates to the introduction and adoption of agile software development (e.g. Gandomani & Nafchi, 2015; Laanti, Salo, & Abrahamsson, 2011; Maruping, Venkatesh, & Agarwal, 2009), and the use of different agile practices (e.g. Balijepally et al., 2009; Hoda et al., 2011; Strode, 2015). Research into the introduction and adoption of ASD advances the understanding of antecedents of agile practices (Chow & Cao, 2008; J. Iivari & Iivari, 2011; G. Lee & Xia, 2010) and the effects of being agile (G. Lee & Xia, 2010; Maruping, Zhang, & Venkatesh, 2009). Also, research into practices has helped us to understand the dependencies of practice selection (Misra, Kumar, & Kumar, 2009; Strode, 2015) and the impacts of practice selection (Balijepally et al., 2009; Hoda et al., 2011).

However, more research is needed to further advance the field. First, scholars have not yet researched and understood the methodical extension of ASD, for instance, through the combination with other well-known

methods (R. G. Cooper & Sommer, 2016; da Silva et al., 2011; Sohaib & Khan, 2010). This methodical extension is needed, since it can help to address practitioners' concerns (Gregory et al., 2016). For instance, ASD's key focus is the development of useful software. Yet, ASD can benefit from aspects of user-centered design (UCD) and its focus on usable software (Fox, Sillito, & Maurer, 2008). Also, when investigating methodical suggestions for early activities that should happen prior to development activities, popular agile methods such as Scrum or XP provide little or no guidance (e.g. Rising & Janoff, 2000).

Second, more research is needed into the human factors during ASD (Dybå & Dingsøyr, 2008). Particularly the investigation at the team-level has received little attention in the research, despite its importance in software development (Sawyer, 2004). While effects at the individual-level can provide valuable insights (Balijepally et al., 2009), they may be neutralized in a team setting (Hollenbeck et al., 2002). Also, the research has often focused on the entire organization (Gandomani & Nafchi, 2015; J. Iivari & Iivari, 2011). While a single team can be very agile, the organization may not share the same values and ideas, resulting in different challenges to researchers and academics (Gregory et al., 2016). These research challenges must be investigated in order to advance the field towards developing an agile theory.

1.2. Research Questions and Objectives

The thesis objectives are to advance agile research by investigating antecedents on the team-level and to develop principles for the next evolution of agile methods. I develop theoretical explanations for the enhancement of agile teams and of ASD as a methodical phenomenon. The primary research question is:

RQ: *Which human factors and design principles enhance agile software development?*

To address the primary research question, I derive four distinct subresearch questions. I address each subresearch question in a separate section. Answering all subresearch questions provides the foundation for answering the primary research question. I will now describe each subresearch question.

Responding to methodical challenges of ASD, the combination and integration with existing dominant software creation methods has drawn increasing interest among scholars (e.g. R. G. Cooper & Sommer, 2016). Particularly, the combination of ASD and UCD has been suggested to provide complementary benefits (Fox et al., 2008). While ASD focuses on software's functionality and usefulness, UCD seeks to identify software with higher usability. Study 1 derives clear principles that constitute a user-centered and agile software development (UCASD) approach by answering the following subresearch question (SRQ):

SRQ1: *Which principles constitute a user-centered agile software development approach?*

While agile methods such as Scrum and XP provide guidance for the adoption and execution of agile methods during development, they provide little or no guidance concerning early activities that should take place be prior to the main development activities (Larusdottir, Gulliksen, & Cajander, 2017). Given this knowledge void, the third subresearch question is:

SRQ2: *Which principles guide a software product discovery method?*

Given the multiple calls for more research into the human factors in ASD (Dybå & Dingsøyr, 2008) and the need for more team-level research as the central unit of operations (Sawyer, 2004), research subquestion 2 focuses on agility as a team-level phenomenon. Given that development teams are complex, we investigate individual, team, and organizational factors. For instance, individual-level effects can be negated at a team-level (Hollenbeck et al., 2002). I also seek to answer calls for more research benefitting from theoretical ideas from mature reference disciplines. I focus on the team-level, benefiting from prior theoretical contributions in this field (e.g. Kozlowski & Ilgen, 2006). To investigate and explain agility as a team-level phenomenon, subresearch question 3 is:

SRQ3: *Which factors influence the team agility of agile software development teams?*

Besides the emergence of different states, emotions are an important cue for a team's coordination behavior (Homan, van Kleef, & Sanchez-Burks, 2016). While positive emotions have been shown to increase the productivity of a team and to dominate negative emotions (van Kleef et al., 2009), negative emotions require regulation in order to prevent negative outcomes (Goldenberg, Halperin, Van Zomeren, & Gross, 2016; T. Sy, Côté, & Saavedra, 2005). In addition to investigating the tension between positive and negative emotions, this thesis investigates the roles of social network theoretical properties in the form of team collaboration structure and team collaboration function. These properties are suggested to influence emotional contagion from OSS team leaders' emotional display onto the team emotional display. Therefore:

SRQ4: *What are the moderating effects of collaboration structure and collaboration function on emotional contagion?*

Academics and industry decision-makers will benefit from the thesis findings in multiple ways. First, this thesis is one of the first to investigate team agility as an emergent state. This helps us to explain the varying degrees and individual experiences related to agile maturity (F. S. Silva et al., 2015). The results also respond to calls for more theory-driven research based on mature reference disciplines (Dingsøyr, Dybå, & Abrahamsson, 2008; Dingsøyr, Nerur, Balijepally, & Moe, 2012; Dybå & Dingsøyr, 2008). Second, results identify different human factors that help to explain observations in agile teams. These factors relate to the

development team as the unit of observation. Thus, I answer calls for more research into human factors and team-level investigations in ASD research (Dybå & Dingsøyr, 2008; Moe, Dingsøyr, & Dybå, 2010). Third, I identify design principles that guide the development and construction of methodical extensions of ASD. On the one hand, design principles are derived to extent ASD with concepts and ideas from UCD. This need has been communicated by research (da Silva et al., 2011; Sohaib & Khan, 2010). On the other hand, design principles are derived to clarify early activities in ASD. Early activities, often collectively described as a discovery phase, are rarely specified, and the need to clarify has been indicated in the literature (da Silva et al., 2011; Larusdottir et al., 2017). Fourth, the explanations of identified relationships further advance understandings of ASD and agility. An example is the investigation of team agility as an emergent state. The self-organizing process within a team helps to explain the relationships between different influencing factors within the team, its members, and their context toward team agility.

1.3. Research Design

Overall, the thesis follows a positivist perspective, assuming that reality can be observed and exists independently of its observer (Orlikowski & Baroudi, 1991). The thesis assumes the existence of an objective truth in reality.

Given the identified research gaps (see Figure 1), I have formulated corresponding subresearch questions. Based on the identified subresearch questions, I apply different research methods. Both methodical extensions are based on the literature (SRQ1, SRQ2). The studies follow a systematic literature review approach in order to aggregate and synthesize the existing literature (Kitchenham & Charters, 2007). Thus, both studies derive design knowledge (Gregor & Hevner, 2013). I validated and improved findings using expert interviews (Schultze & Avital, 2011).

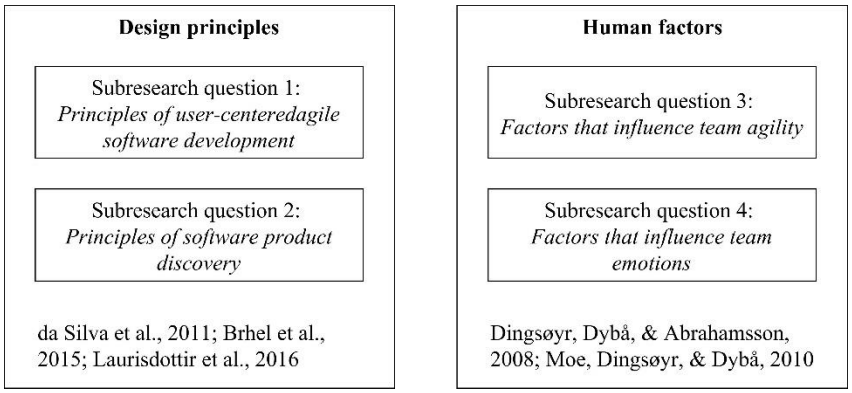

Figure 1 – Overview of Research Questions and Research Results.

Study 3 (SRQ3) investigates a contemporary phenomenon, opting for a positivist multicase study research design (Dubé & Paré, 2003; Yin, 2008). I gathered multiple sources of evidence, and findings result from triangulation of different data points. The study presents within-case analyses of each case and cross-case comparisons.

Study 4 (SRQ4) investigates the emotional influence of team leaders' emotional display on the team's emotional display. Further, I investigate the moderation roles of team collaboration structure and team collaboration function. Thus, I mine an open source repository to extract and analyze large datasets (S. W. J. Kozlowski, Chao, Chang, & Fernandez, 2015; Müller, Junglas, Brocke, & Debortoli, 2016). Following the extraction and enrichment of the dataset, I apply algorithms to identify sentiments and emotions based on documented communication. I investigated and compared different algorithms.

1.4. Thesis Structure

The thesis has six chapters. Following this introduction, I provide an overview of three related research streams: ASD, UCD, and development teams. In the main body, I investigate the methodical extension of and the human factors in ASD teams (Figure 2). Chapter 3 presents Studies 1 and 2. Study 1 investigates SRQ1, identifying principles that reflect commonalities between ASD and UCD approaches to software development. Study 2 derives design principles for a product discovery phase, since this phase has seen little attention in the ASD literature. Chapter 4 describes Studies 3 and 4. In Study 3, team agility is seen as an emergent state in the development team, and I investigate different factors using a multicase study design. In Study 4, I investigate developers and team emotions as predictors for team productivity. This study uses a large dataset from an open source repository. In Chapter 5, I present summary of the findings, theoretical and practical contributions, limitations, and future research avenues. Chapter 6 concludes this thesis.

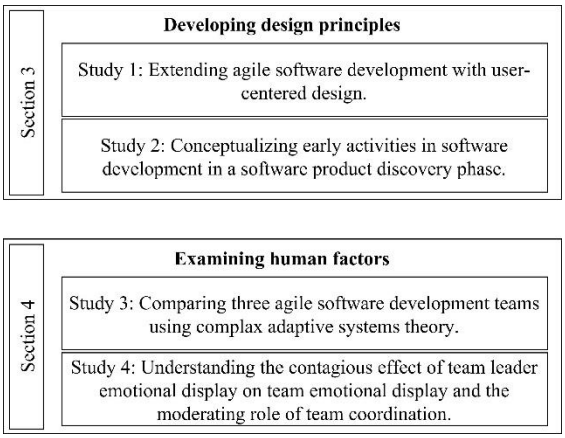

Figure 2 – Overview of the Studies in this Thesis.

2. Theoretical and Conceptual Foundations

To understand the nomological net and the research domains that this thesis contributes to, in this chapter, I provide an overview of the literature. Figure 3 presents an overview of the three research domains: the ASD, UCD, and development team literatures. Further, I point out the gaps in and contributions to the literature concerning the intersections of these domains.

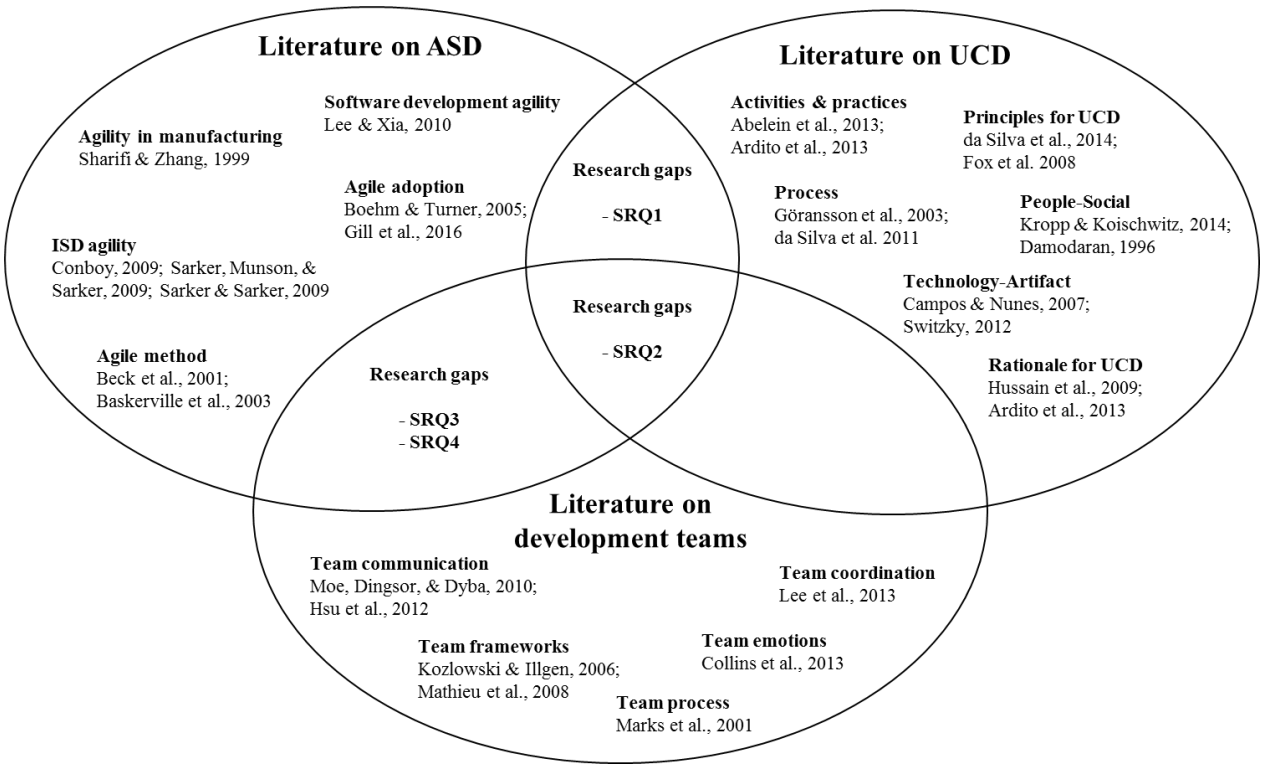

Figure 3 – Overview of the Research Gaps Identified in the Literature.

2.1. Agile Software Development

To value the benefits of agile software development (ASD), one must understand the prior development perspective. Prior to the introduction of ASD, software development evolved from waterfalls and V-models towards incremental and iterative processes, such as RUP or spiral development (Benington, 1983; Boehm, 1988). While these evolutions results from early learning processes that are now common project management knowledge (Brooks, 1975), they led to the adaptation of the development process towards a more flexible, quicker swift approach. This movement also resulted in software development becoming an enjoyable profession rather than merely process execution. In an effort to change this, ASD was introduced by 17 practitioners in 2001 (Beck et al., 2001). The authors' intention was to suggest commonly shared values amongst developers that improve their work environment and philosophy.

Since then, different instantiations of the agile approach have been suggested, many suggested by the original authors of the agile manifesto. These include Scrum and XP, which were among the most popular once (Beck, 1999; Schwaber & Beedle, 2001). Particularly Scrum has gained practical relevance, given its established certification structure and prescription of clear roles, techniques, and tools. Scrum suggests software development in cross-functional and self-organizing teams (Schwaber & Sutherland, 2013) and prescribes specific roles, such as the product owner, who is responsible for maximizing a product's value and managing the releases. A scrum master is established; it is their responsibility to ensure that the development process is understood and inherently enacted. Such enactment includes the time-boxing of development work into sprints and the moderation of synchronized events, such as the daily scrum (a face-to-face standup meeting). XP follows a slightly different approach, suggesting values, principles, and development practices that help a team to develop software. Its values emphasize communication between stakeholders and developers, the assumption that every problem has a simple solution, the focus on quick feedback cycles, focusing development efforts on today's challenges, and respect for others and their work (Beck, 1999).

Research into ASD often focuses on agility. While other disciplines have investigated the concept of agility, for instance in manufacturing or new product development (R. G. Cooper & Sommer, 2016; Sharifi & Zhang, 1999), most of this research has related to software stems from information systems research and software engineering research. The IS literature presents us with two dominant definitions of agility (see Table 1). Agility has been defined both as flexibility and leanness (Conboy, 2009) and as flexibility and evolutionary development (G. Lee & Xia, 2005, 2010; Sarker & Sarker, 2009). In IS research and software engineering research, there are two cross-sectional research streams. On the one hand, scholars research the adoption of ASD; on the other hand, they investigate agile practices. The following describes both streams in some depth.

Table 1 – Overview of Agility Definitions in Information Systems Research.

Definition	Reference
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.	Conboy, 2009, p. 340
A software team's ability to efficiently and effectively respond to and incorporate user requirement changes during the project lifecycle	Lee & Xia, 2010, p. 90
Agility in a distributed ISD setting is a distributed team's capability to swiftly accomplish ISD tasks and to rapidly adapt and reconfigure itself to changing conditions by a) drawing on appropriate IS personnel and technological resources; b) utilizing appropriate ISD methods, mechanisms for bridging temporal distances, and routines to anticipate, sense, and react to changes in the distributed team's project environment; and c) forging and maintaining links across the communicative and cultural barriers among the distributed team members.	Sarker & Sarker, 2009, p. 458

2.1.1. Agile Adoption

Agile adoption research seeks to understand the introduction of ASD (Gandomani & Nafchi, 2015; Laanti et al., 2011; Müller-Wienbergen, Müller, Seidel, & Becker, 2011), its scalability and maturity (Gill, Henderson-Sellers, & Niazi, 2016; Gren, Torkar, & Feldt, 2015; F. S. Silva et al., 2015), and its performance impacts (Chow & Cao, 2008; G. Lee & Xia, 2010; Maruping, Venkatesh, et al., 2009).

The introduction of ASD requires the consideration of different facets (see Table 2). The agile transition and adoption framework presents the relationships of different factors, such as prerequisites, transition facilitators, challenges, and issues (Gandomani & Nafchi, 2015). Such introduction and transition has positive effects on individuals (Laanti et al., 2011). Developers feel higher satisfaction levels as well as more effective and happier following the introduction of ASD. The support, uses, and impacts of this development method have also been shown to be affected by organizational culture (J. Iivari & Iivari, 2011).

Table 2 – Overview of Articles on the Introduction of Agile Software Development.

Reference	Outlet	Overview
J. Iivari & Iivari, 2011	IST	Derives several propositions and hypotheses that explain the effects of organizational culture orientation on software development method deployment.
Laanti, Salo, & Abrahamsson, 2011	IST	ASD leads to higher satisfaction, a feeling of effectiveness, increased quality and transparency, increased autonomy and happiness, and earlier detection of defects. Overall, 60% of responses would not return to the old way of working.
Gandomani & Nafchi, 2015	JSS	Suggests an iterative, gradual, continuous, and value-based adoption framework for agile approaches.

For an organization to grow, ASD needs to scale and gain maturity over time (see Table 3). To scale agile, a hybrid model has been suggested that combines traditional and agile methods (Gill et al., 2016). The challenge is that traditional and macro-level processes increase planning capabilities, while agile is a micro-level planning process that challenges traditional planning ideas. The hybrid approach is the result of a case study and suggests a comprehensive reference architecture (Gill et al., 2016). A conceptual work has integrated CMMI with ASD, concluding that agile practices only suffice for CMMI levels 2 and 3 (F. S. Silva et al., 2015). For higher CMMI levels, other practices are required that are not prescribed by ASD. To measure maturity, an extension of Sidky's adoption framework has been developed (Gren et al., 2015; Sidky, 2007). While the old framework has already suggested various dimensions, such as collaborative planning, task volunteering, or empowered and motivated teams, the extension adds five measurement dimensions (Sidky, 2007); these include the dedication to teamwork and results, agile planning, honest feedback to management, open communication, and leadership style (Gren et al., 2015).

Table 3 – Overview of Articles on the Scalability and Maturity of Agile Software Development.

Reference	Outlet	Overview
Gren, Torkar, & Feldt, 2015	JSS	Develops a measurement instrument based on Sidky's agile adoption framework. Identifies five new factors: dedication to teamwork and results, agile planning, honest feedback to management, open communication, and leadership style.
Silva et al., 2015	IST	Results are grouped into organizational and process-level results. Finds that agile methods apply to capability maturity model integration (CMMI) levels 2 or 3. For CMMI levels 4 and 5, other practices are needed.
Gill, Henderson-Sellers, & Niazi, 2016	ISF	Developed a hybrid adaptive method reference architecture.

Following the introduction of ASD and its potential growth, measuring its performance impacts enables the comparison of results across projects or teams (see Table 4). The perceived success of agile projects along quality, scope, time, and cost is influenced by the delivery strategy, the agile techniques, and the team capabilities (Chow & Cao, 2008). Similarly, the use of agile methods explains an increase in software quality (Maruping, Venkatesh, et al., 2009). This effect is explained by extent of requirements changes, self-control, and outcome control. Other dimensions, such as on-time completion, within-budget completion, and software functionality can be predicted by a software team's response efficiency (G. Lee & Xia, 2010). In addition, the software functionality can be predicted by a software team's response extensiveness. Both software team response extensiveness and efficiency have been used as dimensions of agility. These dimensions can be predicted by team autonomy and team diversity.

Table 4 – Overview of Articles on the Performance Impacts of Agile Software Development.

Reference	Outlet	Overview
Chow & Cao, 2008	JSS	Identifies three critical success factors for agile development projects: delivery strategy, agile software engineering techniques, and team capability.
Maruping, Venkatesh, & Agarwal, 2009	ISR	Predicts variance in four project quality objectives (bug severity, component complexity, coordinative complexity, and dynamic complexity) based on the data of 862 developers in 110 teams.
Lee & Xia, 2010	MISQ	Explains the relationship between software team characteristics (team autonomy and team diversity) and software development performance (on-time completion, within-budget completion, and software functionality), and the mediating effect of software development agility (a software team's response extensiveness and response efficiency). While the survey of 399 software project managers forms the primary data source, a 10-case study design was used post hoc to explain results.

2.1.2. Agile Practices

Agile practices helps developers to embrace the agile philosophy and to instantiate it in their projects. Studies either seek to investigate agile practices comprehensively, or seek to understand a particular practice (see Table 5). The research suggests a comprehensive view towards best practices in agile

development by categorizing them into one of three dimensions (product-related, process-related, or people-related best practices) (Meso & Jain, 2006). Similarly, the knowledge, process, and people dimensions have been suggested in a dependency view of ASD projects (Strode, 2015). The knowledge dimension considers requirements, expertise, historicity, and task allocation within a project. Different activities and business processes indicate the process dimensions. The resource dimension is indicated by the team members (entity) and technical needs. Misra et al. (2009) identified 14 success factors for the successful adoption of agile practices.

A popular agile practice of XP is development within programming pairs who can review and comment on each other's results during the coding process. Prior research suggests that pair programming does not lead to superior quality (Balijepally et al., 2009), yet results suggest that pairs deliver quality at the level of best programmers of nominal pairs. While nine practices relate to organizational factors, five practices relate to individual factors. As a result, the study confirms nine practices that help to explain agile projects' success. Another key practice is collaboration with customers. Research suggests that customer involvement is an important aspect of ASD (Hoda et al., 2011); the lack thereof negatively affects a team's self-organization.

Table 5 – Overview of Articles on Agile Practices.

Reference	Outlet	Overview
Meso & Jain, 2006	ISM	Postulates three dimensions of best practices: product dimension (artifact), process dimension (development), and people dimensions (software team).
Balijepally et al., 2009	MISQ	Programming pairs perform at the level of the best performers in nominal pairs. Thus, there is no *assembly bonus effect*.
Misra & Kumar, 2009	JSS	Customer satisfaction, customer collaboration, customer commitment, decision time, corporate culture, control, personal characteristics, societal culture, and training and learning are nine identified success factors that explain agile project success.
Hoda, Noble, & Marshall, 2011	IST	Suggests a lack of customer involvement and the identification of strategies that seek to cope with insufficient and ineffective involvement of customers.
Strode, 2015	ISF	Develops a dependency taxonomy for agile projects with three dimensions (knowledge, process, resource) and eight corresponding subdimensions (requirements, expertise, historical, task allocation, activities, business processes, entities, and technical)

2.2. User-centered Design

In the literature on user-centered design (UCD), the well-known and established usability standard for human-centered design activities ISO 9241-210 and its predecessor, ISO 13407, have been used by scholars from different fields (ISO, 1999, 2010), such as human-computer interaction, psychology, computer science, and IS (Bevan, 2009). Thus, unsurprisingly, the literature contains different definitions of UCD (see Table 6). ISO 13407 provides rather an overview guidance for the management and planning of user-centered design, than to provide detailed coverage of techniques and methods (ISO, 1999). While ISO 13407

summarizes UCD along activities, principles, planning, and rationale, Barksdale and McCrickard's (2012) framework uses practices, processes, technologies, people, and social types instead. Thus, the literature is classified into practices and activities, principles, processes, rationales, social and people-related factors, and technologies and artifacts. The category *activities and practices* covers practices, activities, and deliverables. Activities and practices are practiced or formalized procedures, which have been proven as utilitarian and successful (Engesser, Claus, & Schwill, 1993), such as story cards or code analysis. Deliverables are an activity's outcomes, for instance, a mock-up that results from a sketching process. Principles reflect generic statements for the application of UCD. The process dimension reflects the processes structure, synchronization, or integration independently of the activity's context, leaving the process modifiable. A process is a specific sequence or regularity of activities. *Social and people-related factors* is a category that pools people's attributes and their characteristics as well as the social environment and the interactions between these people. Example factors include attitudes, education, communication, collaboration, decision-making, roles, and responsibilities. Rationale relates to the reasons and justifications for the adoption and integration of UCD. The last category, *technologies and artifacts*, investigates the use of tools and technologies, for instance, automated evaluations or informal tools. I will now summarize the literature along the six primary dimensions activities and practices, social and people-related factors, principles, processes, rationales, and technologies and artifacts.

Table 6 – Overview of User-centered Design Definitions in Information Systems Research.

Definition	Reference
UCD is "a multidisciplinary design approach based on the active involvement of users to improve the understanding of user and task requirements, and the iteration of design and evaluation".	Mao, Vredenburg, Smith, & Carey, 2005, p. 105
UCD "puts the end users of the system at the center of the design and evaluation activities, through a number of methods and techniques".	Humayoun, Dubinsky, & Catarci, 2014, p. 1
"User-centered design (UCD) software is an approach focusing on making systems usable."	N. Iivari & Abrahamsson, 2002, p. 3260
"User-centered design' (UCD) is a broad term to describe design processes in which end-users influence how a design takes shape. It is both a broad philosophy and variety of methods. There is a spectrum of ways in which users are involved in UCD but the important concept is that users are involved one way or another."	Abras, Maloney-Krichmar, & Preece, 2004, p. 763

2.2.1. Activities and Practices

The stream activities and practices originates from ISO 13407 (ISO, 1999). The subdimensions mentioned in ISO 13407 are user research (e.g. Beyer, 2010; Stewart & Seaton, 1992), conceptualization (e.g. Göransson, Gulliksen, & Boivie, 2003), product design (e.g. Mao, Vredenburg, Smith, & Carey, 2005), and evaluation (e.g. Taylor, Ji, & Yun, 2006); these are well researched. The literature also provides evidence that guides teams, projects, and activities concerning their general methods and concerning balancing formal

and informal methods according to the context (Macaulay et al., 2009). Thus, two additional topics – ambidexterity (e.g. Stary, 1995) and coordination mechanisms (e.g. Jokela, 2004) – are suggested. Ambidexterity, which often comprises alignment and adaptability, describes the use of aligned or adapted methods and the combination of both (Hudson, 2001; Spraragen, 2005; Stary, 1995). Aligned methods are inflexible, formal, or planned activities and practices, such as formal usability tests or weekly meetings, while adapted methods are characterized by flexibility, informality, and opportunities to use them at short notice, such as in informal meetings or informal tests (D. Sy, 2007). The topic is based on studies about the uses of formal and informal methods, and a lack of discussion about ambidexterity, as a key characteristic of methods (Göransson, Gulliksen, et al., 2003). Coordination mechanisms are a universal characteristic of identified practices, since they apply in and for each of the four core activities. Studies have investigated communication and reallocation (Ardito, Buono, Caivano, Costabile, & Lanzilotti, 2013; Beyer, 2010a; Fox et al., 2008), creating cohesion (Jokela, 2004; Jurca, Hellmann, & Maurer, 2014), design chunking (D. Sy, 2007), planning (Gulliksen et al., 2003; Jokela, 2004), and tracking processes (Beyer, 2010a; Jokela, 2004; McInerney & Maurer, 2005). User research is the first of the four core activities, and covers every practice or activity performed during the analysis of users and their contexts. Examples are contextual inquiry (Beyer, 2010a; Evnin & Pries, 2008), environmental analysis (S. Bodker, 1996; Kropp & Koischwitz, 2014; Salah, Paige, & Cairns, 2014c), interviews (Rexfelt & Rosenblad, 2006; Spraragen, 2005; Switzky, 2012), and surveys (Hudson, 2001; Macaulay et al., 2009). Outcomes of user research are i) conceptualized as success criteria (e.g. a product's usability), ii) guidelines, or iii) determined constraints (Jokela, Iivari, Matero, & Karukka, 2003). The activities produce and design transform gathered knowledge into design solutions, including low-fidelity prototypes, high-fidelity prototypes, and running prototypes (Ardito et al., 2013; Fox et al., 2008; Hussain, Slany, & Holzinger, 2009a). Finally, designs are evaluated through evaluative activities and practices in order to assess whether or not they meet the users' requirements (Sohaib & Khan, 2010). There are several prospects of evaluating designs. Examples are pure user evaluation or user participation, but also the assessment without direct user involvement towards more objective evaluation-based heuristics (Ardito et al., 2013; Macaulay et al., 2009).

While the literature often focuses on benefits and successes as a result of the implementation of UCD activities and practices (see Table 7), few cases have discussed challenges for their implementation and adoption. For instance, practitioners tend to use some methods (e.g. use cases) as a holy grail, without considering alternatives. UCD is neither horizontally nor vertically complete, suggesting the need for more research and detailed classification of existing methods (J. Iivari & Iivari, 2006; Rexfelt & Rosenblad, 2006). However, some challenges relating to this subject are often preventable through a better understanding of methods by practitioners. Thus, there is a clear need for UCD guidelines for practitioners (Vredenburg, Mao, Smith, & Carey, 2002).

Table 7 – Overview of Activities and Practices in User-centered Design with Example References.

Activities and practices	Example reference	Mentions of activities and practices in the examples
User research	Abelein, Sharp, & Paech, 2013	Discusses elements of contextual inquiry, such as observations, discussions, and meetings.
Conceptualization	Göransson et al., 2003	Presents different design artifacts that help one to conceptualize, such as guidelines, mockups, personas, scenarios, and use cases.
Produce & design	Ji & Yun, 2006	Articulates different activities that help to create better designs, such as card-sorting exercises, the use of a central design record, or prototypes.
Evaluation	Ardito et al., 2013	Suggests the use of functional tests and heuristic evaluation as examples of usability evaluation. Also, suggests informal usability testing and thinking-aloud as in-person tests.
Ambidexterity	D. Sy, 2007	Discusses ambidextrous elements through alignment vs. adaptability of formal and information methods of UCD.
Coordination	Jokela, 2004	Presents different coordination mechanisms within UCD teams, such as creating cohesion, planning, or tracking processes.

2.2.2. Principles for User-centered Design

In the stream principles for UCD, five general principle are identified (see Table 8): active and continuous stakeholder involvement (Abras et al., 2004; Xu, Creighton, Boulila, & Demmel, 2013), coherent and cohesive design (Beyer, 2010a; Jokela, 2004), flexibility and quick response (Bertholdo, da Silva, de O. Melo, Kon, & Silveira, 2014; Beyer, 2010a), multidisciplinary design (Campos & Nunes, 2007; Vredenburg et al., 2002), and user focus (Göransson, Gulliksen, et al., 2003; J. Iivari & Iivari, 2006). Active and continuous stakeholder involvement comprises the continuous involvement of (end-) users, and all stakeholders involved in the design and development process, for instance, customers, managers, and software developers. The literature distinguishes prospective users into three types: primary, secondary, and tertiary users (Abras et al., 2004). The literature often discusses stakeholder involvement in agile and non-agile contexts (Williams & Ferguson, 2007). The findings are inconsistent and incomparable, since they frequently use the terms user, customer, or stakeholder (Fox et al., 2008; Mao et al., 2005; Xu et al., 2013) without clearly defining and distinguishing them.

While few studies explicitly use the terms coherent or cohesive design (Beyer, 2010a; Gasson, 2003), more studies highlight their importance when discussing iterative, incremental, and evolutionary design (Gasson, 2003; J. C. Lee, McCrickard, & Stevens, 2009). Coherence means the consistency between items or entities, such as the interface parts or technologies used, and describes the extent to which they are equally consistent. Cohesion is the extent of logical interconnection between these items or entities, as well as the extent of conclusiveness. Coherent and cohesive design depends on the available technologies, the exhaustiveness and accuracy of the available user requirements, and can be enhanced using parallel user experience streams (Beyer, 2010a; Gasson, 2003; J. C. Lee et al., 2009). While scholars have discovered a relationship

between little design upfront (LDUF) and cohesive overall design, Beyer (2010) highlights the need to provide space for a deep understanding of users to emerge. According to coordination practices, cohesion within social structures is an additional important factor that need to be considered, especially because it is hard for multidisciplinary teams to be cohesive (Beyer, 2010a; Jokela, 2004; Jurca et al., 2014).

Table 8 – Principles of User-centered Design with Example References.

Principles	**Example reference**	**Mentions of principles in the examples**
Stakeholder involvement	Fox et al., 2008	Suggests the continuous involvement of users and other key stakeholders. The resulting continuous feedback improves the software design.
Coherence and cohesion	Beyer, 2010	Discusses benefits of a coherent design and cohesion between team members.
Flexibility and responsiveness	Bertholdo et al., 2014	Presents the benefits of flexibility and responsiveness, such as an increased ability to react to changes.
Multidisciplinary design	Mao et al., 2005	Indicates the benefits of having different specialists participate in the design process.
User focus	Göransson et al., 2003	Suggests the involvement of users in order to increase usability and to consider environmental factors.

Flexibility and responsiveness are key pillars stemming from ASD (Beck et al., 2001). They suggest the prioritization of informal, dynamic, and flexible methods over formal, fixed, and inflexible methods. Thus, projects achieve short reaction times during project delivery, typically performed in sprints or cycles. In the UCD literature, both concepts are often combined with an ambidextrous lens, for instance, flexible vs. inflexible or responsive vs. static (Bertholdo et al., 2014; Beyer, 2010a; D. Sy, 2007). Implementations are not limited to practices, but also extend to technologies and artifacts as well as social and people-related factors (D. Sy, 2007), as expressed in the valuation of "individuals and interactions over processes and tools" (Beck et al., 2001, p. 1) or "highly interactive and highly interpersonal interactions over formal documents and structured interactions" (Beyer, 2010, p. 6).

Multidisciplinary design, as mentioned in ISO 13407 (ISO, 1999), deals with the involvement of different disciplines and functions during the design process. Usually, multidisciplinary design is seen in terms of the team members' disciplinary backgrounds, which would lead to multidisciplinary design as social and people-related factors (A. Cooper, 1999; Gulliksen et al., 2003). However, multidisciplinary design is a function of the entire process and includes every aspect of it.

Few articles have addressed user focus (Boivie, Gulliksen, & Göransson, 2006; Göransson, Gulliksen, et al., 2003; Salah et al., 2014c), suggesting the inclusion of user focus through user involvement, which is addressed much often. While ISO 13407 explains the focus of UCD as "an appropriate allocation of

functions between users and technology" (ISO, 1999), the definition of ISO 9241-210 as an "approach to systems design and development that aims to make interactive systems more usable by focusing on the use of the system and applying human factors/ergonomics and usability knowledge and techniques" provides a better articulation (ISO, 2010). Here, UCD clearly focuses on the need to consider the entire user experience based on understandings of users, tasks, and the environment. Thus, the literature has investigated the environment (Gulliksen et al., 2003), usability (Göransson, Gulliksen, et al., 2003), user experiences (Detweiler, 2007), or users (J. Iivari & Iivari, 2006).

When investigating challenges relating to the articulated principles of UCD, most studies have focused on user involvement or user focus (Damodaran, 1996; J. Iivari & Iivari, 2006). User involvement challenges during the design process relate to user selection (Damodaran, 1996; Gulliksen et al., 2003), insufficient user representation (Gulliksen et al., 2003; J. Iivari & Iivari, 2006), and insufficient environment analysis (Sohaib & Khan, 2010). Challenges relating to user focus indicate the general lack thereof, or an incorrect focus (J. Iivari & Iivari, 2006). Many firms still struggle to consider users, users' environments, and entire usability when designing and developing a new product (Sohaib & Khan, 2010). Despite focusing on the a product's de facto users, firms often derive product needs either from an internal group of individuals or from a limited group of users (Jokela et al., 2003). There is still no general solution to the selection of the right users and to keeping them involved during the entire design process. Different approaches to user involvement have been suggested, from the physical involvement of every user (Göransson, Gulliksen, et al., 2003), to identifying a focus group (Abras et al., 2004), to engaging a usability designer (Boivie et al., 2006), to identifying fictitious users (J. Iivari & Iivari, 2006). There has been less guidance on how to select the best approach. Based on this information gap, practitioners underestimate the de facto benefits, first opting for the most convenient approach. Thus, organizations often design and develop of a product without access to physical users. While it is impractical to involve every user, research must compare different ways concerning profitability, project success, user and customer satisfaction, and time durations. Practitioners need tailored guidance for the correct selection and involvement of users, given their current situation and context.

2.2.3. Processes

The literature on UCD as a process describes design as a subset of the development process (Göransson, Gulliksen, et al., 2003). Topics in this literature stream include (see Table 9) incremental and evolutionary tracks (Begier, 2011; Garrity, 2001), iterative tracks (Gould & Lewis, 1985; ISO, 1999, 2010), parallel tracks (Gasson, 2003; Salah et al., 2014c), and a separate discovery phase (Gasson, 2003; Hochmüller & Mittermeir, 2008). While iterative tracks are considered to be a principle in ISO 9241-210 (ISO, 2010); here, it is the understanding that only subprocesses can be iterative; therefore, the iterative characteristic is related

to specific subprocesses rather than the entire UCD process. Yet, iterations become more powerful when they connect different practices (e.g. prototyping and evaluation) rather than iterating a single practice (e.g. prototyping). Thus, it is important to customize each process to an organization (Göransson, Gulliksen, et al., 2003; Gulliksen et al., 2003; J. Iivari & Iivari, 2006). Besides the concatenation of practices and activities, it is also necessary to evaluate an overall approach's suitability, for instance, agile vs. spiral vs. waterfall. Owing to their different origins, combining two approaches can create additional challenges (Ardito et al., 2013). Often, the decision to use a waterfall lifecycle model depends on the project managers' backgrounds or project constraints (Beyer, 2010b). While current trends indicate the shift towards iterative and incremental design (Losada, Urretavizcaya, & Fernández-Castro, 2013), practitioners lack guidance to how to incorporate UCD into agile processes (Miller & Sy, 2009).

Table 9 – Process Characteristics in User-centered Design with Example References.

Process characteristic	**Example reference**	**Mentions of process characteristics in the examples**
Incremental-evolutionary tracks	Göransson et al., 2003	Suggests that incremental and evolutionary development help one to focus on smaller parts while increasing the quality in future cycles.
Iterative tracks	Carter, 1999	Continuous iterations and assessment help to avoid misinterpretations. The article introduces usability first and suggests the development of usage models.
Parallel tracks	Macaulay et al., 2009	The parallelization of different tracks enhances performance. Yet, organizations must consider integration and synchronization efforts.
Separate discovery	da Silva et al., 2011	The introduction of a discovery phase through little or big upfront design ensures basic user research activities.

Incremental and evolutionary creation influences the design, development, and delivery process. Despite the most exhaustive user research, design might dissatisfy users and stakeholders, because users sometimes find it challenging to properly articulate their needs (Begier, 2011; Beyer, 2010a; Garrity, 2001; Göransson, Gulliksen, et al., 2003; Xiong & Wang, 2010a). Thus, incremental and evolutionary design offers the possibility to focus on parts of the design process. Thus, a team has a better focus, requires less resources to deliver a design, while maintaining in-depth user analysis in order to determine new product features and the success thereof (Begier, 2011; Göransson, Gulliksen, et al., 2003; Gulliksen et al., 2003).

The literature introduces the idea of iterative design and development, which is central to the nature of UCD, as suggested by different scholars (Gould & Lewis, 1985; Gulliksen et al., 2003). Similarly, iterative design follows established characteristics of ASD, which has attained increasing popularity (Beyer, 2010a; Humayoun et al., 2014a; Raison & Schmidt, 2013). Iterative design is often discussed jointly with incremental and evolutionary design. For instance, scholars suggest the use of iterative design and the evaluation of results before the main development starts, so that failures can be identified early (Fox et al.,

2008; Mao et al., 2005). To ensure this throughout the process, continuous and cyclical iterations are suggested (Gasson, 2003; Gulliksen et al., 2003; Rexfelt & Rosenblad, 2006). While design and development complement each other, they require different expertise. Thus, prior studies suggest conducting these activities in parallel and recommend joint use with iterations in order to design consistent information flows (Beyer, 2010a; Fox et al., 2008; Hussain, Slany, et al., 2009a; Salah et al., 2014c; D. Sy, 2007). Yet, parallel design and development requires more communication and coordination efforts, with the goal to synchronize and integrate both tracks (Humayoun, Dubinsky, & Catarci, 2011; Hussain, Slany, et al., 2009a; Salah et al., 2014c).

The notion of holistic design, i.e. the parallel development of every design-affecting aspect simultaneously, can be identified in the literature (Gulliksen et al., 2003). Holistic design increases the need for a coherent and cohesive design, since a separation would render partial outcomes irrelevant to the other participants in the design and development process (Gasson, 2003). The development track often requires input from the design track. Scholars describe this dependency as deferred tracks, referring to the notion of conducting the design activities one or two iterations before the corresponding development activities (e.g. Bertholdo et al., 2014; Raison & Schmidt, 2013; Salah et al., 2014c; Ungar & White, 2008; Xiong & Wang, 2010a).

Studies on to the UCD process also recommend prior investigation and upfront design activities, sometimes also referred to as *cycle zero*. While literature on ASD favors LDUF (Ungar & White, 2008), the UCD literature suggests more big design activities upfront (Xiong & Wang, 2010b). Given UCD's potential impacts on a product's long-term success, the extent of appropriate upfront design activities remains undetermined and is part of an ongoing controversial debate among scholars (Hochmüller & Mittermeir, 2008).

When discussing design in the context of software development, design is often perceived as an add-on to the development process. Scholars suggest a lack of guidance concerning the combination of single activities into a holistic process (Gulliksen, 2007). For many developers, it remains unclear when design activities should take place (Boivie et al., 2006). Doing UCD as single activities prevents the creation of a horizontally and vertically integrated process. Given the lack of a integrative end-to-end approaches (Humayoun et al., 2011), practitioners need a customized and modular approach that is easy and flexible to adopt.

2.2.4. Social and People-related Aspects

The literature stream on social and people-related aspects discusses the challenges and benefits of interpersonal and hierarchical questions in UCD (see Table 10). Team members' different professional backgrounds often leads to different attitudes (Damodaran, 1996; Gulliksen et al., 2003), communication needs (Garrity, 2001; Suchman, 1995), collaboration (S. Bodker, 1996; Carter, 1999), decision-making (Beath & Orlikowski, 1994; Gasson, 1999), and learning (Garrity, 2001; Kropp & Koischwitz, 2014) while designing a successful product. For each process, the literature presents input and outcome factors.

Table 10 – Social and People-related aspects of User-centered Design with Example References.

Social and people-related aspects	**Example reference**	**Mentions of social and people-related aspects in the examples**
Attitudes	Damodaran, 1996	A professional and positive attitude to user involvement increases the willingness to represent end-users properly during the design process.
Decision-making	Beath & Orlikowski, 1994	The active involvement of the users does not guarantee their engagement as equals during the design process.
Education	Kropp & Koischwitz, 2014	The exchange of knowledge through a learning process from users to experts and vice versa is key.
Collaboration	S. Bodker, 1996	Active collaboration, for instance through a cooperation committee, is important.
Communication	Garrity, 2001	Communication is key for intra-team and inter-team information and knowledge exchange.

Attitudes is an important factor that requires consideration throughout the UCD process when engaging internal and external stakeholders (Damodaran, 1996; Gulliksen et al., 2003). While professional, positive, and user-centered attitudes create commitment, support, and willingness to properly represent end-users (Damodaran, 1996; Gulliksen et al., 2003; J. Iivari & Iivari, 2006), users' attitude affect their willingness to cooperate and be involved in the development process (Amoako-Gyampah, 1997). Top managers' attitudes influences the organization as a whole and signals the extent of its commitment (Damodaran, 1996).

Scholars have also suggested differences in attitudes to users when comparing user participation with user involvement. The former does not imply the inclusion of users as equal participants, creating an imbalance in power (Beath & Orlikowski, 1994; Bjorn-Andersen, 1988; Gasson, 1999). Such imbalance can be prevented by giving users real authority during the design process. It is also important to clearly define roles and responsibilities, and to address power constraints (Abras et al., 2004; Damodaran, 1996; Gasson, 1999). An underlying reason for people's disrespect for other roles is a lack in education. Various methods, such as (design) workshops, UCD training sessions, newsletters or other mutual and reciprocal learning

techniques have been suggested to resolve existing prejudices (e.g. S. Bodker, 1996; Garrity, 2001; Gasson, 1999; Jokela, 2004; Jurca et al., 2014; Macaulay et al., 2009). Educating colleagues about the importance of users and their roles increases appreciation for UCD (Damodaran, 1996; Ungar & White, 2008).

Communication is the basis for any collaboration, agreement, and knowledge transfer in a team and across stakeholders (Jurca et al., 2014). Several authors consider informal communication to be beneficial (Gulliksen et al., 2003; Macaulay et al., 2009; McInerney & Maurer, 2005). Information and requirements should be communicated in useful ways, for instance, through the use of prototypes, use cases, or other artifacts (Beyer, 2010a; Boivie et al., 2006; Nebe & Grötzbach, 2006). Generally, scholars agree on the benefits of using artifacts that facilitate communication and support understanding by all stakeholders (Abras et al., 2004; Boivie et al., 2006; Göransson, Gulliksen, et al., 2003; Gulliksen, 2007; Gulliksen et al., 2003; J. Iivari & Iivari, 2006; Jurca et al., 2014; Macaulay et al., 2009; Nebe & Grötzbach, 2006).

Collaboration has gained much attention by scholars investigating its role in the UCD context. The interpretations vary from stakeholder collaboration, to the collaboration with groups or single individuals. Scholars have stressed collaboration and have introduced the creation of a cooperation committee to embed the inclusion of key stakeholders (S. Bodker, 1996; Carter, 1999). While stakeholders in the UCD context often refer to customers and users, studies suggest the importance of involving managers, given their organizational and social influence, for instance, by contagion (Ardito et al., 2013). However, management involvement can inhibit intended collaboration (Gasson, 1999). While scholars agree on the benefits of face-time during collaboration (Williams & Ferguson, 2007), they disagree on the intensity of such collaboration (Garrity, 2001; Suchman, 1995).

Problems mentioned in this stream are often rooted in communication issues, such as misinterpretation or failed expectation management (Gulliksen et al., 2003). Attitude or awareness problems often appear in such communication challenges (Gulliksen et al., 2003). A bad attitude expressed through a lack of commitment or disrespect (Garrity, 2001) makes it difficult to communicate effectively in a team. Similarly, a lack of awareness fuels existing communication challenges (Abras et al., 2004). Social challenges are often rooted in a lack of education on the part of participants or their lack of knowledge about responsibilities (Garrity, 2001; Gulliksen et al., 2003; Hudson, 2001), collaboration (Gasson, 1999), roles (Pancake, 1997), and culture (N. Iivari, 2004; N. Iivari & Abrahamsson, 2002). Studies have identified key antecedents of social and people-related aspects and their effects on outcomes. Particularly, personal settings such as attitudes, awareness, education, knowledge, and role clarity cause many social problems and interpersonal conflicts (Salah et al., 2014c). Personal matters are an underestimated factor in this nomological net. Given the investigation of antecedents, unsurprisingly, these can negatively affect team processes and

outcomes (Salah et al., 2014c). In addition, a company's power structure can fuel additional problems (Gasson, 1999). Regarding processes, strict guidance concerning communication and collaboration seems advisable (Sutcliffe, Thew, & Jarvis, 2011; Ungar & White, 2008). For instance, uniform standards on communication and collaboration, as well as making them understandable for every participant, are one way to address this challenge.

2.2.5. Technologies and Artifacts

Another research stream investigates the uses of technologies and artifacts during the design process (see Table 11), such as automation (Beyer, 2010a; McInerney & Maurer, 2005), remote testing (Hussain, Slany, et al., 2009a; Switzky, 2012), IDE integration (Humayoun et al., 2011), and tools (Humayoun et al., 2011). Automation can be achieved through automated evaluation and automated testing. While it is a lesser known method, automated usability evaluation is already known in the agile (Hussain, Slany, & Holzinger, 2009b). Automated testing has found greater acceptance in the developer community (Beyer, 2010b; McInerney & Maurer, 2005). Particularly with the rise of crowdsourcing, remote usability testing has gained more attention (Hussain, Slany, et al., 2009b). Studies have investigated the integration of these techniques into the integrated development environment of developers (Campos & Nunes, 2007; Humayoun et al., 2011). Prior work on tools can be segmented by notation, tool usage, and collaboration style (Campos & Nunes, 2007). Tools assist developers who are integrating usability standards and guidelines into the process (Carter, 1999). Collaboration tools support communication and collaboration between different stakeholders, and guide the ideation and creation processes (Campos & Nunes, 2007).

The literature has presented only a few concerns relating to technologies and artifacts. Few tools support UCD as a method, particularly in light of UCD being horizontally and vertically incomplete (Hudson, 2001; J. Iivari & Iivari, 2006).

Table 11 – Technologies and Artifacts in User-centered Design with Example References.

Technologies or artifacts	**Example reference**	**Mentions of technologies or artifacts in the examples**
Automation	Beyer, 2010	Suggests the use of automated testing as part of acceptance testing and quality assurance.
IDE integration	Campos & Nunes, 2007	Presents the integration of usability evaluation capabilities based on an extended development environment (IBM's Rationale Rose).
Remote testing	Switzky, 2012	Conducts a remote usability test by inviting users through Twitter to complete eight tasks.
Tools	Campos & Nunes, 2007	Uses a collaboration tool called CanonSketch.

2.2.6. Rationale for User-Centered Design

Finally, one research stream investigates reasons and rationales for the integration of UCD (see Table 12). Rationales concerning inputs relate to process enhancements, resource needs, and user involvement. Process enhancement articulate strengths and weaknesses of specific methods, such as the increasing or decreasing of needed resources (Ardito et al., 2013; Rosenbaum, Rohn, & Humburg, 2000) in terms of money (Vredenburg et al., 2002; Xiong & Wang, 2010a), people (Ames, 2001), and time (Mao et al., 2005). Several authors propose user involvement for various reasons (Begier, 2011; Gasson, 1999; Vredenburg et al., 2002). UCD's outcomes can be positive (Ames, 2001; Begier, 2011; Gasson, 1999) or negative (Abras et al., 2004; Gulliksen et al., 2003). The involvement of users is integral, since the incorporation of their purposes, requirements, and needs into the development process leads to products that are more safer as well as more effective and efficient (Abras et al., 2004).

The literature often focuses on either weaknesses or strengths. Yet, it is important to talk about methods' strengths and weaknesses so as to make them comparable, and to create awareness about their usefulness to users and practitioners. Many statements highlight only positive or negative aspects concerning UCD, often limiting themselves to short-term impacts (Abras et al., 2004), particularly when investigating resource needs. Unilateral statements do not address users and practitioners' concerns. For instance, user involvement has been proven concerning product assessment and the mitigation of uncertainty (Begier, 2011). Involving users creates strong trust, motivates users to participate more, and increases knowledge about personal needs (Carter, 1999; Gasson, 1999); further, it is positively related to product success (Gasson, 1999). While user involvement does not necessarily relate to system success, it positively influences users' perceptions of a product's usefulness (Amoako-Gyampah & White, 1993; Baroudi, Olson, & Ives, 1986; Gasson, 1999). UCD increases workload, but also achieves user and job satisfaction (Ames, 2001). UCD is often considered resource-intensive (Ardito et al., 2013; Rosenbaum et al., 2000), while improving the products' usability and usefulness (Mao et al., 2005), users' satisfaction, and product competitiveness (Hussain, Slany, et al., 2009b; Ronggang Zhou, Shengshan Huang, Xiangang Qin, & Huang, 2008). There is broad consensus about UCD's positive influence on product quality, product utility, product usability, users' perceptions of system success, users' product satisfaction, and job satisfaction (Ames, 2001; Begier, 2011; S. Bodker, 1996; Gasson, 1999; Gulliksen et al., 2003). In short, there is a clear necessity to make methods and approaches comparable to users and practitioners in order to provide simple and tangible reasons for their adoption.

Table 12 – Rationales for User-centered Design with Example References.

Rationale	Example reference	Mentions of rationales in the example
Process enhancement	Vredenburg et al., 2002	Investigates the benefits and weaknesses of different UCD methods. Focus groups are considered the most beneficial, and task analyses the least beneficial.
Resources	Ardito et al., 2013	Finds that UCD is often considered resource-intensive owing to a lack of knowledge.
Product quality	Hussain, Slany, et al., 2009	Suggests that UCD improves user satisfaction and product competitiveness.
User involvement	Carter, 1999	Finds that high involvement increases trust in the organization and increases the design activities' quality.

2.3. Software Development Teams

Teams have been researched in various contexts (Sundstrom & McIntyre, 2000), such as military operations or firefighting. In general, their common denominator is a goal (Levi, 2014) or a shared interest among team members (Marschak & Radner, 1972). A team's objective is to generate an outcome that is useful for the organization (Goodman, 1986). A team's outcomes are often measured by team performance or team effectiveness (Mathieu, Maynard, Rapp, & Gilson, 2008). A more comprehensive definition of teams is presented by S. Kozlowski and Ilgen (2006), who define teams as "(a) two or more individuals who (b) socially interact (face-to-face or, increasingly, virtually); (c) possess one or more common goals; (d) are brought together to perform organizationally relevant tasks; (e) exhibit interdependencies with respect to workflow, goals, and outcomes; (f) have different roles and responsibilities; and (g) are together embedded in an encompassing organizational system, with boundaries and linkages to the broader system context and task environment" (p. 79).

To study team performance, scholars have built on the idea of the input-process-output (IPO) model (McGrath, 1964). The model has since received widespread attention and has been subject to several incremental improvements (e.g. Hackman & Oldham, 1980; S. Kozlowski & Ilgen, 2006; J. Mathieu et al., 2008). However, the model has been criticized for its lack of detail regarding the process dimension (Mathieu et al., 2008). As a result, the IPO framework has evolved and enhancements have been suggested (see Figure 4). An example is the input-mediator-output-input (IMOI), which mitigates the IPO model's shortcomings (Ilgen, Hollenbeck, Johnson, & Jundt, 2005). A key difference is the framework's focus on the different factors' effects. An adaptation of the model towards affective processes suggests an input-process-state-outcome (IPSO) relationship, while the state is an emergent affective state (Collins, Lawrence, Troth, & Jordan, 2013). Another example is the team effectiveness model, which suggests a circular characteristic of an adjusted version of the IPO model, so that team effectiveness as the outcome in turn influences the input through contextual contingencies and/or environmental dynamics (S. Kozlowski

& Ilgen, 2006). The model also suggests levers that shape and align team processes. Example levers are team learning and team leadership.

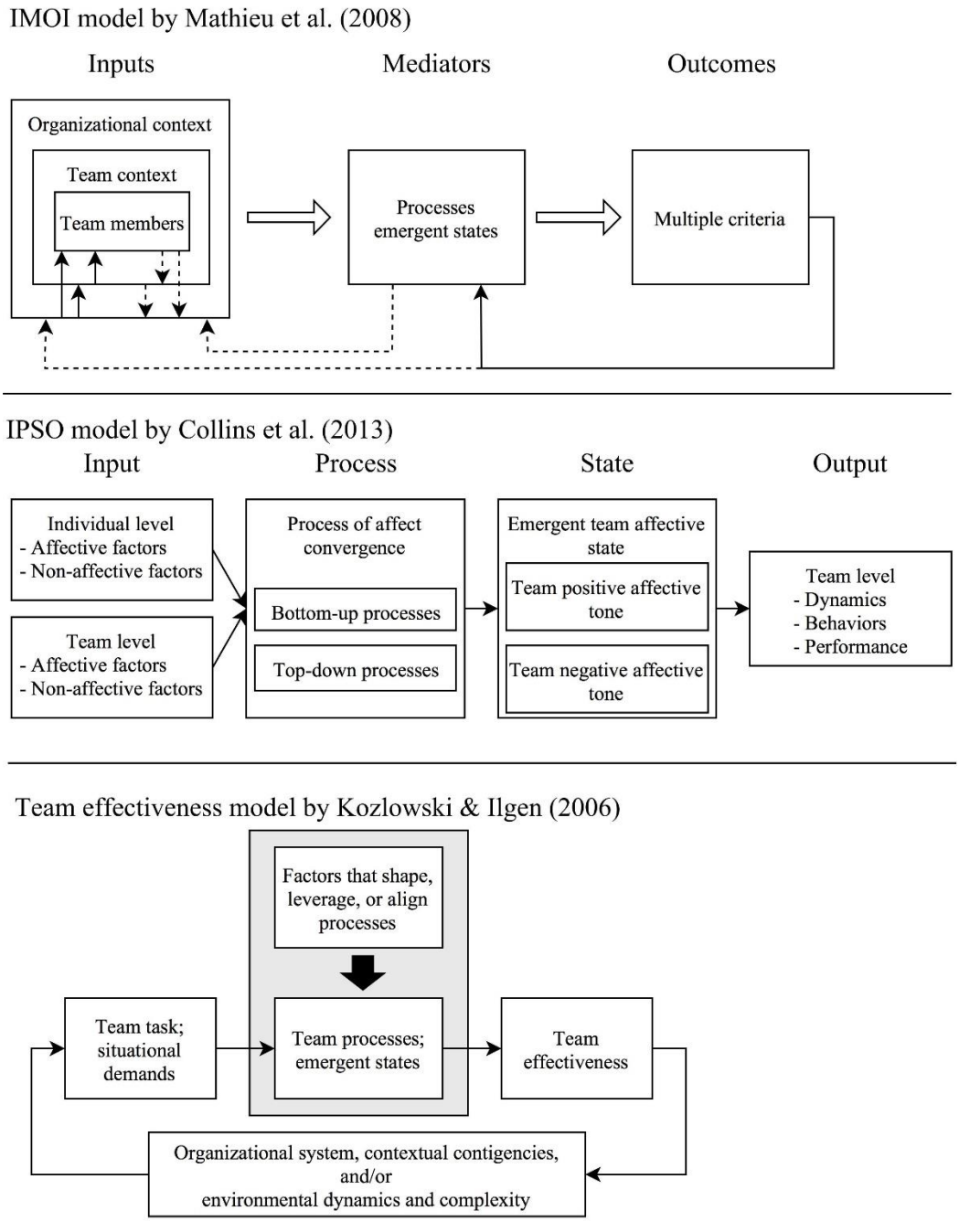

Figure 4 – Overview of Models from Team Research.

2.3.1. Input

In team research, different characteristics influence the team process or the emergent state. Team characteristics are often subsumed as lower-level factors of team design. Team size (de O. Melo, S. Cruzes, Kon, & Conradi, 2013; Misra et al., 2009) is often used as a control variable, measured either in fixed (Misra et al., 2009) or relative terms (Wageman, Hackman, & Lehman, 2005). Team diversity and team dispersion are known influences on the team process. Team diversity refers to team members' variance along different factors, such as role, tenure, nationality, or gender (He, Butler, & King, 2007; Kearney, Gebert, & Voelpel, 2009). Yet, others conceptualize it in terms of interpersonal compatibilities (de O. Melo et al., 2013). Team dispersion is often divided into time, space, and cultural elements (Alzoubi & Gill, 2014; Holmstrom et al., 2006). Team member personalities can influence team processes through increasing agreeableness and reducing social distances (Acuña, Gómez, & Juristo, 2009; Balijepally, Mahapatra, & Nerur, 2006). Unlike team design, team task influences team processes. In the team task design, complexity (Alzoubi & Gill, 2014; G. Lee, Espinosa, & DeLone, 2013) and interdependence (Acuña et al., 2009; G. Lee et al., 2013) are investigated task characteristics. Depending on the task design, a task may be incomplete and may enable different autonomy levels (Wageman et al., 2005).

2.3.2. Processes and Emergent States

Team processes and emergent states mediate the relationships between input and outcomes (Collins et al., 2013; Ilgen et al., 2005). Thus, they explain the effects of different input factors (e.g. team design) on team outcome (e.g. performance). Generally, scholars distinguish between three levels of team processes and emergent states (S. Kozlowski & Ilgen, 2006). First, the literature presents research on behavioral team processes, such as communication and coordination. Second, researchers have investigated emotional and affective level processes within the team. Third, cognitive processes and structures, such as shared mental models, are investigated. While team processes have been researched for decades, the first formal definition only appeared in 2001, from Marks et al. (2001), who define a team process "as members' interdependent acts that convert inputs to outcomes through cognitive, verbal, and behavioral activities directed toward organizing task work to achieve collective goals" (p. 357). On the other hand, emergent states are "constructs that characterize properties of the team that are typically dynamic in nature and vary as a function of team context, inputs, processes, and outcomes" (Marks et al., 2001, p. 357).

Examples of behavioral team processes are communication, coordination, and collaboration. Coordination is defined as "managing dependencies between activities" (Malone & Crowston, 1994, p. 90). It has been suggested that coordination is a key mediator between task environment complexity and development success (G. Lee et al., 2013). Communication helps to build cohesion among team members, yet it is increasingly difficult to achieve in globally dispersed teams (Holmstrom et al., 2006). It is also negatively

correlated to team size, since an increase in team size also increases the need for team communication efforts (de O. Melo et al., 2013). Collaboration is affected by personalities, yet strongly influences a team's effectiveness, since it influences their problem-solving capabilities (Balijepally et al., 2006). Team collaboration can be influenced by team leadership.

Trust, cohesion, affect, and conflict are examples of affective and emergent states. The research shows a positive effect of personality and task interdependency on team cohesion (Acuña et al., 2009). The study also found that task conflicts and social conflicts are negatively associated with team satisfaction (Acuña et al., 2009). Since trust is particularly difficult to establish in a dispersed setting, a study has investigated different factors that increase or prevent the emergence of trust in agile projects (Dorairaj & Noble, 2013). It reports seven concepts that build trust within a team; examples include social communication, evidence of expertise, and collaboration initiatives. Affective processes have also received increasing interest in the research community (Barsade & Gibson, 2007; Barsade & Knight, 2015; Collins et al., 2013).

Examples of cognitive processes and structures are shared mental models and transactive memory systems. While a mental model explains, describes, and forecasts an entity's thoughts, a shared mental model described the aggregation of joint mental models across team members (Xiang, Yang, & Zhang, 2016). A shared mental model is important for teams, so that all team members work towards the same goals (Moe et al., 2010; Salas, 2005). Besides a shared mental model, research has investigated the team as a transactive memory system (Ren & Argote, 2011; Z.-X. Zhang, Hempel, Han, & Tjosvold, 2007). A transactive memory system describes a collection of individuals (e.g. a team) with their use, encoding, and retrieving of information using systems thinking (Ren & Argote, 2011). Both are considered representations of a team's memory (Spohrer, Gholami, & Heinzl, 2012). In a study of 51 development teams, researchers have investigated the formation of team cognition, measured as awareness of expertise location and shared task understanding, based on team communication and diversity (He et al., 2007).

According to S. Kozlowski and Ilgen (2006), different levers influence a team process; examples include different training strategies, team development, team leadership, and performance regulation. One lever, team leadership, has received particular attention in research into development teams (Barsade & Knight, 2015). Team leadership has been found to promote team cooperation and collaboration (Flouri & Berger, 2010). Similarly, prior studies have suggested that good coaching behavior by a team leader increase knowledge sharing in the team (J. Lee, Lee, & Park, 2014; Srivastava, Bartol, & Locke, 2006). While agile teams are expected to have a certain extent of self-managing capabilities (Dybå & Dingsøyr, 2015), prior studies found that the leadership role is often taken on by the scrum master or product owner (Moe et al., 2010). Early studies of team leaders' emotional affects suggest reduced team member turnover when the team leader exhibits positive affects (J. M. George & Bettenhausen, 1990).

2.3.3. Outcomes

In the team literature, different indicators for team outcomes have been suggested. Scholars have often measured effects on performance or effectiveness (e.g. J. E. Mathieu, Gilson, & Ruddy, 2006; Nguyen-Duc, Cruzes, & Conradi, 2015). The project literature often defines performance as the sum of effectiveness and efficiency. However, team research has often investigated team effectiveness as the overarching concept and has used different outcomes to measure it. For instance, it has been suggested that team effectiveness is the sum of performance and viability (Sundstrom, De Meuse, & Futrell, 1990); performance and satisfaction (Goodman & Shah, 1992); or performance (e.g. efficiency, quality, and customer satisfaction), attitudes (e.g. employee satisfaction and trust in management), and behaviors (e.g. turnover and safety) (Cohen & Bailey, 1997).

Alternative outcomes are emotions, productivity, and satisfaction. Productivity is a more fine-grained performance indicator. Prior studies define productivity in the context of software development "as the amount of work results divided by the effort spent" (Vanhanen & Lassenius, 2005, p. 338). A study of team cohesion and team conflict as team processes has explained variances in team satisfaction (Acuña et al., 2009). Especially research into affective processes considers the identified emotion as the results (Kelly & Barsade, 2001). Here, emotions can be measured discretely or dimensionally. The differential emotions theory suggests 10 fundamental emotions that form the human motivation system (Izard, 1977). Interrelationships between emotions are described in the circular model of emotions (Plutchik, 1980). A comprehensive framework with six primary emotions is presented by Parrott (2001). The dimensional view is based on the two dimensions of (dis)pleasure and arousal/sleep (Russell, 1980). However, the recent literature simply refers to the dimensions as valence and arousal (Russell, 2003). Especially the combination of emotions with big data analytics opens new avenues for future research into emotions and sentiments relating to software development, as recent studies suggest (Murgia, Tourani, Adams, & Ortu, 2014; Tourani, Jiang, & Adams, 2014).

3. Design Principles for Agile Software Development

Design principles are generic statements, allowing for a rule based examination of methods (Gregor & Hevner, 2013). These principles have the objective to develop methodical extension of ASD by combining it with other well-established methods (R. G. Cooper & Sommer, 2016; T. S. da Silva, Martin, et al., 2011; Sohaib & Khan, 2010). These extensions are closely linked to practical problems. In this chapter, study 1 explores the extension of ASD with practices and processes of UCD. Building on study 1, study 2 investigates early activities of the development process, where agile methods such as Scrum or XP provide little to no guidance.

3.1. Exploring Principles of User-Centered Agile Software Development: A Literature Review[1]

3.1.1. Introduction

The question of how to meet the challenge of organizing software development activities has concerned researchers and practitioners ever since software engineering (SE) emerged as an independent scientific discipline in the 1970s (Brooks, 1975; Royce, 1970). In the last decade, the mechanistic view of software development prevalent in earlier phase-based linear approaches has been replaced by an understanding of development activities as a dynamic process characterized by iterative cycles and the active involvement of all stakeholders (Nerur & Balijepally, 2007). This is reflected in *agile software development* (ASD), which responds to unpredictable change by relying on people and their creativity rather than on processes (Cockburn, 2006), and limit software development strictly to activities that add business value for the customer (Beck, 1999; Conboy, 2009; Schwaber & Beedle, 2001).

With agile methods becoming mainstream even for large-scale organizations in the software industry (Blau & Hildenbrand, 2011; Boehm, 2011), software is being delivered on time and in budget, and customer demands are being met increasingly often (Batra, Xia, VanderMeer, & Dutta, 2010; Brian Fitzgerald, 2012). Nevertheless, agile methods focus on the question of how *useful* software can be developed, with customer value being understood as primarily driven by providing an appropriate functional scope. They do not necessarily focus on developing software that is considered *usable* (Blomkvist, 2005; Constantine, 2002; Ferreira, Noble, & Biddle, 2007a), i.e. usability defined as the extent to which a software can be used by specified users to achieve specified goals effectively, efficiently, and satisfactorily in a specified use context (ISO, 1998). While usability is not a central topic in SE, in which it is considered one of many non-functional requirements and quality attributes (Seffah, Desmarais, & Metzker, 2005), it has become crucial for economic success in highly competitive markets and can be used to set the product apart from that of the

[1] This section is based on Brhel, Meth, Maedche, and Werder (2015)

competition (Mayhew & Tremaine, 2005; Scheiber et al., 2012; Seffah & Metzker, 2004). *User-centered design* (UCD) ensures that the goals and needs of the system's end-users is the focus of the product's development. In the field of human-computer interaction (HCI), terms like UCD, usability engineering, and interaction design often have a very similar meaning (T. S. da Silva, Martin, et al., 2011). UCD is driven by continuous end-user evaluation and the iterative refinement of design concepts and prototypes (Fox et al., 2008).

Given the need to deliver business value to the customer in a rapidly changing environment while taking the needs of end-users into account, the integration of UCD and ASD seems to be a promising endeavor and has received increasing attention in recent years (Barksdale & McCrickard, 2012; T. S. da Silva, Martin, et al., 2011; Sohaib & Khan, 2010). In contrast to plan-driven SE, whose properties often impede the integration of UCD (Costabile, 2001; Göransson, Lif, & Gulliksen, 2003), similarities between ASD and UCD provide a common ground, which eases integration. Moreover, owing to these similarities, a multitude of integration approaches has been suggested in the literature (T. S. da Silva, Martin, et al., 2011; Fox et al., 2008; D. Sy, 2007). Thereby, some authors focus on the benefits of integrating particular usability practices into ASD (J. C. Lee & McCrickard, 2007; Sohaib & Khan, 2010) or vice versa, while other authors propose integrated *user-centered agile software development* (UCASD) approaches (Barksdale & McCrickard, 2012; T. S. da Silva, Martin, et al., 2011).

Several literature reviews on UCASD research have been conducted. Although this study identified a plethora of existing works in these reviews, they suffer from several shortcomings: First, existing reviews lack a comprehensive coverage of the different dimensions of UCASD. They mainly focus on the actual software development practices to be used, and rarely describe further dimensions, such as the overall process to be followed, the people and social aspects involved or the technology that may be leveraged. Second, findings have been insufficiently abstracted and systematized, impeding a generalization of the results and applicability in specific context. For example, while it might be helpful to know the most elaborated practice to conduct end-user evaluations in a usability lab, this specific practice might be inappropriate when an according infrastructure is not available. In these cases, a more generic principle, i.e. a rule based on the examination of underlying concepts (Conboy, 2009), might be a more suitable starting point. Finally, from a methodical point of view, the conducted literature reviews on UCASD lack clear quality criteria for paper selection and need to be complemented by current findings. In order to address these research gaps, this paper aims at capturing and analyzing the current state-of-the-art in UCASD. More specifically, this study investigates the following research question: *Which principles constitute a user-centered agile software development approach?* Thus, following an approach similar to the *agile manifesto* (Beck et al., 2001), deriving a set of grounded principles for UCASD from the literature.

The remainder of the paper is organized as follows: In subsection 2 establishes the foundation by summarizing related work and outlining the existing research gap. Subsection 3 introduces the research method applied in this paper, describing the sourcing and search strategy, the paper selection process, and the final analysis. In the following sections, the results of the systematic review are presented and discussed: First, subsection 4 presents an overview of the results. The identified principles of UCASD are subsequently discussed in subsection 5. Finally, subsection 6 provides a summary of the paper and outlines the limitations and contributions of this work.

3.1.2. Foundations

ASD and UCD evolved from different motivations. Software engineers aim to satisfy customers through timely releases and responsiveness to change requests without compromising software quality. These goals are difficult to achieve through plan-driven SE approaches, resulting in proposals for ASD (Clegg et al., 1997; Petersen & Wohlin, 2010). UCD aims at ensuring appropriate usability of the implemented software, a characteristic that has not been considered sufficiently in traditional, plan-driven approaches or in agile approaches. UCD addresses this issue but does not consider agile principles. Therefore, first attempts to integrate ASD and UCD approaches were made about a decade ago. In the following, existing literature reviews on UCASD are presented and analyzed. Thereafter, the study introduces a comprehensive framework of UCASD dimensions.

Summary of Existing Literature Reviews

Sohaib and Khan (2010), da Silva et al. (2011), and Barksdale and McCrickard (2012) provide comprehensive overviews of approaches to UCASD. The reviews, which are briefly summarized in the following, each have a unique focus and pursue different objectives.

Sohaib and Khan's (2010) review aims to identify tension fields between UCD and ASD. They find that the most important questions are whether software development should focus on the customer or on the end-user of the software, whether developers' aim should be to create useful or usable software, and whether unit testing or usability testing is more important. They provide references that highlight different aspects of these questions. Furthermore, they address discrepancies between the use of little and extensive upfront design. Based on their findings, Sohaib and Khan (2010) suggest an integrated approach, bridging the gap between the two fields. The approach focuses on the recommendation to use specific practices. In their review, Sohaib and Khan (2010) do not attempt to give an exhaustive overview of extant publications on UCASD. Instead, they use references to illustrate each of the identified tension fields. Consequently, they describe neither the search process nor the criteria for inclusion or exclusion of primary sources.

In contrast, da Silva et al. (2011) explicitly address their work as a systematic literature review and follow the research method described by Kitchenham and Charters (2007) to cover existing literature on the integration of UCD and ASD as completely as possible. Following Kitchenham and Charters (2007) recommendation, they explicitly define their inclusion criteria for publications. Moreover, having conducted an automated search in four databases and a manual search in three conference proceedings, their search scope is extensive. The systematic approach resulted in the analysis of 58 articles, whereby they provide extensive descriptive statistics on various aspects of the retrieved publications. Furthermore, da Silva et al. (2011) identify key aspects of UCASD, again focusing only on relevant practices. They argue that the most significant principles for integrating UCD and ASD are "little design up front" (LDUF) – i.e. the design of a small proportion of the overall system prior to the start of system development – a close collaboration between usability and development experts, and a deferred development with the designers working one sprint ahead of the developers. Based on their findings, da Silva et al. (2011) propose a high-level integrated framework with parallel design and development activities. Additionally, the most commonly recommended practices when integrating UCD into ASD are identified and mapped to their proposed framework.

The main purpose of Barksdale and McCrickard (2012) literature review is to point out an insufficiently covered area of research, highlighting the need for their own research project while placing it in a broader context. Thus, the main contribution of their review lies in the organization of extant research efforts concerning the integration of UCD and ASD along five dimensions. Barksdale and McCrickard (2012) thus distinguish between a focus on practices, the process itself, technological means to support software development, and the people, or social aspects, involved in the process. They adopt a broad view of practices, referring to "the whole art of performance" (Bourdieu, 1977, p. 20), covering all actions of performing or actually doing something within the software development process. Following this generic classification approach, all included publications are tagged with a clear focus to one of these research areas. Furthermore, the study provides illustrative examples of each integration strategy, i.e. depicting the relationships of important concepts from both fields. Barksdale and McCrickard (2012) indicate the criteria they used to select studies, but solely drew on Google Scholar to conduct an automated search.

Gap Analysis of Existing Literature Reviews

An analysis of the related work suggests three major shortcomings: First, none of the existing reviews broadly covers UCASD across its different dimensions, such as practices, process, technology, people, and social aspects. While the works of Sohaib and Khan (2010) and da Silva et al. (2011) focus strongly on practices, Barksdale and McCrickard (2012) are most interested in the people/social dimension. Therefore, none of the reviews that cover all relevant dimensions of UCASD in an integrated manner.

Second, the existing reviews do not sufficiently abstract and generalize the results. While recommendations for specific practices are given, these practices are not sufficiently abstracted. As the applicability of specific methods and practices depends on the given context, it is difficult to implement the proposed UCASD approaches in a real-life software development project. A further analysis of existing works, aiming at a clear separation of generic principles and their implementation through specific instantiations would increase the applicability of the findings.

Third, while each review provides summarized information on the included primary articles, none of them includes a quality assessment of the encompassed primary studies, as Kitchenham and Charters (2007) recommends. In an attempt to update the findings of prior reviews, a quality analysis of the used sources should be included, and the results of this analysis should be taken into account when interpreting the results. Finally, the systematic literature review by da Silva et al. (2011), which can be considered the most extensive and rigorous review to date, was conducted in 2010. Given the dynamics of the analyzed field of research, major contributions might have been published since then, which was also confirmed in a preliminary assessment of recent publications in the field of interest.

Summarizing, the motivation for conducting a systematic literature review tailored to the scope of the envisioned UCASD approach is sufficiently substantiated. Still, the findings of the three assessed reviews have to be included in this work, and a crosscheck of the retrieved literature with the literature contained in the three published reviews should be performed.

3.1.3. Research Method

The literature review was carried out by following the established guidelines for conducting systematic literature reviews suggested by Kitchenham and Charters (2007), which are a proven means to arrive at a complete and thorough census of existing research within a domain (Brereton, Kitchenham, Budgen, Turner, & Khalil, 2007; Dybå & Dingsøyr, 2008; Kitchenham et al., 2009). Before conducting the actual review, it was planned in detail by establishing a review protocol. The review protocol guided each of the following steps in detail in order to ensure rigor and transparency in the research process (Carlsson, Henningsson, Hrastinski, & Keller, 2011; vom Brocke et al., 2009). It was based on the objective of finding guidance and recommendations for UCASD, which reflects the research question introduced earlier (see subsection 3.1.1), and included the following elements:

- A *search strategy*, i.e. the approach to identify appropriate search terms for querying scientific databases as well as the resources to be searched.

- *Selection criteria and procedures*, which are suitable to identify relevant primary studies in the full list of results emerging from the automated database search. In the selection process, the sample of included studies is narrowed down in multiple stages based on previously defined inclusion and exclusion criteria.

- The attributes used for a *quality analysis* of the included primary studies, which allows an assessment of the existing knowledge base.

- A *data extraction and synthesis* method, representing the systematic approach to consolidate and integrate existing knowledge.

The review protocol was developed in cooperation with the first and second authors, and validated by the third author prior to conducting the review. The following describes the implementation of the review protocol.

Data Sources and Search Strategy

Contributions to the research topic at hand may be found in different domains (i.e. information systems, and computer science), and in different sub-domains within these domains (e.g. HCI). For each of these domains, the most relevant databases were selected for the search. The IS literature has focused on three of these databases, namely ProQuest, Elsevier ScienceDirect, and EBSCO Host. Additionally, the three databases, IEEE Xplore, ACM Digital Library, and Springer Link, which mainly focus computer science topics, have been queried. The search string was composed to cover both fields of interest, i.e. UCD and ASD. Key words relevant to the first field were derived from an exploratory literature review, while key words for the second field were adopted from Dybå and Dingsøyr (2008), who performed a systematic literature review of empirical studies on ASD. Additionally, the terms "software development" and "systems development" were added to reflect the activity of interest and restrict the search space. Table 13 lists the key words and their corresponding logical operators.

Table 13 – Composition of the Search String.

AND	OR	Ergonomics, Human-Computer Interaction, Computer-Human Interaction, Interaction Design, Usability, User Experience, User-Centered Design, UI Design, Interface Design
	OR	Agile, Scrum, eXtreme Programming, Lean, Crystal Clear, Feature Driven Development, Dynamic Software Development
	OR	Software Development, Systems Development

Paper Selection and Pre-Assessment

The selection process involved four stages. In each stage, the sample size was reduced based on the inclusion criteria applicable in this stage as depicted in Figure 5. In the first stage, relevant publications were retrieved by querying the databases mentioned above with the search string depicted in Table 13 (see Appendix C for details per database). All database queries were made in the first two weeks of October 2012. This yielded a total of 1152 initial results, and 1034 publications after the removal of duplicates, which were included in the second stage.

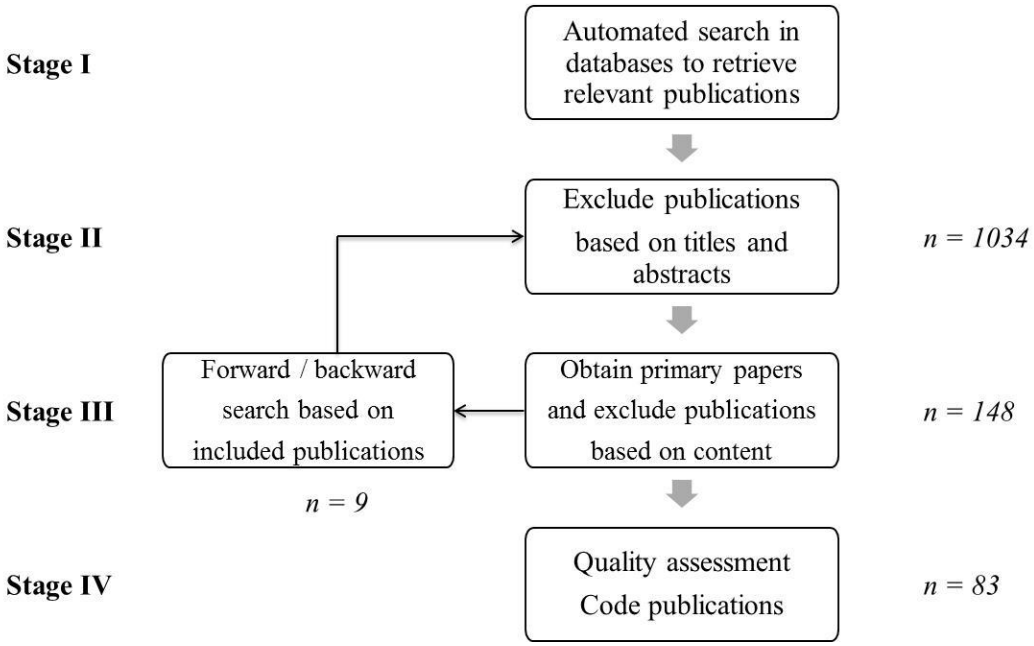

Figure 5 – Selection Process (Based on Dybå and Dingsøyr (2008)).

In the second stage, publications were excluded based on their titles and abstract. Criteria for inclusion in the following stage were that the article focuses on the integration of UCD and ASD. Special formats, such as editorials, prefaces, article summaries, interviews, news, correspondence, discussions, and readers' letters, were excluded. Moreover, articles summarizing tutorials, workshops, panels, or poster sessions were excluded. As Brereton et al. (2007) note, titles and abstracts of IS, CS, and SE publications are often of low quality. Thus, articles were included in the next stage only if the evaluation of titles and abstract did not lead to a conclusive decision as Dybå and Dingsøyr (2008) demonstrate. After the analysis of titles and abstracts, 148 publications were included in the following stage.

The third stage involved a more detailed analysis of the articles' content. Thus, the full text of each publication in this stage was retrieved for local storage. Each article in the sample's introduction, conclusion, and, in case of doubt, content was skimmed to assess its relevance. In this stage, stricter criteria were applied to determine the relevance of articles, as studies or practitioner reports should give recommendations or

requirements for software development integrating UCD and ASD. These recommendations or requirements may be given explicitly, for example: "UCD practitioners must be given ample time in order to discover the basic needs of their users before any code gets released into the shared coding environment" (Chamberlain, Sharp, & Maiden, 2006, p. 151). On the other hand, they might emerge as a result of the study, for example, if a positive impact of applying a certain principle or practice has been observed. Based on these criteria, 74 articles were considered relevant for inclusion in the next stage. Following the strategy suggested by Webster and Watson (2002), the list of key publications served as a basis for a forward and backward search. The retrieved new and potentially relevant articles were fed to the second stage for further processing, resulting in an iterative extension of the sample. This resulted in an additional set of nine articles for inclusion in stage four. It has to be noted that both da Silva et al.'s (2011) and Sohaib and Khan's (2010) reviews were included in the stage three sample. However, they were excluded from the detailed analysis in stage four as they do not present the results of primary research. In total, 83 papers were selected for the final sample[2].

In the fourth stage, the final sample passed a first categorization and a quality assessment. During the categorization, each paper was assigned to one of the four research foci, which will be introduced in the next subsection. A quality assessment of each paper was subsequently conducted to obtain additional information supporting the interpretation of the paper's recommendations. This assessment was based on four questions, which are given in Table 14. Each of the four criteria was graded on a dichotomous scale as "yes" or "no", while question 3b was answered by means of the respective method, which was either explicitly stated in the paper or determined by the researcher based on the available information.

Table 14 – Criteria for Quality Assessment.

1.	Is there a clear statement of the research goals (e.g. in an explicitly verbalized research question)?
2.	Is there an adequate description of the context in which the research was carried out?
3.	(Only applicable to empirical research papers):
	a. Is the research method explicitly stated?
	b. Which research method was chosen?
4.	Are there explicit recommendations, which could be aggregated as principles?

Paper Analysis - Identification of Dimensions

To analyze prior research, a comprehensive, hierarchical coding system was established. UCASD is essentially about the integration of UCD and ASD. Therefore, to identify the basic dimensions of the coding system, it is helpful to investigate on which levels integration may be accomplished. Barksdale and

[2] The list of papers and their corresponding classifications can be accessed at https://madata.bib.uni-mannheim.de/80/

McCrickard (2012) present an approach, in which the existing literature is classified into five types of integration for UCASD: Process integration, practices integration, people integration, social integration, and technology integration. *Process integration* is understood as the merging and synchronizing of independent UCD and ASD processes, providing a unified process incorporating both perspectives. *Practices integration* represents the incorporation of UCD practices into ASD and vice versa. *People integration* is understood as changes in the team composition to bring experts of the two different disciplines together (e.g. adding a designer to a team of developers). *Social integration* reflects social interaction and the joint creation of knowledge. Finally, *technology integration* entails the use of technological means to support and coordinate activities.

The different levels of integration presented by Barksdale and McCrickard (2012) allow for a comprehensive and differentiated classification of existing works on UCASD. Therefore, the literature analysis used these integration levels as a starting point to determine the basic dimensions of the coding system. In contrast to Barksdale and McCrickard (2012), this study does not differentiate between people integration and social integration, as these two aspects are almost inseparable.

Paper Analysis - Identification of Codes

The qualitative data analysis software *MAXQDA*[3] was used to code the papers. As a universal tool for qualitative data analysis, *MAXQDA* is usually employed to analyze textual data, such as interview transcripts, and supports a variety of qualitative research methods. Various authors (e.g. Dybå & Dingsøyr, 2008; Wang, Zheng, Xu, Li, & Meng, 2008) have reported positive experiences concerning the use of similar software (e.g. NVivo[4]) to synthesize data from a corpus of texts emerging from a systematic literature review. These experiences motivate the adopted approach to process a large amount of textual data rigorously and transparently. This seems especially useful as evidence found in extant literature, which may be used to derive principles, can be retrieved easily in later stages of the research process. Using *MAXQDA*, codes can be assigned to text segments or images in documents. Besides descriptive statistics on the occurrence of codes in documents, *MAXQDA* provides a convenient way to extract coded text passages, allowing for easy data synthesis. Moreover, MAXQDA helped to conduct a quality analysis of the included papers. To be able to retrieve papers within a certain category in a convenient manner, the categorization was done by assigning a code reflecting the appropriate category to the title and abstract of each paper. Furthermore, if a criterion given in Table 14 was assessed to be fulfilled, the text segment providing the information was coded accordingly. The analysis of the document corpus was aimed at identifying aspects

[3] http://www.maxqda.de/

[4] http://www.qsrinternational.com/products_nvivo.aspx

that have to be taken into account when integrating UCD and ASD, with the overarching aim of deriving principles. During the analysis of the document corpus, observations made concerning different aspects of integration were assigned a meaningful code, with the aim of consolidating the perspective different authors take on the same issue later on.

The dimensions introduced above formed the basis of a first set of high-level codes. For simplification reasons, the terms “process”, “practices”, “people/social”, and “technology” were used as high-level codes in the following. Based on the analysis and comparison of the existing studies, as presented in subsection 3.1.2, initial sub-codes were assigned to each high-level code (see Table 15). Each of the papers in the final sample was carefully read, analyzed, and coded according to this system. While the initial set of codes proved to be exhaustive, additional lower level codes and hierarchy levels had been added during the analysis of the literature to document the insights reported in these studies. This process becomes omnipresent when comparing the two coding systems. The initial coding system reflects 21 codes with two hierarchy levels, whereas the final coding system contains 73 codes in four hierarchy levels. Various reasons, such as the addition of specific practices as codes, drove the extension of the coding system. During the coding process, text passages were related to one of the codes. While the group assignment according to the main research focus was mutually exclusive, this does not mean that only aspects concerning this stream of research were discussed in the paper. For example, when the authors of a paper with a clear process focus recommend the use of personas, the corresponding text passage was coded as “personas”, which was applied to every document in the corpus discussing personas independently of its research focus. Moreover, throughout the coding process, potential new codes were challenged against existing codes and vice versa. Initially, this led to many changes in the coding system. However, the codes became more stable as they developed organically.

Table 15 – Initial Coding System.

Level 1	**Process**	**Practices**	**People/Social**	**Technology**
Level 2	Little Design Up Front	Prototyping	Close Collaboration	Data Exchange
	User Testing*	User Testing*	Cross-Functionality	IDE Integration
	One Sprint Ahead	User Stories	Knowledge Transfer	
	Cohesive Overall Design	Scenarios		
	Parallel Tracks	Personas		
	Iterative Design / Development			
	Incremental Design / Development			

Identification of Candidate Principles

In order to identify candidate principles, each code was investigated concerning two characteristics. First, the *frequency of occurrence* was assessed to gain an impression of the relative importance of each aspect. As an example, the code "prototyping" was assigned in 40 (48.2%) articles in the literature review, indicating that the use of prototypes was discussed in this document. In contrast, the idea of "remote usability testing" was only found in one (1.2%) article, leading to the conclusion that, while the former aspect has a high significance for the scientific community, the latter is a marginal topic. For this assessment, the insights reported in existing literature reviews were also considered (cf. subsection 3.1.2).

Second, *guidance and recommendations* for the conceptualization of an integrated UCASD approach had been collected. While 17 (20.5%) of the articles included in the review contain explicit advice on how to integrate UCD and ASD, implicit recommendations can be derived too if applying a certain practice has been observed as having a positive impact. Thus, the content of the coded text passages was analyzed for either explicit or implicit guidance concerning the integration of UCD and ASD. Finally, to obtain an overview of the recommendations in the extant literature, the coded text passages were reviewed once again and organized along emerging patterns. These patterns represent related ideas or concepts presented in different publications along the four dimensions, leading to the merger of independent aspects in the coding system. The identification of such patterns was based on a qualitative analysis and the expertise of the researchers in this domain. In this step, also the findings of the three assessed reviews were drawn upon to identify themes for potential principles (Carlsson et al., 2011).

3.1.4. Results

Number and Distribution of Publications

When analyzing the result set, it became apparent that, while the integration of UCD and ASD was first discussed a decade ago, the topic gained momentum in 2007 and, since then, a constantly high number of relevant articles have been published every year, as Figure 6 illustrates. This reflects that, while the idea of integrating UCD and ASD has been around for some time, many integration issues are still unresolved and research is ongoing. In particular, at least 17 relevant articles have been published since da Silva et al.'s (2011) systematic literature was conducted in late 2010, justifying an update of their findings.

Moreover, the research type of each paper is presented in Figure 7. Each paper was classified according to one of four research types, which are non-exclusive of each other. The types identified are qualitative research (N=30), (i.e. action research (N=2), case study (N=17), ethnography (N=4), grounded theory (N=4) and interview (N=3)), quantitative research (N=4), theoretical work (N=23), and none (N=27) of the aforementioned (e.g. practice reports). Industrial experience reports refer to work that present practical

experience. This is supported by existing evidence that industrial experience strongly influences agile research (T. S. da Silva, Martin, et al., 2011; Kuusinen, Mikkonen, & Pakarinen, 2012).

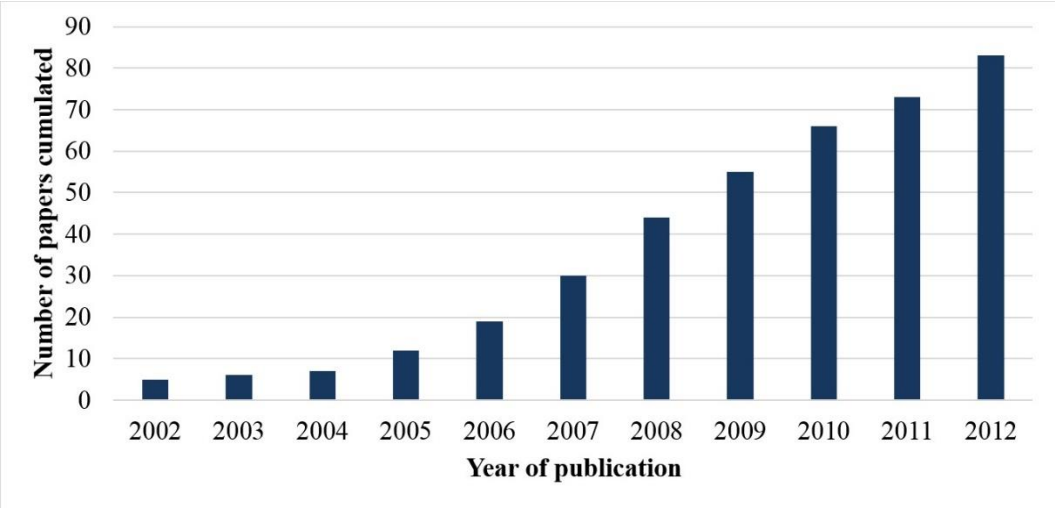

Figure 6 – Number of Publications by Year.

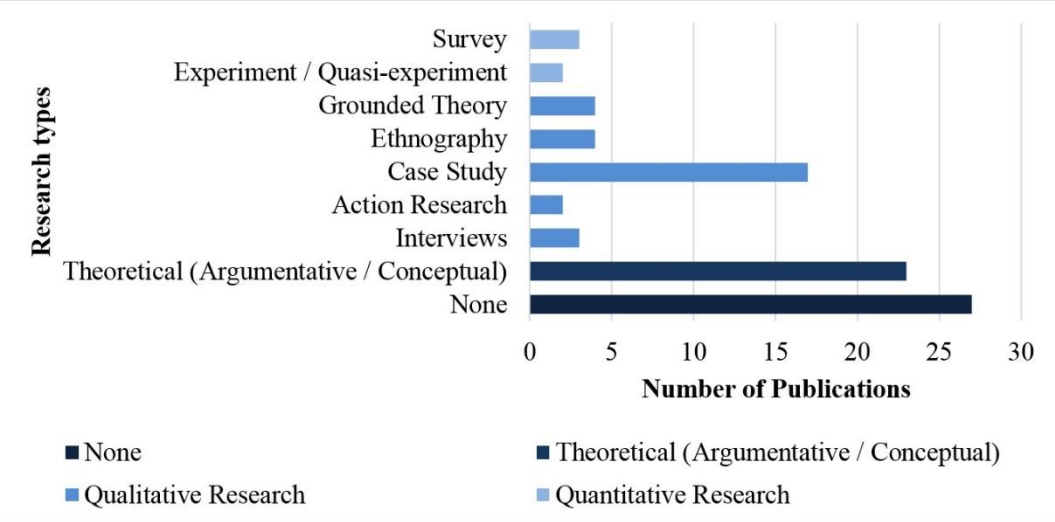

Figure 7 – Number of Publications by Research Type.

Table 16 presents the results of categorizing the 83 publications included in the final sample according to their main research focus based on the dimension of the coding system. The analysis revealed that both process and practice integration aspects are strongly represented in the final sample. While the former are discussed in 24 (28.9%) publications, the latter form the biggest group with 38 (45.8%) of the included

articles. However, the high number of publications discussing process integration aspects does not mean that a variety of mature integrated approaches exists. Among the 24 publications that focus on process integration, only 15 present novel approaches, while the remaining articles discuss only approaches that were published earlier by the same authors. In general, it is difficult to distinguishing between a focus on process or practice integration aspects, as authors frequently describe their work as integrated approaches, but only discuss the integration of UCD into ASD while neglecting the developer's perspective, which did not meet the definition of the process focus taken here. People and social integration aspects were discussed in 14 (16.9%) papers, while technological integration received the least attention with seven (8.4%) of the included publications looking into it.

Table 16 – Overview of Publications in the Final Sample.

Dimension	**Included Publications**	**Number of Publications**
Process	Benigni, Gervasi, Passeri, & Kim, 2010; Budwig, Jeong, & Kelkar, 2009; Felker, Slamova, & Davis, 2012; Ferreira et al., 2007a; Ferreira, Noble, & Biddle, 2007b; Fox et al., 2008; Andreas Holzinger, Errath, Searle, Thurnher, & Slany, 2005; Hussain, Lechner, Milchrahm, Shahzad, Slany, Umgeher, & Wolkerstorfer, 2008; Hussain, Lechner, Milchrahm, Shahzad, Slany, Umgeher, Vlk, et al., 2008; Kuusinen et al., 2012; J. C. Lee, Judge, & McCrickard, 2011; J. C. Lee et al., 2009; J. C. Lee & McCrickard, 2007; Losada, Urretavizcaya, & de Castro, 2011; Losada et al., 2013; Losada, Urretavizcaya, López-Gil, & Fernández-Castro, 2012; Memmel, Gundelsweiler, & Reiterer, 2007a, 2007b; Miller, 2005; Najafi & Toyoshiba, 2008; Paelke & Nebe, 2008; Paelke & Sester, 2010; Wolkerstorfer et al., 2008; P. Zhang, Carey, Te'eni, & Tremaine, 2005	24 (28.9%)
Practices	Adikari, McDonald, & Campbell, 2009; Beyer, Holtzblatt, & Baker, 2004; Broschinsky & Baker, 2008; Carvalho, 2010; Chamberlain et al., 2006; Cho, 2009; Constantine, 2002; Constantine & Lockwood, 2002; B. S. da Silva, Aureliano, & Barbosa, 2006; Detweiler, 2007; Düchting et al., 2007; Evnin & Pries, 2008; Ferre, Juristo, & Moreno, 2005; Fisher & Bankston, 2009; Haikara, 2007; Hansson, Dittrich, & Randall, 2006; Hellmann, Hosseini-Khayat, & Maurer, 2010a, 2010b; Hennigs, 2012; Hodgetts, 2005; Humayoun et al., 2011; Hussain, Milchrahm, et al., 2009; Illmensee & Muff, 2009; Isomursu, Sirotkin, Voltti, & Halonen, 2012; Kane, 2003; M. K. Larusdottir, 2011; M. K. Larusdottir, Bjarnadottir, & Gulliksen, 2010; S.-H. Lee, Ko, Kang, & Lee, 2010; Medina, Burella, Rossi, Grigera, & Luna, 2010; Memmel, Reiterer, & Holzinger, 2007; Meszaros & Aston, 2006; Obendorf & Finck, 2008; Obendorf, Schmolitzky, & Finck, 2006; Patton, 2002a, 2002b; Petrovic & Siegmann, 2011; Rafla, Robillard, & Desmarais, 2007; Rittenbruch, McEwan, Ward, Mansfield, & Bartenstein, 2002	38 (45.8%)
People & Social	Barksdale & McCrickard, 2010, 2012; Barksdale, Ragan, & McCrickard, 2009; J. Brown, Lindgaard, & Biddle, 2008; J. M. Brown, Lindgaard, & Biddle, 2011; Ferreira, Sharp, & Robinson, 2010, 2012; Kollmann, Sharp, & Blandford, 2009; Leszek & Courage, 2008; Lievesley & Yee, 2006; M. Singh, 2008; Ungar, 2008; Ungar & White, 2008; Williams & Ferguson, 2007	14 (16.9%)
Technology	Feiner & Andrews, 2012; Gonçalves & Santos, 2011; Hosseini-Khayat, Hellmann, & Maurer, 2010; Humayoun, Dubinsky, Catarci, Nazarov, & Israel, 2012; J. C. Lee, 2006; Peixoto, 2009; Peixoto & da Silva, 2009	7 (8.4%)
Total		83 (100.0%)

Synthesized View

A key objective of the literature review was the synthesis of evidence found in extant literature in order to derive principles by applying a coding system to each article's content. Figure 8 depicts the final version of the coding system, which evolved for the "process", "people/social", "technology", and "practices" dimensions. Some codes were omitted to improve readability (see Table 38 for additional Codes in Appendix A).

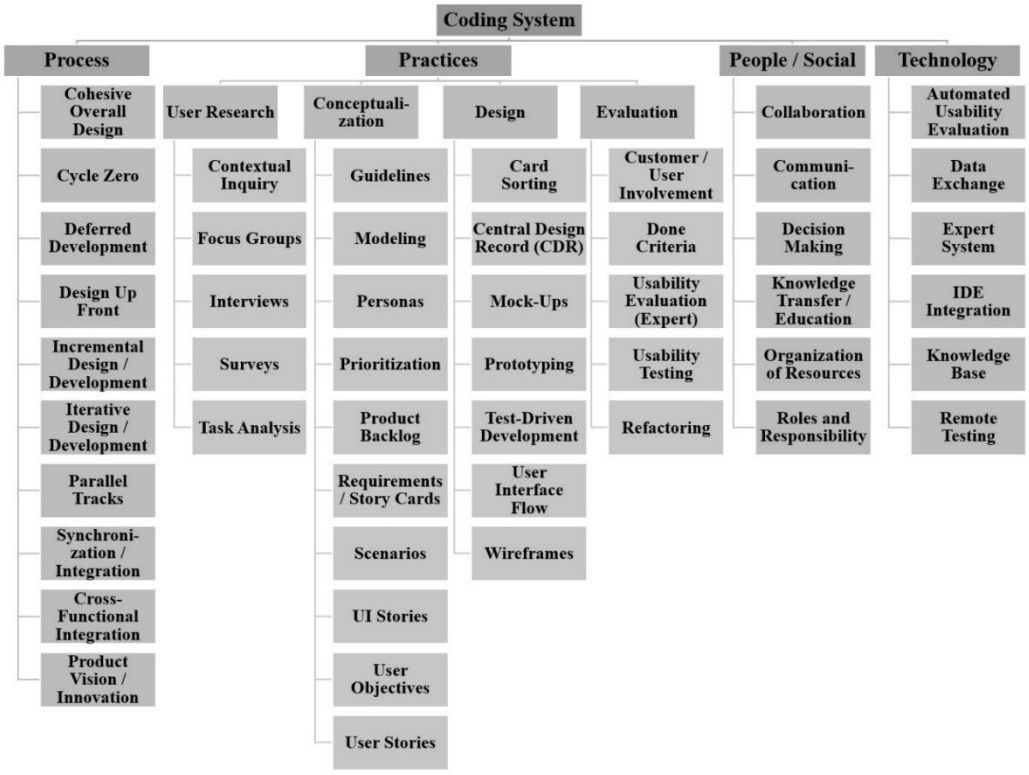

Figure 8 – Final Coding System for the Process, People/Social, Technology, and Practices Dimensions.

The coding system shows that, while some codes explicitly emerged from the integration of ASD and UCD approaches (e.g. "synchronization" and "collaboration"), other codes represent aspects of the individual, original approaches (e.g. "iterative design" or "incremental design"). This result illustrates that integrated approaches strive to keep the best aspects of the original approaches on the one hand, while trying to enable a smooth merge of the two worlds on the other hand. Interestingly for the process dimension, many more codes have been identified for the other two dimensions. Moreover, the codes in the process dimension are mainly specific to software development, while the majority of the codes in the other dimensions are rather generic (e.g. codes like "collaboration" or "data exchange"). This might be due to the overall number of papers assigned to the "process" dimension being larger than in the other dimensions. Apart from that, it might also be an indicator of the depth and maturity of discourse in each of the dimensions. While aspects

of process integration have been intensively discussed and refined, the discussion in the people/social and technology dimension seems to be rather limited in UCASD research.

Owing to the large variety of practices discussed in the review, sub-categories for the practices dimension were created based on the process phases. While strict process phases are uncommon in ASD, UCD proceeds along pre-defined (but not necessarily sequential) phases. The classification follows the four phases defined in the ISO standard for human-centered design: research, specification, design, and evaluation (ISO, 2010). The four aforementioned phases are considered generic enough for identifying sub-codes within the practices dimension, One example is the inclusion of user research, which is absent in other development methods. Abstracting these phases from the general activities to be done (investigate the context, specify requirements, design the software, and, finally, test its congruence with the requirements), this classification also allows for mapping typical agile practices (such as user stories). Figure 8 displays all codes for the practices dimension and their assignment to sub-dimensions.

The results confirm that, especially in UCD, there is a plethora of different methods as others (cf. Mao et al., 2005) have shown. Recommendations for individual practices differ across the studies, resulting in a heterogeneous picture of practices with an overlapping, if not identical scope. For example, in the specification phase, scenarios and UI stories, as well as mock-ups and wireframes could be used alternatively. As suggested earlier, it is therefore helpful to abstract phases from specific practices.

The final coding system contained in Figure 8 provides a comprehensive overview of aspects relevant to UCASD. Consequently, these codes, respectively the underlying text passages, were the key takeaway from the literature review, whose overarching aim is to derive principles to guide the integrated UCASD approach. Throughout the data gathering process, a lack of content for the technology dimensions was identified. While aspects concerning technological means to support user-centered agile processes (N=10) were found, the text passages were of a general nature and did not provide guidance on technology support. Therefore, no principles concerning technology integration were derived. A different challenge emerged for the people/social dimension. Although more sources were found, the recommendations were contradictory. While ASD suggests the collective accountability of the team (Kettunen, 2009), UCD methods usually suggest individual responsibility, for example, allocating the responsibility of developing a usable product to the UCD expert (A. Cooper, 1999; Gould & Lewis, 1985; Preece, Rogers, & Sharp, 2002). Furthermore, the team setup led to contradictory evidence. On the one hand, two *dedicated teams,* which handle design and development activities respectively, can be formed. On the other hand, a *cross-functional team,* handling both tasks, might be established (Ferreira et al., 2010, 2012). Therefore, no principles for the people/social dimension were derived. The paper's conclusion discusses how future work might derive corresponding principles for the technology and people/social dimensions.

3.1.5. Principles of UCASD

The following discusses the identified principles for the integration of UCD and ASD organized along the two central dimensions processes and practices.

Process Principles

First, a process perspective is necessary to decide how activities involved in the software development process are organized on a fundamental level. Various aspects focusing on the process perspective were frequently mentioned in the identified publications and appeared to have a high level of maturity as a consensus was identified for most of them. In Figure 9, the codes assigned to the process perspective are listed along with their occurrence in the identified publications. The items have been consolidated into three key principles forming a UCASD approach.

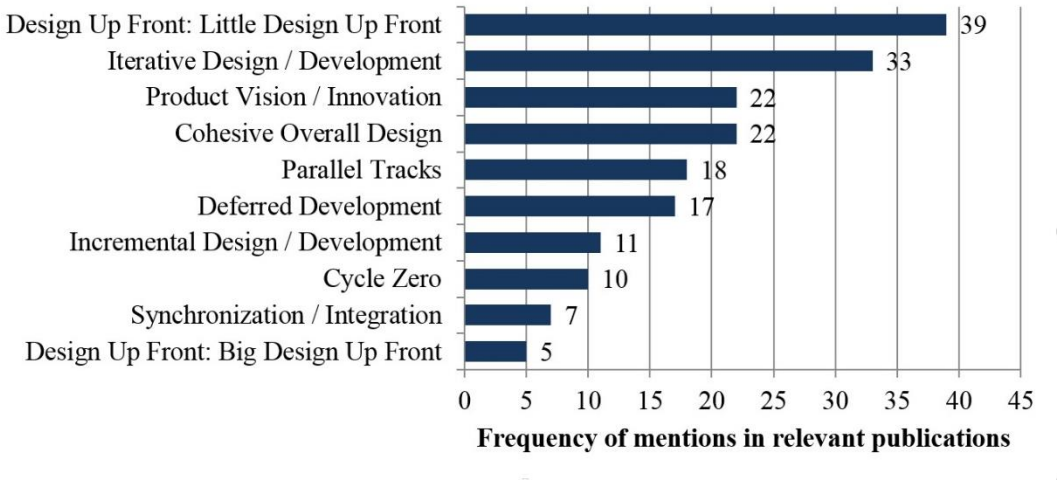

Figure 9 – Codes and Articles Related to the Process Dimension.

Separate Product Discovery and Product Creation

Focus Area. The valuation of upfront analysis and design was recognized as one of the main tension points between UCD and ASD early on. While the former promotes the extensive upfront analysis of user requirements and the design of the users' interaction with the system, the latter focuses on delivering a working code at the expense of an exhaustive planning and design phase. Moreover, there is no common definition of what constitutes sufficient preparation, and the effort put into initial analysis and design activities ranges from days (e.g. Constantine, 2002) to weeks (e.g. Budwig et al., 2009).

Evidence. The results of the literature review confirm the importance of an upfront analysis and design: 42 (50.6%) of the included publications discuss the necessity of upfront design efforts. Seven of these present empirical evidence. A total of 39 articles argue that, while an extensive upfront design is against

agile principles, *little design up front* (LDUF) is needed for successful UCD efforts in an agile environment. As an example, Miller (2005) and D. Sy (2007) suggest that interaction designers use an upfront cycle for planning and collecting customer input. Fox et al. (2008) are more specific with regard to methods and reserve an upfront phase for contextual inquiry and iterative low-fidelity prototyping, resulting in a preliminary design and a list of requirements that are handed over for development. This is in line with the results of T. S. da Silva et al. (2011), who find that 31 of the 58 papers they reviewed recommend a strictly limited upfront design effort.

In particular, 10 (12.0%) of the included publications implement the LDUF concept by reserving a *cycle zero*, or sprint zero, for analysis and design activities before actual agile development iterations start. The idea of time-boxing initial analysis and design for the duration of a development cycle was first suggested in Miller's (2005) integrated process model and has since been adopted and modified by various authors, such as D. Sy (2007), Fox et al. (2008), and T. S. da Silva et al. (2011).

There is evidence that a limited upfront design effort is particularly necessary to provide a consistent and cohesive user interface and navigation structure as it supports designers in finding a suitable design concept from the very beginning (Adikari et al., 2009; Ferreira et al., 2007b). Patton (2002a) states that user needs and goals have to be clearly defined before conducting design and development efforts in order to ensure the usability and usefulness of the software in ASD. Various authors have confirmed this view, presenting user research as a key aspect of upfront design efforts in agile environments (Chamberlain et al., 2006; T. S. da Silva, Martin, et al., 2011; Fox et al., 2008; J. C. Lee et al., 2011; Miller, 2005; D. Sy, 2007). In total, 22 (26.5%) of the papers in the review confirm a relationship between LDUF and a *cohesive overall design* of the final product, which is challenging to achieve in an agile environment. Meszaros and Aston (2006) assert that evolutionary design is not suitable for user interfaces when insufficient conceptual guidance is given. Moreover, Constantine (2002) argues that late revisions of the user interface are unacceptable as such revisions would be highly disruptive for users already familiar with the application.

While there is no broad agreement in the literature on the extent and outcome of LDUF activities, it is evident that what is commonly called upfront "design" also has the character of a constituting analysis involving extensive stakeholder interactions. Other than the actual UI design itself, user research does not yet involve any commitment, thus maintaining flexibility. Consequently, upfront design activities should first and foremost be understood as a preliminary exploration phase, harnessing the potential of UCD to generate new ideas. Twenty-two (26.5%) of the publications included in the review address this issue, and see UCD as a means to establish and communicate a *product vision* in order to "improve the line of sight between strategy and team-level execution" (Fisher & Bankston, 2009, p. 225). While UCD provides suitable concepts for approaching ill-defined problems and delivering a product with a high degree of

innovation, exploratory activities are insufficiently addressed in ASD (Ferre & Medinilla, 2007). Kettunen (2009) points out that, while agile methods are useful means to iteratively select and refine product features, large-scale product innovations are beyond their scope. He answers the question of how to build a product correctly, i.e. delivering a valuable product to the market as soon as possible, while maintaining flexibility along the way. However, product scoping or ideation in general, which would address the question of building the right product, is not part of agile methods. This shortfall can be countervailed by drawing on UCD, in which collecting stakeholder needs, expectations, and ideas are core functions. To conclude, the usefulness focus of SE has to be complemented by usability concerns not only during product development, but also during product planning. In order to deliver a consistent and cohesive design throughout software product development, such concerns should be included in the product vision (Budwig et al., 2009; Ferreira et al., 2007b; Fisher & Bankston, 2009; Kollmann et al., 2009; Kuusinen et al., 2012; Lievesley & Yee, 2006; Meszaros & Aston, 2006; M. Singh, 2008).

Suggestion. A shift from an upfront design to upfront analysis concept is identified. In order to deliver a product that is valuable to the customer and end-user in that it is both usable and useful, product discovery and product creation should be separated in UCASD. Overall, as listed in Table 17, four codes from the coding system support the suggestion to include an additional principle in this specific context.

Table 17 – Codes Related to the Separate Product Discovery and Product Creation Principle.

Included Codes	*Resulting Principle*
Little Design Up Front Cycle Zero Cohesive Overall Design Product Vision / Innovation	Separate Product Discovery and Product Creation

Besides making room for sufficient upfront design activities as a prerequisite for delivering a usable product with a consistent user interaction concept, it allows for mitigating agile shortcomings with regard to product discovery, fostering the delivery of a useful product. Thus, first principle is:

> ***Principle 1 (Separate Product Discovery and Product Creation):*** *User-centered agile software development should be based on separated product discovery and product creation phases*

While the literature clearly evidences the importance of product discovery in general, there is currently a knowledge gap with regard to the extent of product discovery activities. This is also visible within the ten listed items identified in the publications, in which a pair of antipodes can be found, namely the contrast between little or extensive upfront design efforts. Thus, further empirical research is required to investigate the extent of product discovery activities in respect of the different contingency factors on the team, product,

and organizational levels. For example, it would be of interest to understand how different software product characteristics, such as the maturity, complexity, and degree of interaction, influence the extent and outcome of product discovery activities.

Iterative and Incremental Design and Development

Focus Area. Both ASD and UCD promote an iterative and incremental approach to software development. This allows for using feedback collected in previous iterations to enhance the emerging product in future iterations. Thus, intermediate solutions to the development problem are used as a pathway to finding a complete and desirable solution for the customer or end-user.

Evidence. The main commonalities of ASD and UCD have been identified in earlier literature reviews by T. S. da Silva et al. (2011), and Sohaib and Khan (2010). Specifically, their reviews confirm the significance of iterative and incremental design and development. There is general agreement among UCD researchers that an *iterative* approach is key for developing a product with high usability (Gould & Lewis, 1985; Gulliksen et al., 2003). According to Mao et al. (2005), design iterations are the most commonly used UCD paradigm; however, feedback is also a key aspect in ASD. In ASD, the overall project is typically composed of subsequent self-contained cycles, each of which comprises analysis, design, programming, and test activities (Larman, 2003). Among the included papers, 33 (39.8%) explicitly discuss design or development iterations. In accordance with UCD approaches, empirical feedback is used to revise designs in the next iteration. The iterative refinement and evaluation is continued until the user's needs are met, thus ensuring that an adequate level of usability is achieved (A. Cooper, Reimann, Cronin, & Noessel, 2014; Preece et al., 2002). In particular, all publications presented in the process dimension promote an iterative process setup.

An iterative approach allows for the product under development to be refined based on feedback collected at an earlier stage. In contrast, an *incremental* strategy means that system functionality is partitioned into thin vertical slices from the user interface to the database layer. Each slice is functionally coherent and demonstrable. Instead of delivering the required functionality all at once, the scope of the system grows progressively with the addition of consecutive slices (Cohn, 2004; R. C. Martin, 1999). Solutions to software development problems are often complex and exhibit many degrees of freedom. Thus, eliciting requirements from users often results in what Boehm (2000) calls the IKIWISI symptom: "I don't know how to tell you, but I'll know it when I see it" (p. 99). Especially for interaction features, users might not be able to specify their needs unless confronted with a tangible representation of the system, such as a working product increment that would allow them to clarify and refine their vision of the software (Boehm, 2000, 2011; Larman, 2003). Drawing on this insight, the aspect of incremental development was mentioned in 11 (13.3%) articles, typically with reference to functional tests or usability evaluations.

The associated papers identified in the literature provide no empirical evidence for the outcome of iterative and incremental design and development. This may be because the empirical studies not only consider the user-centered design or agile development perspective independently from each other, but as a combination of the two (Chamberlain et al., 2006; Ferreira et al., 2007b; Fox et al., 2008; Kollmann et al., 2009; Kuusinen et al., 2012; J. C. Lee et al., 2011; Lievesley & Yee, 2006).

Suggestion. In summary (see Table 18), developing software in an iterative manner creates short feedback cycles, while an incremental development in vertical slices allows for evaluating working product increments. Additionally, adapting the product scope after each iteration embraces change and is a prerequisite to deliver valuable software.

Table 18 – Codes Related to the Iterative and Incremental Design and Development Principle.

Included Codes	*Resulting Principle*
Iterative Design / Development Incremental Design / Development	Iterative and Incremental Design and Development

Beyond the frequent occurrence of the two aspects in the literature, a general acceptance of this strategy can be inferred from references to iterative and incremental concepts for ASD (e.g. Scrum) or UCD (e.g. ISO 9241-210). Even though the reviewed papers do not explicitly discuss an iterative and incremental approach, none of them challenges this paradigm. Thus, the second principle is articulated as follows:

> ***Principle 2 (Iterative and Incremental Design and Development):*** *User-centered agile approaches should support software design and development in short iterations and in an incremental manner.*

Parallel Interwoven Creation Tracks

Focus Area. Extending the separate product discovery and product creation principle, the third principle shapes the subsequent course of action after the start of regular design and development activities. As a restricted upfront phase of analysis and design allows for specifying the user interaction only for the most important features of a system, features with a lower priority have to be considered in later iterations in parallel to development. Thus, it is necessary to conduct user research and prepare designs for the upcoming development cycle at least one iteration (or sprint) ahead of the development team.

Evidence. In total, 18 (21.7%) of the publications included in the literature review support this principles, whereas five are empirically grounded. On the one hand, the literature explicitly combines the LDUF concept with the organization of continuous analysis and design activities. On the other hand, development

activities need to be organized in *parallel tracks*. The literature offers no alternative approach for the latter. The incremental nature of ASD enables UCD experts to prepare the interaction concept for envisioned system features consecutively according to their priority. Thus, an initial interaction concept for the most important system features can be prepared in an upstream iteration prior to development. Enhancing this concept in parallel to development allows for maintaining a suitable balance between necessary upfront design and flexibility, while designs are prepared for implementation just in time. Additionally, engaging in continuous analysis and design activities in parallel to development allows for incorporating changing user and customer needs, thus leading to an increased degree of software usability and usefulness.

The concept of designing *one sprint ahead* is commonly found in existing UCASD approaches, and was mentioned in 17 (20.5%) publications in the review. However, only two publications support this argument with empirical evidence (J. C. Lee et al., 2011; Najafi & Toyoshiba, 2008). As examples, Miller's (2005) and D. Sy's (2007) proposals schedule activities for the design team to gather customer data and develop user interface specifications for the next cycle, while the development team simultaneously implements specifications prepared in the previous cycle. Fox et al. (2008) moreover propose that the development team implements the design concept prepared in the preceding iteration, while the UCD staff conducts a contextual inquiry and prepares a design prototype for the next iteration.

Despite the need to work in parallel, seven (11.86%) of the publications in the review point out that a deferred development necessitates mechanisms for the *synchronization and integration* of design and development work. However, the scheduling of resources in practice is a non-trivial task and can be identified as one constraint. Researchers investigating the topic of *cross-functional integration* have discussed the question of how to integrate the development and design functions since the inception of UCASD more than a decade ago. The potential of cross-functional integration has also been intensively investigated from an empirical point of view in product development research (e.g. Botzenhardt & Meth, 2011; Troy et al., 2008). Two specific dimensions of cross-functional integration in the context of UCASD are identifiable in the literature. On the one hand, the necessity of functional diversity on the team-level, i.e. the degree to which team members differ with regard to their functional background and experiences (Gebert, Boerner, & Kearney, 2006), results in an organizational challenge to integrate design and development experts. On the other hand, team members with multi-knowledge, individual knowledge, and experience across different functional areas (Hyung-Jin Park, Lim, & Birnbaum-More, 2009) can be important enablers of cross-functional work in software development projects. Concerning the former aspect, the preferable organizational setup for UCASD is yet to be determined. The options are to appoint two dedicated teams to handle design and development activities, respectively, or to establish a cross-functional team that handles both tasks (Ferreira et al., 2010, 2012). The latter is in line with ASD, which

promotes the idea of a team including "people with all the skills and perspectives necessary for the project to succeed" (Beck, 1999, p. 38). Schwaber and Sutherland (2013) emphasize that an agile team should have "all competencies needed to accomplish the work without depending on others not part of the team" (p. 4). Similarly, 34 (41.0%) of the papers included in the literature review describe a cross-functional setup, in which UCD expertise is available within an agile development team. According to Kuusinen et al. (2012), this is also practitioners' most frequent proposal to improve the cooperation between UCD experts and developers. As it is frequently mentioned, a cross-functional team allows for a direct and unmediated exchange of knowledge and information between design and development experts (Barksdale & McCrickard, 2012; J. M. Brown et al., 2011; Ferreira et al., 2012). In particular, it allows maximizing collaborative activities in order to create possibilities for knowledge transfers. As an example, Ungar and White (2008) suggest workshops in which designers, developers, and stakeholders collaborate and explore design alternatives as a forum for ad-hoc knowledge exchanges between the involved parties.

Suggestion. Design and development activities as part of the product creation phase are often described as parallel tracks. It has to be kept in mind that an interaction concept destined for implementation in an upcoming iteration has to be finished prior to the start of its development. Table 19 lists the codes that built the foundation for suggesting one further principle, i.e. parallel interwoven creation tracks, from a process perspective.

Table 19 – Codes Related to the Parallel Interwoven Creation Tracks Principle.

Included Codes	*Resulting Principle*
Parallel Tracks Design One Sprint Ahead Synchronization / Integration	Parallel Interwoven Creation Tracks

To summarize, design activities need to start one sprint ahead of the development activities. Given the parallelization, mechanisms need to be adopted to assure the synchronization and integration of such tracks.

> ***Principle 3 (Parallel Interwoven Creation Tracks):*** *In user-centered agile approaches, design and development should proceed in parallel interwoven tracks.*

Further empirical research is required to investigate different cross-functional UCASD creation track setups under consideration of contingency factors. Existing literature providing empirical insights into cross-functional integration in product development might serve as a useful starting point. Such insights would require further refinement to focus on UCASD and its outcomes on a team and product level.

Practices for Principles

The practices perspective considers concrete methods that are executed within the coding system. The literature included in the review discusses a large variety of practices (see Figure 18 in Appendix B). In an effort to consolidate this diversity of practices, principles for the practices dimension of UCASD should capture the rationale behind these practices rather than pay attention to the practices themselves. Consequently, practices conducted in an UCASD approach have been classified along two holistic principles.

Continuous Stakeholder Involvement

Focus Area. While ASD is inherently people-centric and the active involvement of stakeholders is one of its key elements (Nerur & Balijepally, 2007; Pikkarainen, Haikara, Salo, Abrahamsson, & Still, 2008), it lacks a clear distinction between the customer and the end-user of the software (Chamberlain et al., 2006). It is not unusual for one person, for example a domain expert or a product manager from the client organization, to fill both the customer and user roles (A. Martin, Biddle, & Noble, 2004). Such roles are not able to represent all stakeholders in the system (Beyer et al., 2004). However, UCD approaches require direct and unmediated contact with the end-user of the software (Gould & Lewis, 1985; Gulliksen et al., 2003).

Evidence. Various publications in the literature review discuss the decision between direct or mediated stakeholder relations from a general perspective without reference to specific concepts. Thereby, 17 (20.5%) of the included publications promote the *direct involvement* of stakeholders, i.e. direct and unmediated contact between stakeholders and design or development experts. In their ethnographic study focusing on the commonalities of agile methods and user-centered development, Chamberlain et al. (2006) confirm the importance of continuous stakeholder integration. In contrast, eight (9.6%) of the articles recognize different motivations to involve *stakeholder representatives*, which might be preferable considering resource constraints or difficulties to gain direct access to prospective users. Beyond the specific scope of UCASD, empirical literature has intensively researched the importance of user integration in software development (e.g. Harris & Weistroffer, 2009; Rasmussen & Christensen, 2011).

Going beyond the general perspective, two focal areas of stakeholder involvement in the context of UCASD became apparent in the review. The evaluation of the system can be recognized as the most important motivation to establish direct contact with customers, i.e. end-users. In particular, 37 (44.6%) of the reviewed publications discuss *usability testing* with end-users. Nevertheless, practices used in the early phases of software development, in which interactions with stakeholders are necessary to establish system requirements, are equally important. While 12 (14.5%) of the articles only take a generic perspective on

user research by highlighting the necessity to conduct user research or by discussing the inclusion of user research in an agile development process, 10 (12.0%) publications give explicit attention to *contextual inquiry* in an agile environment. Other practices for stakeholder involvement, including *task analysis* (6 publications, 7.2%), *focus groups* (4 publications, 4.8%), *interviews* (2 publications, 2.4%), and *surveys* (2 publications, 2.4%) are mentioned, but receive less attention.

Suggestion. Stakeholders' continuous involvement in the design and development process is one of the main commonalities of UCD and ASD. Both emphasize the human aspect of software development. The value of individuals and interactions is documented in the *agile manifesto* (Beck et al., 2001). According to UCD, understanding users and their tasks should be the focus of software development (Gulliksen et al., 2003). Moreover, UCD and ASD both encourage customer or user participation in the development of new systems or software. Figure 10 depicts the codes on continuous stakeholder involvement. Thus, the study suggests the following principle:

> ***Principle 4 (Continuous Stakeholder Involvement):*** *Stakeholders should be actively involved in user-centered agile approaches early on and should remain involved throughout the entire development process to collect input and feedback.*

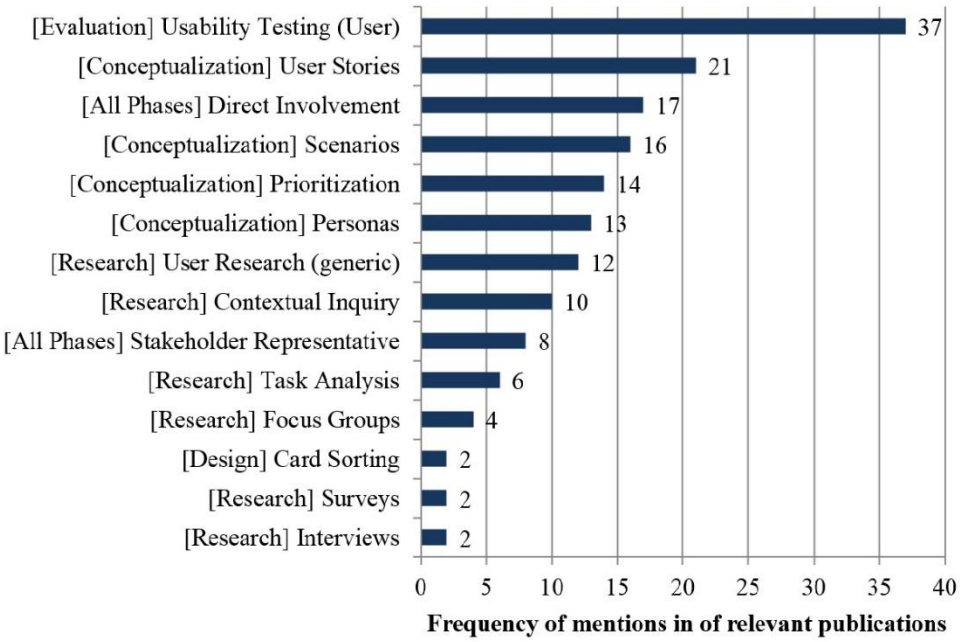

Figure 10 – Codes and Number of Articles Related to the Continuous Stakeholder Involvement Principle.

In the context of UCASD, there is a lack of empirical studies systematically investigating the continuous stakeholder involvement principle. Building on literature on user involvement (Harris & Weistroffer, 2009; Rasmussen & Christensen, 2011), the extent and outcome of continuous stakeholder integration in UCASD requires further empirical research. From a practical point of view, empirical insights could be leveraged to

give context-specific recommendations for applying different stakeholder integration strategies, using associated practices. A deeper understanding of contingency factors and their influence is required to derive corresponding recommendations for successfully applying this principle.

Artifact-Mediated Communication

Focus Area. In contrast to traditional plan-based approaches, ASD discourages the use of formal documentation, as it values "[w]orking software over comprehensive documentation" (Beck et al., 2001, (p. 1), and knowledge about products and processes is supposed to become tacit and should be exchanged through interactions between team members (Nerur, Mahapatra, & Mangalaraj, 2005). In order to reduce "waste," documentation should be reduced to a minimum, with the team only documenting what is absolutely necessary (Dingsøyr et al., 2012; Nerur & Balijepally, 2007). As Kuusinen et al. (2012) state, there are as yet no commonly agreed upon concepts for communicating design or documenting user requirements in UCASD. Nevertheless, while analyzing the document corpus in the literature review, it became apparent that, besides stakeholder involvement, the use of various *artifacts* related to the design and development process is most prominently discussed in research publications. Artifacts are defined as an "… aspect of the material world that has been modified over the history of its incorporation into goal-directed human action." (Cole, 1998, p. 117). This definition incorporates both the tangible and mutable nature of artifacts and their purposeful use. Artifacts, such as those described below, have a long tradition of being used for documentation purposes in design and development activities (J. Brown et al., 2008). Besides supporting creative processes, they are an important means for organizing communication and collaboration among internal and external parties.

Evidence. Discussed in 40 (48.2%) of the publications included in the literature review, prototypes, i.e. visual instantiations of the design concept (Buxton, 2007), is the most frequently occurring artifact type. However, only three papers provide empirical evidence supporting this principle (J. Brown et al., 2008; Kuusinen et al., 2012; J. C. Lee et al., 2009). Remaining very general, Lee *et al.* propose a new approach that combines user-centered and agile software development using central design records (CDRs) as artifacts (J. C. Lee et al., 2009). Kuusinen et al. (2012) focus on the interaction of the UX team and the development team in a large software development organization, identifying the creation and communication of designs as an important task of the UX team. Going into more detail, an ethnographic study by J. Brown et al. (2008) analyzes the role of artifacts during the interaction between design and development.

While J. Brown et al. (2008) mention prototypes with different levels of fidelity, they usually recommend transitioning from low-fidelity to high fidelity prototypes in design iterations. Additionally, 17 (20.5%) of

the included publications consider the use of sketches or mock-ups as artifacts to support the early ideation stage of the design process Buxton (2007), and nine (10.8%) articles mention the use of *wireframes* to depict the envisioned layout of a user interface. Kuusinen et al. (2012) describe challenges during the interaction of design and development efforts based on the example of a multi-continental development team. Two studies (2.4%) further discuss the use of multiple wireframes in a *user interface flow*, which is a set of wireframes that visualize which elements of a user interface are used in an interaction path through the system.

A second group of artifacts is inherently linked to stakeholder involvement, as the documentation of stakeholder properties and needs are also of interest to the user-centered agile community. The employment of *user stories* to describe features providing business value to the customer is popular, and mentioned in 21 (25.3%) of the publications in the review. While user stories stem from the agile world, the idea of modeling the system's value proposition in *scenarios* providing a step-by-step narrative on how a prospective user will benefit from the envisioned product is a user-centered development concept, and is discussed in 15 (19.3%) articles. Furthermore, the use of *personas* as concrete representations of user archetypes is highly popular in UCASD, and discussed in 13 (15.7%) publications. Finally, four (4.8%) of the included publications describe the idea of a CDR as a central artifact linking design goals, design decisions, and usability testing results (J. C. Lee et al., 2009). The use of a CDR supports the goal of reaching a cohesive design.

Suggestion. Artifacts are central means of communication for agile and user-centered software development. Figure 11 depicts the codes and number of articles discussing different types of artifact within the software development process.

While ASD focuses on working prototypes, UCD provides different artifacts to document stakeholder needs and communicate design ideas. On the one hand, user stories, personas, and scenarios are examples of conceptualizations of stakeholder needs. On the other hand, prototypes, wireframe, and mock-ups are established artifacts for communicating designs. To generalize these concepts on a higher abstraction level, the following principle is suggested:

> ***Principle 5 (Artifact-Mediated Communication):*** *In user-centered agile approaches, tangible and up-to-date artifacts should be used to document and communicate product and design concepts, and should be accessible to all involved stakeholders.*

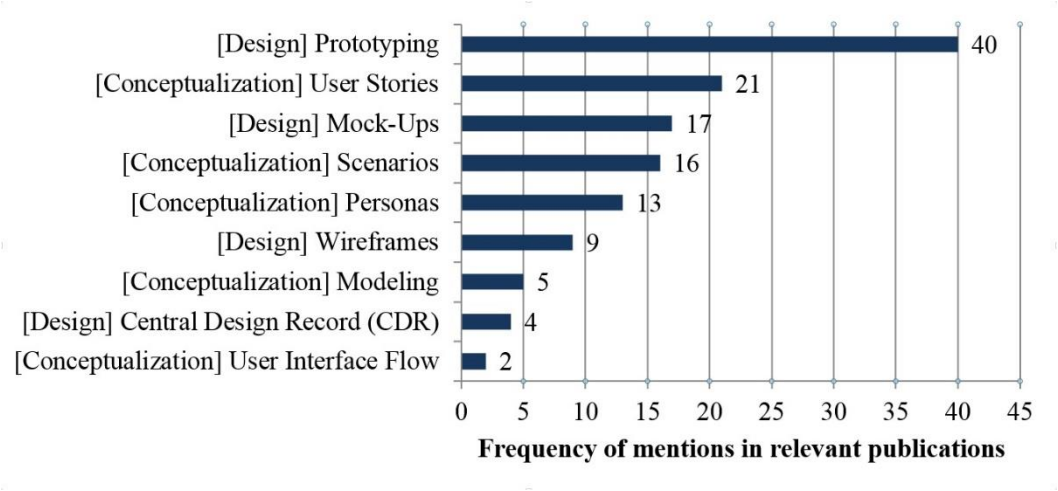

Figure 11 – Codes and Number of Articles Related to the Artifact-mediated Communication Principle.

For many years, artifact-mediation communication has been leveraged intensively in other industries, for example, clay modeling in the automotive industry when designing new cars or scale models used by architects to realize a new construction. Surprisingly, empirical evidence for this important principle in the context of UCASD is limited to three publications in the literature review. Following the line of argumentation in principle 4, empirical insights could be leveraged to make context-specific recommendations for applying corresponding artifact-mediated communication strategies and associated practices from a practical point of view. Again, a deeper understanding of contingency factors and their influence on design and development outcomes is needed to derive recommendations for successfully applying this principle.

3.1.6. Conclusion

This paper captures the current state of ASD and UCD integration and identifies generic principles that constitute an integrated UCASD approach. To achieve this, a systematic review of existing literature has been conducted. In total, 83 publications were identified as relevant and were analyzed along four dimensions. The analysis resulted in a differentiated coding system and created the foundation for suggesting the five principles as summarized in Table 20.

In order to interpret the implications of the findings adequately, the following limitations of the study need to be considered. Even though a systematic review should ensure a relatively complete census of the relevant literature, completeness can never be guaranteed. Accordingly, the study possibly missed some relevant articles (Webster & Watson, 2002). While the terminology used in the database query is commonly accepted and used within the scientific community, different terms may have been used to describe relevant methods. Moreover, although clear criteria have been established for assessing the selected articles' relevance (see subsection 3.1.3), the evaluation was based on the judgment and experience of the authors. Other scholars

might have judged these articles differently. The same limitation applies to the coding of each paper using the presented coding system. During the last step of the literature analysis, generic principles based on the identified codes have been derived. As discussed earlier, only the process and practices dimensions lend enough support to derive principles. The results for the people/social dimension were contradictory and therefore impeded generalization. The results for the technology dimension were thin and contained hardly any explicit recommendations.

Table 20 – Principles for User-centered Agile Software Development.

Principle	*Description*
Principle 1	Separate Product Discovery and Product Creation
Principle 2	Iterative and Incremental Design and Development
Principle 3	Parallel Interwoven Creation Tracks
Principle 4	Continuous Stakeholder Involvement
Principle 5	Artifact-Mediated Communication

Regardless of the depicted limitations, the literature review has led to the following theoretical contributions. First, a comprehensive coding system was derived. This system allows for a broader classification of existing works on UCASD, that scholars who are working in the area of UCASD can use. Besides the application in this paper, the coding system might be used to classify and evaluate future research in this area. Second, the current state-of-the-art in UCASD has been depicted based on the derived coding system. Providing a broad overview of existing work, this compilation can be used as a starting point for scholars who want to research this area. From a practical point of view, the derived principles and their association with specific practices and processes might help scholars apply UCASD in specific context. Consequently, this work might contribute to reaching the major goal of UCASD: The delivery of useful *and* usable software.

Our systematic literature review did focus on the 10 years' time frame of 2002 to 2012. As user-centered agile software development is an active research field, as confirmed by a crosscheck of the results with most recent publications in the years 2013 and 2014 using the same search strings and the same databases. Consequently, a set of 487 papers has been identified. After a screening of the titles and abstracts, 26 papers actually focusing on the subject of user-centered agile software development have been analyzed further and classified along the identified dimensions as depicted in the Table 21.

In the process dimension the identified papers describe integrated approaches combining the two fields, confirm earlier prescriptions and discuss key challenges with regards to this dimension: E.g. models addressing the trade-offs between usability and agile methods (Butt, Ahmad, & Rahim, 2014) and models supporting the collaboration between software engineering and usability experts are suggested (Humayoun,

Dubinsky, & Catarci, 2014b; Wan Ahmad, Butt, & Rahim, 2013). Moreover, established concepts, such as sprint 0, iterative evaluation and designing one sprint ahead are further supported (Bertholdo et al., 2014; T. S. da Silva, Silveira, & Maurer, 2013; Kuusinen, 2014; Plonka, Sharp, Gregory, & Taylor, 2014). It is recognized that usability methods are often used too late in the development process. Managing the product vision and time boxing the user-centered design work are well-known continuous challenges (Kuusinen, 2014; Plonka et al., 2014; Salvador, Nakasone, & Pow-Sang, 2014). Overall, one can conclude that the three principles identified in the systematic literature review have been confirmed by recent research in the field.

Table 21 – Overview of Recently Published Papers in 2013/2014 along the Four Dimensions.

Dimension	Included Publications
Process	Butt, Ahmad, & Rahim, 2014; T. S. da Silva, Silveira, & Maurer, 2013; Humayoun, Dubinsky, & Catarci, 2014b; Kuusinen, 2014; Maurer & Hellmann, 2013; Plonka, Sharp, Gregory, & Taylor, 2014; Salvador, Nakasone, & Pow-Sang, 2014; Wan Ahmad, Butt, & Rahim, 2013
Practices	Ardito et al., 2013; Arnowitz, 2013; Bertholdo et al., 2014; Caballero, Moreno, & Seffah, 2014; Cajander, Larusdottir, & Gulliksen, 2013; T. S. da Silva, Silveira, & Maurer, 2013; Häger et al., 2015; Inayat, Salim, Marczak, Daneva, & Shamshirband, 2015; Kuusinen, 2014; Lizano, Sandoval, & Stage, 2014; Maurer & Hellmann, 2013; Peres et al., 2014; Plonka et al., 2014; Salah et al., 2014c; Salvador et al., 2014
People & Social	Arnowitz, 2013; Bertholdo et al., 2014; Cajander et al., 2013; T. S. da Silva, Silveira, & Maurer, 2013; Häger et al., 2015; Heimgärtner & Solanki, 2014; Jurca et al., 2014; Kropp & Koischwitz, 2014; Kuusinen, 2014; Kuusinen et al., 2012; Lizano et al., 2014; Plonka et al., 2014; Raison & Schmidt, 2013; Wale-Kolade, Nielsen, & Päivärinta, 2013
Technology	Humayoun et al., 2014b; Salvador et al., 2014

In similar vein, the identified papers related to the practices dimension provide further contextualized and enhanced integration concepts and describe specific challenges. While practices from user-centered design in Scrum improve the understanding of actual user's needs, challenges remain in the actual application of usability practices (Kuusinen, 2014; Lizano et al., 2014). Recent work has suggested patterns for the integration of ASD and UCD (Caballero et al., 2014) and a framework for integrating practices (Peres et al., 2014). However, the prioritization of non-functional usability requirements in comparison to functional requirements remains challenging (Inayat et al., 2015; Plonka et al., 2014). Fast prototyping, individual inquiry, formal tests, and heuristic evaluations are recognized as the most frequent usability practices (Salvador et al., 2014). Specifically, Low-Fi prototypes and user story maps are suggested as strategic and cost-effective concepts for creating artifacts (Maurer & Hellmann, 2013). Specifically looking at distributed teams, digital low-fidelity prototypes are suggested as a powerful concept to improve the communication of designers and developers (Ardito et al., 2013; T. S. da Silva, Silveira, & Maurer, 2013). Beyond these practices for stakeholder integration and artifact-mediated communication, the establishing of explicit usability goals and acceptance criteria has been discussed (Arnowitz, 2013; T. S. da Silva, Silveira, & Maurer, 2013). In sum, recent research was conducted that helps to further refine and contextualize

UCASD practices. However, the translation of UCASD practices from academia into practice remains challenging (Ardito et al., 2013), future empirically grounded research is required.

Recently, there has been increasing interest in the people/social dimension of UCASD. A clear definition for the responsibility of the user perspective within a scrum team is needed (Cajander et al., 2013), as such role influences the product owners' satisfaction (Kuusinen & Mikkonen, 2013). It is recognized that the usability professional needs to be a full member of the agile team (T. S. da Silva, Silveira, de O. Melo, & Parzianello, 2013) with clearly defined responsibilities and authority (Cajander et al., 2013; Kuusinen, 2014). The expected contribution may change throughout different development stages (Kropp & Koischwitz, 2014), being one of the key challenges for its integration (Lizano et al., 2014). While additional usability training of developers might be a tempting solution (Lizano et al., 2014), usability professionals require encouragement, authority, and enough time to accomplish her tasks (Arnowitz, 2013; Jurca et al., 2014; Kuusinen, 2014). Moreover, her co-location with the development is a key enabler to improve overall communication. It is important to break old habits in the development team (Kropp & Koischwitz, 2014).

Overall, contextual factors such as organizational support and cultural change are not to be neglected, as they may lead to power struggles between usability professionals and developers, or a lack of required usability resources for the actual work (Jurca et al., 2014). Strategic and ongoing support, and a collaborative culture are recognized as important factors within an organization facilitating the integration of ASD and UCD (Cajander et al., 2013; Raison & Schmidt, 2013). Particularly managers and product owners should be drivers of this cultural change (Cajander et al., 2013; T. S. da Silva, Silveira, de O. Melo, et al., 2013). As a result of an adaptation process in mindset and culture, a team's creativity, pro-activity and speed can be improved (Heimgärtner & Solanki, 2014). In sum, the research community has a growing interest to look at the people/social dimension of UCASD. Still, as discussed earlier in the literature review, it is challenging to organize the manifold contextual factors into stable principles to enable meaningful guidance for successfully applying UCASD in practice.

Finally, only very limited new suggestions have been identified regarding the technology dimension of UCASD. One approach for the automation of usability methods application to reduce cost is described in (Salvador et al., 2014). Furthermore, a prototypical example for the integration of an integrated UCSAD approach within an IDE to provide guidance to developers (Humayoun et al., 2014b). As mentioned earlier, research concerning the technology dimension of UCASD seems to be currently still in an early stage.

Future research might extend the work presented in this paper. It became obvious during the deeper analysis of the identified UCASD publications in the systematic literature review that only a limited number of papers present rigorous empirical findings. Future research may specifically investigate the identified UCASD

principles from an empirical point of view. In this context, it is highly relevant to gain a better understanding of the influence of contingency factors, such as software product, team, organization, or further characteristics. In addition, future research should focus on identifying further principles in the people/social as well as the technological dimension of UCASD. Specifically, instead of analyzing existing UCASD literature, subsequent research could explore whether knowledge from other domains (e.g. organizational science or sociology for the people/social dimension and other research streams of computer science for the technology dimension) is applicable to UCASD. This may result in additional, people- and technology-related principles, which could further extend the results. Second, complementing and building on empirical research, the identified principles can be deployed in real-world software development projects following an (action) design research approach (Sein, Henfridsson, Rossi, & Lindgren, 2011). Expected outcomes of this type of research could be the creation and evaluation of a UCASD procedure model supporting the execution of software development projects targeting the delivery of useful and usable software.

3.2. PDISC - A Method for Software Product Discovery[5]

3.2.1. Introduction

Early activities within the design and development of products are key in delivering innovation (R. G. Cooper & Kleinschmidt, 1994). Decisions and choices made in this phase influence the subsequent activities within a software development project. Therefore, incorrect decisions can lead to a significant waste of organizational resources (R. G. Cooper & Kleinschmidt, 1994). However, these early activities, often described as product discovery, provide new challenges to the product teams and yet, we know little about their extent and consequences (Brhel, Meth, Maedche, & Werder, 2015; Larusdottir et al., 2017). Because of the cost escalation factor of early decisions (Boehm, Rombach, & Zelkowitz, 2005), balancing a products viability, desirability, and feasibility are key elements of early activities (Brhel et al., 2015).

When developing new software, product teams mainly rely on one of two central paradigms. On the one hand, we have user-centered design (UCD) that has the objective to deliver usable software (ISO, 2010). On the other hand, we have agile software development (ASD) that emphasizes the usefulness of software (Beck et al., 2001). Prior research suggests that both tend to focus on central elements of the development and design process and tend to neglect early activities (Brhel et al., 2015; Fox et al., 2008). UCD generally acknowledges the need for upfront design activities. Yet, there is disagreement about its extent (Brhel et al., 2015). Some argue for LDUF, while others call for the need of big upfront design (T. S. da Silva, Martin, et al., 2011). On the contrary, ASD lacks the description of early activities entirely (Fox et al., 2008). Hence, a method needs to be developed that guides product teams in their early activities. Moreover, the method needs to enable the product team to conduct either an extensive or focused product discovery phase.

Product discovery involves two central elements, analysis and design (T. S. da Silva, Martin, et al., 2011; Fox et al., 2008). Especially the analysis of users is an often mentioned fundamental research activity (Chamberlain et al., 2006; T. S. da Silva, Martin, et al., 2011; D. Sy, 2007). In order to conduct these activities, different techniques have been suggested, such as contextual interviews and task analysis (K. Bodker, Kensing, & Simonsen, 2009; T. S. da Silva, Martin, et al., 2011). Scholars suggest an artifact-centered communication approach (Brhel et al., 2015) by utilizing early sketches (Buxton, 2007), low-fidelity prototypes (Smith, 1991) or personas (A. Cooper, 1999). Both elements are often decoupled within literature.

[5] This section is based on Werder, Zobel, and Maedche (2016, 2017)

Therefore, the study answers calls for more research on product discovery (Brhel et al., 2015; T. S. da Silva, Martin, et al., 2011; Larusdottir et al., 2017) and suggest a new method for software product discovery called PDISC. The research objectives are to: a) extract knowledge from existing literature on methods regarding the discovery of software products, b) formalize existing methods from such literature to establish clear design requirements and principles, and c) translate those principles into a formalized method for software product discovery. Consequently, the following research questions are formulated:

RQ1: How can a software discovery method be created?

RQ2: What challenges and recommendations documents the literature for the discovery of software products?

RQ3: Which activities and deliverables exist in the literature?

RQ4: What principles guide a software product discovery method?

The study provides three main contributions. First, the study identifies important activities and deliverables that collectively form the product discovery phase as a result of a systematic literature review (Brhel et al., 2015). Together, the collected activities and deliverables form a checklist for practitioners, allowing them to assess their instantiated product discovery process for completeness and comprehensiveness. Second, the study presents a meta-model that captures clear sequences of activities and deliverables (Larusdottir et al., 2017). In addition, it creates relationships between activities and deliverables, and matches each activity to a specific (sub-) deliverable. Practitioners can use the evaluated meta-model to guide their own instantiation of a discovery process. Third, the study identifies and presents principles that guide the method construction. These allow for a rule based examination of different discovery phases (Gregor & Hevner, 2013).These principles help to articulate the overall goal of a product discovery phase and represent generic statements for the method design. A categorization along product-, user- and team-related requirements is suggested in order to address concerns of viability, desirability, and feasibility of the envisioned product.

3.2.2. Foundations

Product Discovery

The term product discovery has been heavily used in the pharmaceutical domain and the area of drug discovery (e.g. Nwaka & Hudson, 2006). In recent years the term is used to describe a phase of upfront activities preceding the product development and product design phases (Brhel et al., 2015). Hence, it describes early steps before the elicitation and engineering of more fine granular requirements. Others describe the phase as ideation generation stage (R. G. Cooper, 2011). In this study, the term product

discovery refers to the activities that are conducted before the actual design and development of a software product begins (Brhel et al., 2015). While a product vision is often a central deliverable of a product discovery phase, different objectives and goals are associated with it. Within UCD, upfront activities have been mentioned as means to communicate a clear product vision (Brhel et al., 2015). Examples are little design upfront, sprint zero, cycle zero or phase zero as specific terms used to describe such early activities (T. S. da Silva, Martin, et al., 2011).

Within the domains of new product development and innovation management, product discovery was introduced as a so called front-end phase, with buzzwords such as "fuzzy front-end" and "front-end innovation" (Frishammar, Florén, & Wincent, 2011; Sperry & Jetter, 2009; Stevens, 2014). Stemming from an engineering domain, terms such as product exploration, product scoping and product ideation are used to guide the early activities prior to development (Brhel et al., 2015; R. G. Cooper, 2011). The key goal of these phases is to reduce uncertainty and equivocality that are largely present during the early stages of product development (Frishammar et al., 2011; Kakar & Carver, 2012; Khurana & Rosenthal, 1998; Sperry & Jetter, 2009; Stevens, 2014). Therefore, a method summarizing and structuring early activities related to software development and design helps to reduce uncertainty.

Product discovery helps to determine the actual need for a product and the existence of a user base on the one hand, and the actual feasibility of such a solution on the other hand (Cagan, 2007). Hence, different goals are involved in the discovery of software products (Cloyd, 2001). Three areas are identified, i.e. the business driving the discovery of a new product, the product team conducting the main activities, and the user that is expected to use the product. First, the business often triggers the discovery of a new product and therefore, conducts a market analysis or supplies strategic goals and values (R. G. Cooper, 2011). An organization will emphasize its business goals, values and mission within the discovery process to assure alignment with its current path and viability of the product (R. G. Cooper, 2011; Mintzberg, 1978). Second, the product team is typically cross-functional and conducts the main activities, such as generating ideas (Liedtka, 2015) or building a prototype (R. G. Cooper, 2011). Technology-driven organizations tend to focus on the product's feasibility (Pavlou & El Sawy, 2006). Third, the user is key in order to test ideas, concepts, or early prototypes (Shah, Rust, Parasuraman, Staelin, & Day, 2006). Inviting the user to participate in the development goes even a step further and enhances the team's innovativeness (K. Bodker et al., 2009). User-centric organizations look outwards to assess their early prototypes with prospective users, gaining confidence in its desirability (K. Bodker et al., 2009). A discovery process needs to respond to all three focus areas in order to increase the chances for the product's success.

Development and Design of Software

When creating a product, two paradigms dominate. On the one hand, we have an engineering-driven approach of product development and on the other hand, we have a design-driven approach referred to as product design (R. G. Cooper, 2011). There are corresponding specializations for the development and design of software products, namely ASD (Agile Software Development) and UCD (User-Centered Design) (Beck et al., 2001; ISO, 2010). While ASD is one possible representation of product development phase, UCD serves as representation of product design (see Figure 12). Often, there is no clear line between both approaches. For example, product design is also a key element in various software development methods. Some authors describe it as conceptualization of a solution prior to programming activities (Freeman & Hart, 2004). Product development, in turn, describes the creation or implementation of software artifacts, e.g. by programming (Boehm, 1988). In iterative or agile frameworks these two phases are executed multiple times, allowing multiple feedback cycles (e.g. Brhel et al., 2015; Fox et al., 2008).

Consequently, scholars explore the combination of both methods. For example, blending development practices and techniques with those established in the design discipline (e.g. Barksdale & McCrickard, 2012; Brhel et al., 2015; Fox et al., 2008). Fox et al. (2008) suggest a method combining ASD and UCD including a cycle zero and parallel, yet interwoven tracks. Focusing on the individual, da Silva, Silveira, de O. Melo, and Parzianello (2013) identify the role of UX designers within agile teams. Others propose a model for the integration of interaction design into ASD (T. S. da Silva, Silveira, & Maurer, 2011). In a similar vein, Ferreira, Noble, and Biddle (2007) suggest four steps towards the cooperation of user experience designers and agile developers. Brhel et al. (2015) identify five principles along the processes and practices of UCD and ASD domains, establishing a user-centered agile software development approach.

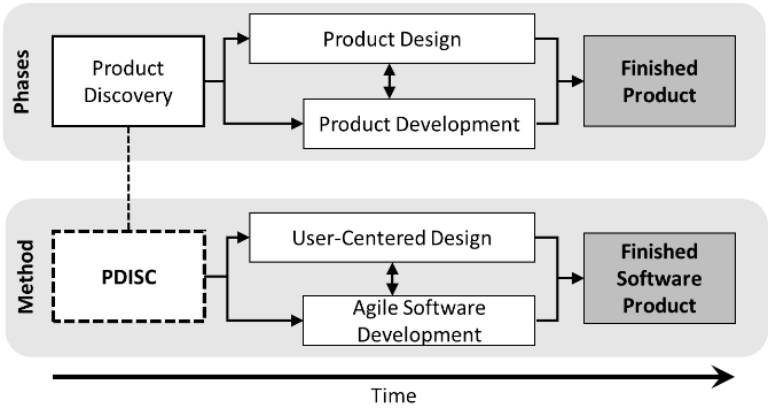

Figure 12 – Placing the Product Discovery Method within Existing Methods and Phases.

Thus, both methods have their commonalities (e.g. Brhel et al., 2015) and can complement each other (e.g. Fox et al., 2008). For example, both methods build on an iterative process. While ASD iterates through

development tasks to deliver working software, UCD iterates through design and usability evaluations (Fox et al., 2008). Furthermore, the people-element plays a much larger role in both approaches than it does in more classical or sequential methods. Even though the focus of ASD towards the customer is different than the user-focus of UCD, both follow a human-centric approach (Fox et al., 2008).

The combination of ASD and UCD also has its challenges. One of the challenges of ASD that harms the goal of UCD is the heterogeneity of end-users and customers. While the focus of UCD lies with the end-user through user involvement, ASD focuses primarily on the customer. Moreover, while ASD demands unit-testing, UCD requires usability-testing (Brhel et al., 2015; Sohaib & Khan, 2010). Consequently, when sticking solely to agile practices, there is no guarantee that the developed software provides any level of usability (Fox et al., 2008; Sohaib & Khan, 2010).

Work related to method engineering suggests the approach of method construction as a tool for organizational engineering (Braun, Wortmann, Hafner, & Winter, 2005). More recently, the importance of methods for modelling in the context of organization design and enterprise engineering has been pointed out (Winter, 2017).

3.2.3. Research Method

The study opts for a systematic literature review (SLR) in order to summarize and synthesize literature in a transparent and structured way (Kitchenham & Charters, 2007; vom Brocke et al., 2009). Therefore, the influence of biases introduced by the researcher is limited and the rigor of the results increased (Hevner, 2007; Kitchenham & Charters, 2007). Prior research indicated the need for specifying the software product discovery phase (Brhel et al., 2015). Hence, the study follows an evidence-based approach to develop generic statements for the method's design, so that these can be instantiated them thereafter (Kitchenham, Dybå, & Jorgensen, 2004). Such evidences are design requirements and method fragments, both are elements that form the method data (Brinkkemper, Saeki, & Harmsen, 1999; Henderson-Sellers, Ralyté, Ågerfalk, & Rossi, 2014). As a result, the study presents PDISC; a method for software product discovery based on prior literature.

The results are evaluated using expert interviews. The interviews follow three main objectives (Schultze & Avital, 2011). First, the interviewer gains insights from the expert's practical experience (appreciative approach). Second, the interviewer questions the expert for feedback on the method from generic to specific elements (laddering approach). Third, the study communicates the design principles and their proposed instantiation in the form of design knowledge (Gregor & Hevner, 2013).

This literature study has four aims: i) develop a method base (RQ1), ii) identify existing challenges and recommendations (RQ2), ii) identify articles that articulate early activities and deliverables (RQ3), and iv) develop a list of requirements (RQ4). A method base accumulates all collected method fragments (Brinkkemper et al., 1999). Those method fragments formalize early activities and deliverables from the literature within process-deliverable diagrams (PDDs).

Search and selection process

A systematic process and transparent documentation of the literature allows the reader to assess the completeness of the review (vom Brocke et al., 2009). Hence, the first step in conducting a SLR is the development of a study protocol (Kitchenham & Charters, 2007). The protocol documents the main research questions, key decisions along the scope (e.g. search strategy; databases; inclusion, exclusion, and quality criteria), and a concept-matrix. Figure 13 presents an overview of the search strategy.

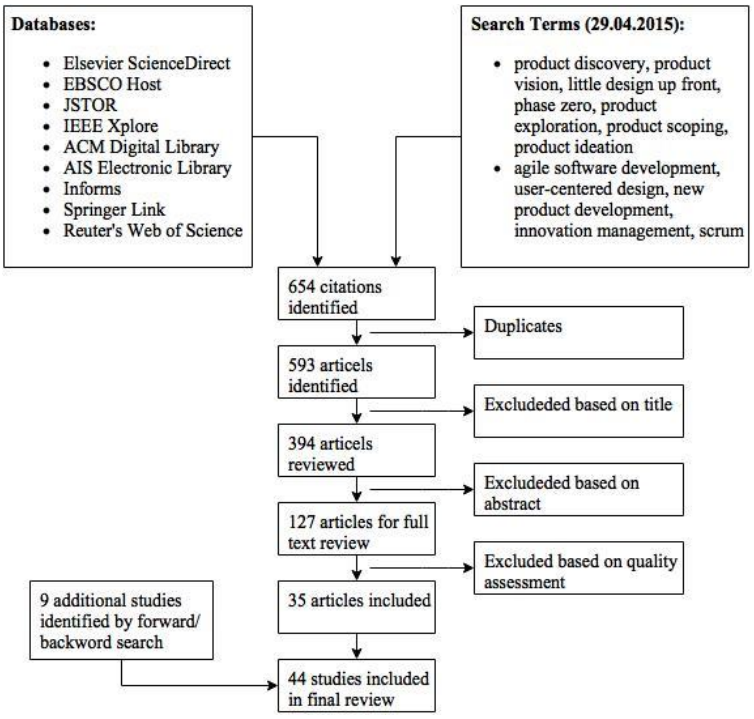

Figure 13 – Search Strategy Diagram.

The sources focus on databases that include publications from the field of information systems, computer science, and general management. In response to some critics (Boell & Cecez-Kecmanovic, 2015), a rather inclusive approach is followed to identify also less impactful, yet relevant publications in the search results. Nine databases with varying search results are included (see Table 22).

Table 22 – Overview of Selected Databases, their Initial Results, and Primary Articles.

Database	**Initial Results**	**Primary Articles**
ACM Digital Library	46	4
AIS Electronic Library	30	1
EBSCO Host	27	1
Elsevier ScienceDirect	158	7
IEEE Xplore	132	16
Informs	6	0
JSTOR	62	0
Reuter's Web of Science	9	0
Springer Link	184	6
Snowballing	-	*9*
Total	**654**	**44**

A primary challenge for SLRs is the development of the search string. A preceding exploratory literature search identifies synonyms and develops a keyword-matrix. The matrix identifies and evaluates prevalent matches in the literature. Moreover, it helps to assure the comprehensiveness and quality of the search results. A pilot test using IEEE as the sample database helps to evaluate the search string. All resulting 132 articles are scanned for comprehensiveness and quality. Therefore, the following search string is formulated and applied to all nine databases:

("product discovery" OR "product vision" OR "little design up front" OR "phase zero" OR "product exploration" OR "product scoping" OR "product ideation")

AND

("agile software development" OR "user-centered design" OR "new product development" OR "innovation management" OR "scrum")

Inclusion and exclusion criteria

The study investigates a timeframe from 1997 onwards until mid-2015. W.r.t. the product discovery phase of software products, articles are included that:

- provide insights on potential shortcomings,
- describe possible solutions, or
- present recommendations.

Articles are excluded that:

- duplicated articles
- irrelevant industrial focus, such as biology and pharmacy or industrial engineering, or
- low quality and relevance based on individual assessment.

Assessment

After reducing the articles by applying the aforementioned inclusion and exclusion criteria, the resulting articles are evaluated along four questions (Kitchenham & Charters, 2007). Each article receives a score on a 5-point Likert scale, whereas a score of one indicates a low relevance and a score of five indicates a high relevance to this study. Articles are assessed whether:

- a goal is mentioned by the authors,
- the article presents empirical data,
- shortcomings or challenges related to product discovery are mentioned, and
- recommendations to product discovery are documented.

Articles scoring one or two points are excluded. As a result, 35 articles are relevant to this study. They distribute as follows: eight articles receive a score of three; 12 articles receive a score of four, and 15 articles receive a score of five. Using these 35 articles, a forward and backward search is conducted (Webster & Watson, 2002), leading to the inclusion of another nine articles. Hence, 44 articles are included in this study in this study (see Table 39 in Appendix D). Results are reviewed and discussed within the author team. In case an article is not directly accessible for the full text review through one of the researcher's institute library system, the authors of the article are contacted through email or ResearchGate[6] (Jennex, 2015).

Data extraction

An a priori developed concept matrix guides the data extraction process (Webster & Watson, 2002). The study analyzes 44 articles for their shortcomings and proposed actions related to product discovery. The database stores information related to each article and its content. Meta-data for each article is extracted, including:

[6] http://researchgate.net – is a networking site for scholars. It facilitates the exchange of publication between scholars and serves as a question and answer site.

- authors, year, title,
- source database (IEEE vs ACM, etc.),
- assessment rating (score of 1-5),
- publication type (journal or conference), name of the outlet and pages of the article

Moreover, the content from each article is systematically extracted including:

- dimension (shortcoming or challenge, recommendation),
- corresponding activity,
- phase and method,

While 28 articles suggest challenges of current practices in a discovery phase, 31 articles provide recommendations on the implementation of early activities. Articles proposing activities are potential contributions towards the method database.

Based on these data collection steps, the study finds nine articles with concrete and multiple activities that form the method database. These articles provide evidence for the development of the method (Brereton et al., 2007). Therefore, the following information are documented for each article (Brinkkemper, 1996):

- a PDD, including all activities, executing role, phases, deliverables and relations
- an activity table, defining all activities
- a concept table, defining all concepts

This resulted in a total of 20 activities and 30 deliverables.

Data aggregation and synthesis

The data is aggregated and synthesized in three steps in order to answer the research questions (Brereton et al., 2007). First, the authors highlight critical sections during the full text review and aggregate them before transferring them into the research database (Seaman, 1999). This step aggregates existing challenges and recommendations. In addition, a list of existing activities and deliverables is built from the literature. During the analysis, closed coding is used to categorize text fragments on a higher level. Following, clusters are formed within each category, e.g. along the commonalities. Second, descriptive statistics, visualization, and tabulation of the data provide an overview of the literature. More specifically, a tabulation of the data by years and sources help to explore the overall data quality and their origins. The papers are presented along

the identified challenges, recommendations, and evaluation score in tabular form. Third, based on the challenges and recommendations, design requirements and their corresponding design principles are documented. Different requirements serve as categories in the coding process. Thereafter, the final method is constructed, instantiating the design principles. The method base helps to compare and aggregate existing method fragments. Each design decision bases on previously derived design principles, while still maintaining the link to the identified literature. The design decisions guide the development of the method, based on the challenges and recommendations identified in the underlying literature, as well as the identified method fragments.

Evaluation

Following the design and development of PDISC, experts are interviews in order to evaluate the results (Schultze & Avital, 2011). The goal of the evaluation is to assess the comprehensiveness and quality of the method. In order to increase the validity of the evaluation, 10 expert interviews from eight different roles are conducted. Different experts and roles (manager, designer, developer, and academic) are mixed to get a comprehensive evaluation (see Table 23). The interviews are transcribed based on audio recordings. As two interviewees do not consent to capturing the interview through recording, these transcripts are based on notes and a verbatim report from memory.

Table 23 – Overview of Interview Roles, Interview Duration, and Interview Type.

Perspective	**Role**	**Experience**	**Time/ M:S**	**Words**	**Expertise**
Development	Lead Software Developer	> 10 years in Software Development	31:46	2958	Agile Projects, Scrum
Management	Group Leader	> 15 years in Software Consulting	55:30	4968	Software Testing, Agile Projects
Development	Lead Software Developer	> 15 years in Software Development	35:44	2864	Agile Projects
Management	Department Manager	> 15 years in Project Management	51:00	6345	Agile Projects
Management	Software Project Manager	> 10 years in SD, > 5 years in Project Management	20:00	792	Software Development, Software Testing, Project Management
Management	Software Project Manager	> 10 years in Software Consulting (Management)	42:00	5929	Consulting, Software Management
Design	Software Designer	> 15 years in Software Design and Modeling	28:30	3000	Design and Agile Modeling
Management	Customer Manager	> 20 years in Management	35:00	1547	Customer Management
Consulting	Business Consulting	> 20 years in Consulting and Methods	22:40	2155	Requirements Engineering, Methods, Formal Processes
Research	Software Business Researcher	> 1 year SD > 2 years Product Management	24:00	1847	Software Product Development
SUM	**5:46:10**			**32405**	

Following a brief introduction, the interviews start with an appreciative interview approach (Schultze & Avital, 2011). The extraction of unbiased insights and experiences from the expert helps to assess the methods comprehensiveness. During the interviews, questioning starts with a retrospective view and moves towards a prospective view. The second part of the interviews follows a laddering approach (Schultze & Avital, 2011). The laddering approach starts from the general towards the detailed and specific feedback. Such approach enables the specific evaluation of the method and therefore, assesses its quality.

For the data analysis, categories along the elements of the PDD are created (Brinkkemper et al., 1999), i.e. related to the activities, deliverables and overall structure. Structural elements relate to the connection between different activities and/or deliverables. The experts may suggest the addition, removal, or adjustment of specific activities used on the process side of the PDD. Likewise, the experts may suggest adding, removing, or adjusting one of the deliverables within the PDD.

3.2.4. Results

The study presents the results of the systematic literature review, following the structure of the study's research questions in subsection 3.2.1. When looking at the number of publications per year, the subject has momentum (see Figure 14).

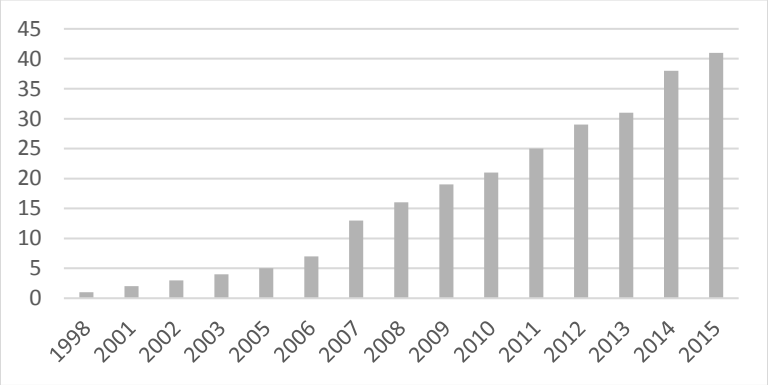

Figure 14 – Cumulative Number of Publications.

Challenges for Software Product Discovery

The study identifies key challenges and recommendations from the extant literature. Overall, the results suggest three related groups of challenges, i.e. product-related, user-related, and team-related. Six common challenges from 28 (64%) articles are identified (see Table 24). Nine articles indicate a lack of focus on hedonic (e.g. user experience) or pragmatic qualities (e.g. usability) in order to include users within early activities (Sohaib & Khan, 2010). Seven articles indicate that exploratory activities (e.g. little design

upfront) are missing or left out for various reasons e.g. due to time constraints (Salah, Paige, & Cairns, 2014a; Salah et al., 2014c). Five articles articulate the imprecision and uncertainty of requirements early on (Inayat et al., 2015). A lack of discussion and communication of the product's content is mentioned by five articles (Vanhanen, Itkonen, & Sulonen, 2003). The scalability of agile methods is a challenge mentioned by four articles (Qumer & Henderson-Sellers, 2007). Four articles also mention the lacking integration of customers and users into the project early enough (Kuusinen, 2014).

Table 24 – Challenges Related to the Discovery of Products Mentioned in Primary Articles.

Dimension	Challenges	References
User (9 articles)	There is not enough focus on usability and user experience	Adikari et al., 2009; Brhel et al., 2015; Fox et al., 2008; Patton, 2002; Salah et al., 2014a, 2014b; Salvador et al., 2014; Sohaib & Khan, 2010; Williams & Ferguson, 2007
Product (7 articles)	Exploratory activities are missing or left out	Brhel et al., 2015; Cloyd, 2001; Ferre & Medinilla, 2007; Kakar & Carver, 2012; Salah et al., 2014a, 2014b; Sibghatullah et al., 2006
Product (5 articles)	Requirements are not specified sufficiently	Adikari et al., 2009; Hollis & Maiden, 2013; Inayat et al., 2015; Kajko-Mattsson & Nyfjord, 2009; Liskin, 2015
Product (5 articles)	Content of the product is not communicated or discussed clearly	da Silva et al., 2011; Ebert, 2006; Nyfjord & Kajko-Mattsson, 2008; Tessarolo, 2007; Vanhanen et al., 2003
Team (4 articles)	Agile does not scale well to larger teams or projects	Gamble & Hale, 2013; Heikkilä et al., 2015; Qumer & Henderson-Sellers, 2007; Vanhanen et al., 2003
User (4 articles)	Customers and users are not integrated early enough	Hildenbrand & Meyer, 2012; Kuusinen, 2014; Rejeb et al., 2008; Sohaib & Khan, 2010

Besides the mentioned challenges, the study also finds recommendations within the literature base (RQ1). Those recommendations indicate best practices and successful activities during the early activities of a product development project (see Table 25). Overall, nine recommendations from 31 (70%) articles are identified. Most often, recommendation to create and specify a product vision are suggested by the literature (e.g. Tessarolo, 2007). Besides others, there are two benefits to it, i.e. the integration of business related questions in order to form a viable idea and the utilization of the vision as a communication artifact that documents the common understanding across the team members (Sarpong & Maclean, 2012; Stevens, 2014). Building on the latter, little design upfront utilizes low fidelity prototypes to enhance the team's communication (T. S. da Silva, Silveira, et al., 2011). Newly generated ideas are documented within an idea pool (Vanhanen et al., 2003). Later, such ideas inform the product backlog. Such a backlog helps the team to prioritize high-level requirements during this early phase and simplifies the integration of the iterative

idea into the process (Hildenbrand & Meyer, 2012). In order to enrich the integration with early design activities, a sprint zero allows the team to conduct design activities before the first development sprint (Kuusinen, 2014). Continuous feedback loops within the team, but also beyond the team to include users as early as possible help to find and shape a meaningful product concept (Inayat et al., 2015). In addition, early and high-level requirements are captured and documented in a requirement specification (Sibghatullah et al., 2006). Clearly defining each team member's role and responsibilities reduce ambiguity and conflict potential (Honious & Clark, 2006).

Table 25 – Recommendations Related to the Discovery of Products Mentioned in Primary Articles.

Dimension	Recommendations	References
Product / Team (15 articles)	Develop and constitute a product vision	Cloyd, 2001; Ebert, 2006; Ferreira et al., 2007; Hildenbrand & Meyer, 2012; Hollis & Maiden, 2013; Jain & Suman, 2015; Kajko-Mattsson & Nyfjord, 2009; Kakar & Carver, 2012; Nyfjord & Kajko-Mattsson, 2008; Salah et al., 2014b, 2014a; Sibghatullah et al., 2006; D. Sy, 2007; Tessarolo, 2007; Vanhanen et al., 2003
User (12 articles)	Integrate the user as early as possible	Barksdale & McCrickard, 2012; Fox et al., 2008; Hildenbrand & Meyer, 2012; Inayat et al., 2015; Miller, 2005; Patton, 2002; Rejeb et al., 2008; Salah et al., 2014a, 2014b; Sibghatullah et al., 2006; Sohaib & Khan, 2010; D. Sy, 2007
Team (9 articles)	Conduct little design upfront with low-fid prototypes	Adikari et al., 2009; Brhel et al., 2015; da Silva et al., 2011; Ferreira et al., 2007; Fox et al., 2008; Inayat et al., 2015; Salah et al., 2014a, 2014b; Salvador et al., 2014
Team (8 articles)	Establish a sprint zero	Brhel et al., 2015; da Silva et al., 2011; Kuusinen, 2014; Miller, 2005; Salah et al., 2014a, 2014b; Stevens, 2014; D. Sy, 2007
Product (4 articles)	Collect requirements to build a requirements specification	Kajko-Mattsson & Nyfjord, 2009; Salah et al., 2014a, 2014b; Sibghatullah et al., 2006
Team (4 articles)	Form an initial product backlog	Ebert, 2006; Hildenbrand & Meyer, 2012; Qumer & Henderson-Sellers, 2007; Vanhanen et al., 2003
Team (4 articles)	Use iterations and feedback loops	Honious & Clark, 2006; Inayat et al., 2015; Nyfjord & Kajko-Mattsson, 2008; Sohaib & Khan, 2010
Team (2 articles)	Build an idea pool	Khurana & Rosenthal, 1998; Vanhanen et al., 2003
Team (2 articles)	Separate roles and responsibilities	Honious & Clark, 2006; Williams & Ferguson, 2007

Activities and Deliverables

The study identifies nine articles that present activities and deliverables. Both are needed for the creation of a PDD (process-deliverable diagram). PDDs document the process and its activities on the left side, the corresponding deliverables on the right side. These nine articles identify 20 activities and 30 deliverables. Depending on the article's depth, a richer description in the form of sub-activities and attributes is provided (e.g. Cloyd, 2001; Kajko-Mattsson & Nyfjord, 2009). The full list of articles with their corresponding activities and deliverables are presented in Table 26.

Table 26 – An Overview of the Identified Activities and Deliverables.

Authors	Activities	Deliverables
Vanhanen et al., 2003	Maintain Product Vision, Conduct Feasibility Study	Idea Pool, Product Backlog
Kajko-Mattsson and Nyfjord, 2009	Product Vision Planning, Product Roadmap and Release Planning	Product Vision, Product Vision Plan, High-Level Requirements Specification, Requirements List, Product Roadmap, Product Backlog
Sibghatullah et al., 2006	Product Vision Building, Approve Product Vision, Form Product Specification	Product Vision Statement, Formal Specification
Cloyd, 2001	User and User-Goal Analysis, Prototyping	User Profile, Requirements List, Paper Prototype, User Feedback
Khurana and Rosenthal, 1998	Vision Development, In-Depth Evaluation	Business Vision, Project Vision, Product Vision, Product Requirements
Salah et al., 2014a	Iteration 0	Product Requirements, Holistic Design Vision
Hildenbrand and Meyer, 2012	Envisioning, Stabilization	Design, Vision, Product Vision, User Stories, Product Backlog
D. Sy, 2007	Cycle Zero	Shared Vision, Design Principles, Design Goals, Personas
Menor, Tatikonda, and Sampson, 2002	Design, Analysis	Idea Concept, Project Order

Designing a Method for Software Product Discovery

Using the selected literature base, design requirements document the needs for the method design. The design requirements respond to the identified challenges (cf. Table 24). Moreover, the requirements form and identify generic statements about the artifact's design in the form of design principles (Gregor & Hevner, 2013; Peffers, Tuunanen, Rothenberger, & Chatterjee, 2007).

Product-Related Requirements. While ASD is an established method with a focus on the software's functionality, it is vague about the starting conditions. Examples are the desirable upfront activities that are not planned in development projects (Salah et al., 2014a). As a result, these steps often lack time and budget during their execution (Oliveira & Rozenfeld, 2010; Salah et al., 2014a, 2014b). Other appeals include calls for a clear position of exploratory activities in software development projects, which are insufficiently addressed by agile methods (Ferre & Medinilla, 2007). Also, the literature suggests less uncertainty and ignorance within pre-implementation phases of agile projects (Nyfjord & Kajko-Mattsson, 2008). Furthermore, scholars criticize the implementation of operational planning activities only in later development phases, and a lack of describing activities required for the creation of product visions or product backlogs (Heikkilä et al., 2015; Hildenbrand & Meyer, 2012). Hence, the study formulates the first Design Requirement (**DR1**): *A method for Software Product Discovery should articulate early activities that improve the software product developing environment.*

Idea generation is a key activity in a proposed pre-phase 0 (Khurana & Rosenthal, 1998), executed prior to the development of a new product, and prior to the identification of detailed customer needs, technological capabilities, or core product requirements (Stevens, 2014). Menor et al. (2002) describe a similar idea generating activity in the area of new service development. An initial ideation step as well as idea generation techniques are also used in new product development (Knoll & Horton, 2011; Sperry & Jetter, 2009). Therefore, the study formulates (**DR2**): *A method for Software Product Discovery should collect initial product ideas.*

Some sources mention that agile practices only start after a vision or more concrete artifacts are established (Hildenbrand & Meyer, 2012). According to Sarpong and Maclean (2012), a product vision can be defined as the "mental image of a yet-to-be-realized product" (p. 695). Others try to conceptualize ways to reach a useful vision. An extended envisioning process is suggested as a way to identify high-level requirements (Hollis & Maiden, 2013). More explicitly, the product vision serves as a key input element for all further activities (Vanhanen et al., 2003). Kajko-Mattsson and Nyfjord (2009) even describe a product vision planning phase in detail, aiming at identifying a so-called product vision plan. Sibghatullah et al. (2006) introduce the so-called product vision statement as a result of visioning activities, and state that the product vision becomes more important the higher the uncertainty or complexity of the product goal is. Using the term "high-level product scope", others propose the upfront activity of building an initial visionary scope and revising it through all following iterations (Inayat et al., 2015). In design thinking, the vision is frequently enriched in the ideate-phase (Hildenbrand & Meyer, 2012). A step towards a vision is the "holistic design vision" that results from an iteration zero (Salah et al., 2014a). Following an iteration zero, different simultaneous versions are possible, for example a business, a project, and a product vision (Khurana & Rosenthal, 1998). Kakar and Carver (2012) state that the creation of a clearly defined product concept is important in order to manage the software development process effectively. Tessarolo (2007) states that a product vision can be important for on-time performance. The process of product vision planning is described by Nyfjord and Kajko-Mattsson (2008). A developed vision can be used for sharing a unified picture, for example, with the developers (D. Sy, 2007). This in turn constitutes towards a unified understanding of the product in the project team (Jain & Suman, 2015). Ferreira et al. (2007) recommend that usability concerns should be a part of the vision, while Ebert (2006) sees translated market needs as input for the vision. In order to include this upfront visioning work in the product discovery phase, the study derives (**DR3**): *A method for Software Product Discovery should form a product vision.*

Looking at the results of the process, the literature suggests that agile development practices should pay more attention to usability concerns. Literature often states that either pragmatic and hedonic qualities do not play a role at all (Adikari et al., 2009; Brhel et al., 2015; Fox et al., 2008; Patton, 2002b; Sohaib & Khan,

2010), or only at a time much too late to have a major impact on the resulting product (Cloyd, 2001; Salvador et al., 2014). Thus, one goal should be to include a user-focus during the process. This is supported by the statement that UCD would be a perfect fit for an agile environment (Williams & Ferguson, 2007), and that requirements should be based on what users would be doing with the product. Hence, the study derives (**DR4**): *A method for Software Product Discovery should extract users' needs for pragmatic and hedonic qualities.*

To improve the software product development process and environment early activities have to be embedded and clearly articulated (DR1). A focus on the software product is broken down into elements that are more detailed by stating that initial product ideas should be collected in a first instance (DR2). Such ideas will be developed into and form a concrete product vision (DR3). Furthermore, the software's need to be usable and useful requires consideration (DR4). As these four design requirements all share the focus of the software product itself, they can be summarized to the first design principle (**DP1**): *A method of Software Product Discovery clarifies product context, goals, purposes and key requirements.*

User-Related Requirements. In ASD as well as UCD, users and customers play a larger role through frequent stakeholder involvement in comparison to traditional development or design techniques. Stakeholders are important in every development project. While there are many stakeholders available, the user is one of the most important ones. However, the user is often neglected throughout the creation process of a software product. For example, ASD does not provide concrete guidance on how to develop software that is user-friendly (Salah et al., 2014a). However, this poses difficulties during the development, especially in terms of software usability. Often, a reason for a lack of usability is the low prioritization of usability during the development. In addition, there is often no guidance on user experience activities (Kuusinen, 2014). Some point out that the lack of user-focus in agile practices is a key reason for a lack of innovation (Rejeb et al., 2008). In order to cope with this issue, different suggestions can be found in literature. One example is a condensed upfront user analysis (Cloyd, 2001; Miller, 2005). Therefore, the study derives (**DR5**): *A method for Software Product Discovery should improve the adoption of practices for researching users' needs.*

Eventually, user requirements have to be collected and evaluated (Liskin, 2015). Hildenbrand and Meyer (2012) mention that even mature processes, such as lean thinking do not provide descriptions on how to gain knowledge of user requirements. Sohaib and Khan (2010) ask how user requirements could be gained from the stakeholders usually involved in agile feedback rounds. In most cases, this is the customer rather than the end-user. D. Sy (2007) propose their version of a cycle zero, also incorporating user research. In order to process such requirements effectively, they need to be properly documented. The results identify

(**DR6**): *A method for Software Product Discovery should properly document and record user requirements as a basis for further design and development activities.*

Building on the two prior design requirements, practices and documentation need to be integrated into the ongoing developments (Loniewski, Armesto, & Insfran, 2011). Even without starting actual programming activities, it is still necessary to prepare for a later phase. It ensures the ability to integrate collected requirements. For cycle zero, D. Sy (2007) proposes the detailed inquiry of collecting data in order to support later phases. For example, the provision of exact target user descriptions. The process following the design thinking principles can help to include the users' wishes into the finished product (Hildenbrand & Meyer, 2012). Other agile techniques, such as user-goal-analysis followed by prototyping activities try to improve this process (Cloyd, 2001; Qumer & Henderson-Sellers, 2007). Following, the study suggests (**DR7**): *A method for Software Product Discovery should enable the integration of user requirements and user-centered practices into further design and development activities.*

The need to improve the adoption of user research practices (DR5) and the proper documentations of their results (DR6) are key tools of a user-focus during discovery. However, these also need to be integrated successfully into further design and development activities (DR7). As these requirements are user related and aim to include the user into the design and development activities, they are summarized into **DP2**: *A method of Software Product Discovery researches and integrates specific users' needs into the software product design and development process*.

Team-Related Requirements. The study suggests shortcomings of the overall understanding or mental image of the product idea by the design and development team. For example, da Silva et al. (2011) point out that the “big picture” of what is expected is gone missing at some point in time throughout the project. This issue can be attributed to the fact that either there is not enough detail of the concepts to begin with, or the formulated product vision poses inconsistencies (Kajko-Mattsson & Nyfjord, 2009; Tessarolo, 2007; Vanhanen et al., 2003). Countermeasures propose the constitution of a shared vision amongst all project members (Frishammar et al., 2011; Gaubinger & Rabl, 2014; Salah et al., 2014a, 2014b; D. Sy, 2007). Hence, the study suggests (**DR8**): *A method for Software Product Discovery should enable the product team to develop a unified understanding of the product.*

Others focus on and requirements engineering, and how these processes lack creative thinking and a dedicated focus when implemented today (Adikari et al., 2009; Hollis & Maiden, 2013). In current development projects, a lack of guidance can be observed that leads to either bad quality or longer development times (Sibghatullah et al., 2006). The combination of creative thinking and structured stepwise progress is grounded in the fundamental difference between design and development. While the design

emphasizes the creative part that can be hard to articulate and document, development stemming from the engineering disciplines builds on stepwise and sequential process improvements. A design and development team needs to master both. Hence, the study derives (**DR9**): *A method for Software Product Discovery should enable the product team to master both, development maturity and creative thinking.*

The lack of upfront activities as mentioned in the literature (e.g. Salah et al., 2014a, 2014b) roots back to a lack of management support and appreciation (Ferre & Medinilla, 2007). Consequently, a lack of other elements, such as maintenance, is found in the literature (Kajko-Mattsson & Nyfjord, 2009). In describing general problems in ASD, Hollis and Maiden (2013) state that the principle of simplicity found throughout agile processes has been taken too far in order to still provide any contribution towards innovation. On the contrary, Ebert (2006) finds fault with agile cycle times being too long to still be productive. In addition, the coordination in agile teams between the different functions leads to project delays (Ebert, 2006). Both, cycle time and project coordination are common management decisions. From a different perspective, Gamble and Hale (2013) describe that scalability in ASD projects is difficult to achieve. In addition, UCD is prone to management challenges. For example, Salah et al. (2014a) find a lack of support by management roles towards user-centered activities, making it difficult for the team to execute them. Summarizing, the study formulates (**DR10**): *A method for Software Product Discovery should assure management support for the product team.*

Furthermore, the need to clearly separate product discovery from product creation has been stressed in the literature (Brhel et al., 2015). Such separation helps to clearly separate roles and responsibilities. For example, within UCD it is important to distinguish between the researcher and the prototyper (Williams & Ferguson, 2007). Within ASD, the importance of clearly upfront specified roles and responsibilities have been suggested (Jain & Suman, 2015). Scrum, one example method of ASD, clearly articulates the roles and responsibilities of team members. Therefore, the study suggests (**DR11**): *A method for Software Product Discovery should clearly distinguish different roles and their responsibilities for the product team.*

Gaining a unified understanding of the product in planning (DR9), providing thorough guidance for designers and developers (DR10), ensuring management support (DR11) as well was specifying and separating team roles and responsibilities (DR12) are design requirements focusing on the team, i.e. all human resources involved. These team requirements are summarized in the last design principle **DP3**: *A method of Software Product Discovery develop a unified understanding of the product and its importance within a diverse team of specialists*.

An overview of all requirements and their principles is presented in Table 27. The number of articles used within each dimension is: 30 (68%) articles with product related requirements, 11 (25%) articles with user related requirements, and 19 (43%) articles with team related requirements.

Table 27 – Overview of Design Requirements and Sources.

Dimension	Requirement	Source
Product	DR1 - articulate early activities	Ferre & Medinilla, 2007; Heikkilä et al., 2015; Hildenbrand & Meyer, 2012; Nyfjord & Kajko-Mattsson, 2008; Oliveira & Rozenfeld, 2010; Salah et al., 2014a, 2014b
	DR2 - collect initial product ideas	Khurana & Rosenthal, 1998; Knoll & Horton, 2011; Menor, Tatikonda, & Sampson, 2002; Sperry & Jetter, 2009; Stevens, 2014
	DR3 - form a product vision	Hildenbrand & Meyer, 2012; Hollis & Maiden, 2013; Inayat et al., 2015; Jain & Suman, 2015; Kajko-Mattsson & Nyfjord, 2009; Kakar & Carver, 2012; Khurana & Rosenthal, 1998; Nyfjord & Kajko-Mattsson, 2008; Salah et al., 2014a; Sarpong & Maclean, 2012; Sibghatullah et al., 2006; D. Sy, 2007; Tessarolo, 2007; Vanhanen et al., 2003
	DR4 - extract users' needs	Adikari et al., 2009; Barksdale & McCrickard, 2012; Brhel et al., 2015; Cloyd, 2001; Fox et al., 2008; Patton, 2002; Salah et al., 2014a, 2014b; Salvador et al., 2014; Sohaib & Khan, 2010
User	DR5 - adopt user research practice	Cloyd, 2001; Kuusinen, 2014; Miller, 2005; Rejeb et al., 2008; Salah et al., 2014a
	DR6 - document and record user requirements	Hildenbrand & Meyer, 2012; Liskin, 2015; Sohaib & Khan, 2010; D. Sy, 2007
	DR7 – integrate user requirements and user-centered practices into design and development activities	Cloyd, 2001; Hildenbrand & Meyer, 2012; Loniewski et al., 2011; Qumer & Henderson-Sellers, 2007; D. Sy, 2007
Team	DR8 - develop a unified understanding of the product	Frishammar et al., 2011; Gaubinger & Rabl, 2014; Kajko-Mattsson & Nyfjord, 2009; Salah et al., 2014a, 2014b; Silva da Silva et al., 2011; D. Sy, 2007; Tessarolo, 2007; Vanhanen et al., 2003
	DR9 - master development maturity and creative thinking	Adikari et al., 2009; Hollis & Maiden, 2013; Sibghatullah et al., 2006
	DR10 - assure management support	Ebert, 2006; Ferre & Medinilla, 2007; Gamble & Hale, 2013; Hollis & Maiden, 2013; Honious & Clark, 2006; Kajko-Mattsson & Nyfjord, 2009; Salah et al., 2014a, 2014b
	DR11 - distinguish different roles and responsibilities	Brhel et al., 2015; Jain & Suman, 2015; Williams & Ferguson, 2007

The PDISC Method

The study relied for the method construction on the method database. The database includes all method fragments from nine articles. An overview of these articles and their respective method fragments is presented in Table 28. Following the extraction of the methods and method fragments from prior literature, a method for product discovery is built (see Figure 15). The expert evaluation led to improvements of the method along its activities, deliverables, and overall structure. Based on the expert interviews, the activities are adjusted by adding a team-building phase in the beginning of the method (I-05, I-06, I-08). While all

projects require an initial idea, the definition of rules, resources, and commitments is a mandatory step before the idea generation can begin. Related to the deliverables, backlog elements that form the product backlog are added (I-04), analogue to the vision elements that form the product vision. Related to the structure, optional iterations for the initial idea generation and the main activities are included (I-01, I02).

The method starts with the team-building phase. This phase includes initial strategic decisions by defining ground rules for the team, organizing resource, and assuring commitment (I-05, I-06, I-08). Resources include human resources, technical resources, and workplaces. In some organizations, such commitment happens through an annual process and hence, needs to be planned. Following the team building, the method continues with an idea generation activity (cf. Khurana & Rosenthal, 1998; Menor et al., 2002; Vanhanen et al., 2003), where each stakeholder can make suggestions in order to develop an idea pool. Interviews or focus groups help to gather ideas early on. If needed, this phase can be iterated to collect more ideas.

Table 28 – Overview of Identified Methods and Focus.

Dimensions	**Activities**			**Deliverables**		
Article	**Product**	**User**	**Team**	**Product**	**User**	**Team**
Vanhanen et al., 2003	X		X	X		
Kajko-Mattsson & Nyfjord, 2009	X			X		
Sibghatullah et al., 2006	X			X		
Cloyd, 2001		X		X	X	
Khurana & Rosenthal, 1998	X		X	X		X
Salah et al., 2014a		X		X		
Hildenbrand & Meyer, 2012	X	X		X	X	
D. Sy, 2007		X			X	
Menor, Tatikonda, & Sampson, 2002	X			X		

Thereafter, an iterative process allows the execution of activities multiple times. Three main activities follow and can be executed in parallel. First, a central product vision document is created, so that a common understanding is developed and documented for management's approval (Kajko-Mattsson & Nyfjord, 2009; Sibghatullah et al., 2006; D. Sy, 2007; Vanhanen et al., 2003). Management and the development team use prioritization techniques to select requirements and convert them into the product backlog. Second, users are engaged and integrated into the discovery process (Cloyd, 2001; Hildenbrand & Meyer, 2012; Khurana & Rosenthal, 1998; Salah et al., 2014a; Sibghatullah et al., 2006; D. Sy, 2007). Marketers or developers use general or contextual interviews, task analysis, or other practices to engage the user and collect requirement. The results are documented, either as an interview protocol or as a backlog element that forms the product

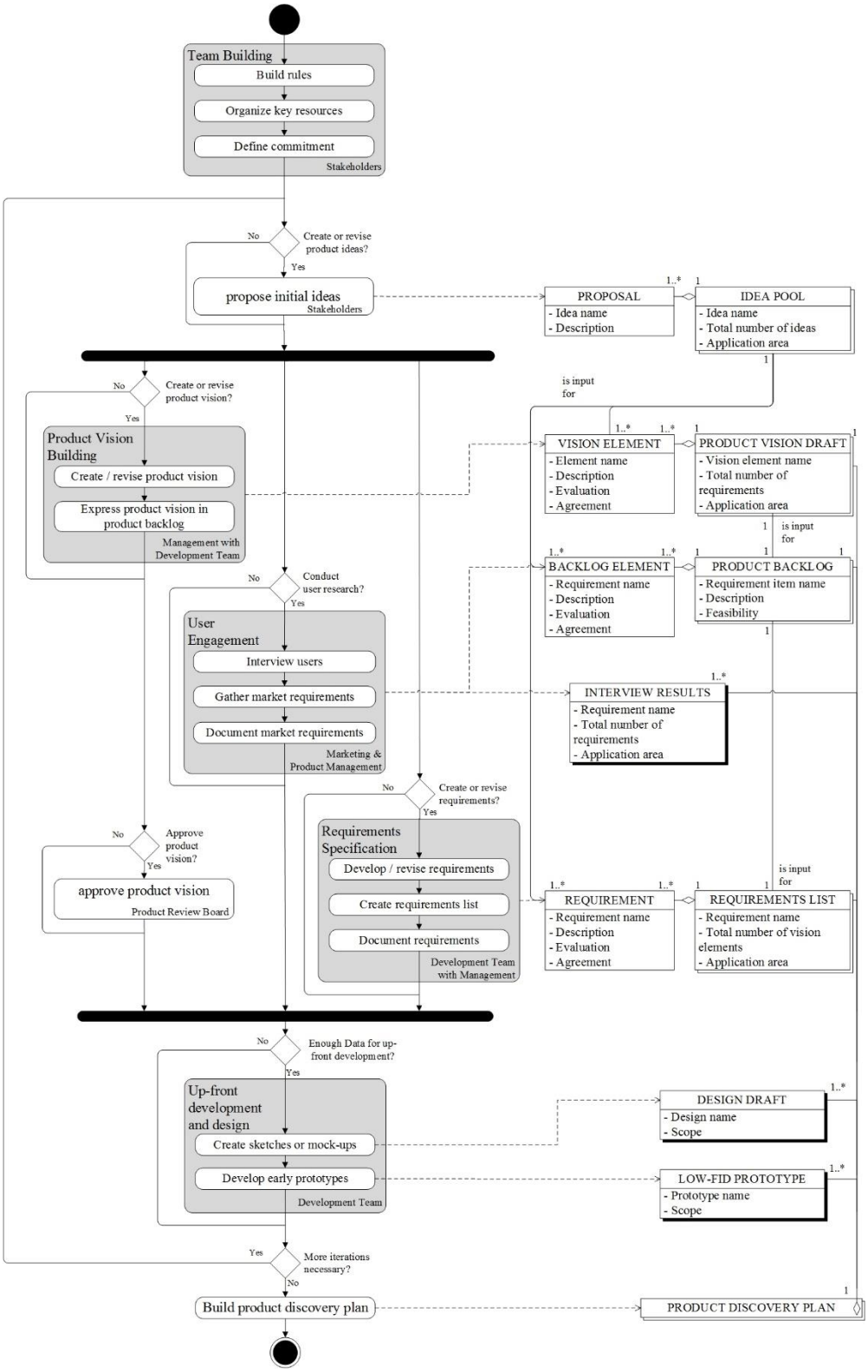

Figure 15 – The Diagram for the Software Product Discovery Method.

backlog. Third, requirements are generated. The requirements are based on the idea pool within the first iteration (Hildenbrand & Meyer, 2012; Kajko-Mattsson & Nyfjord, 2009; Khurana & Rosenthal, 1998; Salah et al., 2014a; Sibghatullah et al., 2006). In later iterations, such requirements are refined using the user engagement results. The documentation of user stories or scenarios can help the development team to document and communicate such requirements with management.

The product vision draft and the requirements list form the product backlog (Kajko-Mattsson & Nyfjord, 2009; Vanhanen et al., 2003). Until this point, the product backlog is the central element combining different activities and serving as input for any later phases. While this rather technical pool of functionality provides value to developers, other roles might need different types of documentation. The iterative cycle of the main activities ends when sufficient information for an initial design draft or low-fidelity prototype is available (Hildenbrand & Meyer, 2012; Salah et al., 2014a; Sibghatullah et al., 2006). Design drafts are generated using sketching or mock-up applications. Early prototypes are created using paper. Later, tools can be used to create digital and interactive prototypes. However, the upfront development and design activities can only start after the collection of user inputs and requirements. However, once a design has been proposed, feedback can be acquired. If needed, a jump to the beginning of the iterative cycle is possible. Last, the product discovery plan needs to be build based on the product vision draft, product backlog, requirements list, design drafts, and low-fi prototypes.

3.2.5. Discussion

The study proposes PDISC, a new method for the discovery of software products. The method structures early activities of product development and their corresponding deliverables. Within existing approaches to software development and design, those early activities are either missing or receive little attention (Fox et al., 2008).

The method for software product discovery suggests important activities and deliverables that collectively present the product discovery phase (Brhel et al., 2015). The suggested activities and deliverables assist teams in identifying relevant work packages with clear results. The activities stem from different areas and assure: i) the inclusion and integration of users (Chamberlain et al., 2006; T. S. da Silva, Silveira, et al., 2011; D. Sy, 2007), ii) a response to business related concerns, such as the reduction of uncertainty (Khurana & Rosenthal, 1998; Sperry & Jetter, 2009; Stevens, 2014), and iii) the feasibility of the solution, e.g. by developing a joint vision across team members (Frishammar et al., 2011; Gaubinger & Rabl, 2014).

Furthermore, the study derives a meta-model that presents sequences and iterations between activities, and depicts dependencies between deliverables (Larusdottir et al., 2017). Despite this aggregated view, the method description uses common terminology to simplify its adoption by method users (Henderson-Sellers

et al., 2014). The sequences and iterations also establish time constraints and describe a sequence of events to reach a certain outcome (Brinkkemper et al., 1999). In addition, interim deliverables suggest anteceding artifacts before the team is able to create the final deliverable.

The study articulates design knowledge by identifying three design principles that improve the discovery of software products (Gregor & Hevner, 2013). First, the team needs to identify a viable solution that provides sustainable business value. Without a viable solution, the product can become a cost factor or might be dismissed after further evaluation of its lacking long-term business value (R. G. Cooper, 2011). Second, the user needs to be involved, so that the team is able to suggest a desirable solution. More often than not, this is a central challenge to development and design teams (Kuusinen, 2014; Sohaib & Khan, 2010). However, the correct involvement and integration of users and other stakeholders enhances the success of the project (Barksdale & McCrickard, 2012; Inayat et al., 2015). Third, the team needs to be capable of developing and delivering the envisioned solution. Different constraints limit the feasibility of a solution and hence, increase the project's risk through inconsistencies or imprecisions (Tessarolo, 2007; Vanhanen et al., 2003). All principles are derived from supporting evidences within the literature.

Threats to validity

The study follows general recommendations for the planning and design of a research project (Verschuren, Doorewaard, & Mellion, 2010). More specifically, the study follows established guidelines for conducting a SLR (Kitchenham & Charters, 2007). Contrary to some trends, the study is rather inclusive and identifies relevant conference publications (Boell & Cecez-Kecmanovic, 2015). Following the main data collection, experts evaluate the results in interviews. The interviews start with an appreciative approach (benefiting from practitioners' experiences) and end with a laddering approach (in-depth understanding of suggested changes) (Schultze & Avital, 2011).

The researchers introduce the main threat to the internal validity. In order to avoid this threat, the study presents a systematic literature review that limits the influence from the researcher (Kitchenham & Charters, 2007). In addition, nine different methods are analyzed to assure the correct interpretation of temporal dependencies. All methods are modeled using the same language. Thus, the study enhances the comparability and ease of integration of the methods found in the literature. Later, the evaluation enhances the method by including the experience and in-depth knowledge from the experts.

More recently, the integration of ASD and UCD gained further momentum. Most of the literature reiterates the identified challenges. For instance, integrating users in development activities (Kuusinen, 2015; Schön, Thomaschewski, & Escalona, 2017), identifying proposals or elements that form product vision (Kristinsdottir, Larusdottir, & Cajander, 2016), remain challenges. Yet, the challenges related to roles

and responsibilities have been recently suggested to be solved by integrating UX-specialists into the development team and therefore, forming cross-functional product teams (Kuusinen, 2016).

Threats to *external validity* minimize the generalizability of the results. The main data collection is a secondary study and therefore, benefits from identifying meta-information that are more reliable compared to a single study. The study introduces quality criteria that remove low quality publication of the list of final articles (Kitchenham & Charters, 2007). While the focus is on software with relevant literature sources from ASD and UCD benefit from adjacent fields are integrated by including relevant terms from new product development and innovation management research. The evaluation by experts further strengthens the generalizability of the results. Here, the study benefits from the experts' wide expertise and experience across different projects and industries.

The *reliability* is threatened when other scholars conduct the same study, but come to different conclusions. The study avoids threats to reliability by having a research team involved in this project. In addition, the study provides high transparency when presenting the research method and results. While important terms are defined, the study also benefits from the modelling language that prescribes an activity table and concept table. The activity table lists all activities used within a PDD and presents a corresponding definition. Likewise, the concept table presents all identified deliverables and their corresponding definition.

3.2.6. Conclusion

The study develops a software product discovery method. For the development of the method, the study starts by extracting recommendations and challenges from existing literature. Thereafter, starting with eleven requirements, the study derives three design principles that guide the method's design. In order to evaluate the method, expert interviews are conducted.

The study presents a new method for the discovery of software products. Therefore, relevant activities and deliverables are identified. The study models sequences and dependencies between the method's activities and deliverables. The documented PDD highlights the product discovery plan as the final deliverable.

The study contributes design knowledge by systematically deriving design principles. Each design principle stems from three to four design requirements. The design principles integrate a central requirement for the future product, i.e. its viability, desirability, and feasibility. The product-related requirements help the product team to design a viable product. Implementing user-related requirements into the method assures that the vision proposes a desirable product. In addition, team-related requirements of the method suggest the design of a feasible product.

The design principles allow practitioners to challenge their own processes for comprehensiveness and completeness. While they may not implement all steps, depending on the size of their organization, the method helps practitioners to design a product that is viable, feasible, and desirable. If a product falls short on any of these three dimensions, the product's success is at risk. Furthermore, the implementation and individual activities provide a stepwise tutorial for creating a product vision. This is especially valuable for those organizations that have to define their software product discovery process yet. In further work, future studies can investigate the adoption and application of the method in exemplary case companies. Different instantiations are interesting facets for future work. Following the development of the meta-model, the investigation of selecting different techniques can yield further contributions. Furthermore, the investigation of possible situational factors depending on organizational context can help to improve the discovery of software products.

4. Human Factors of Agile Software Development

Human factors are suggested to be an important, yet under researched area in ASD research (Dybå & Dingsøyr, 2008). Given the importance of teams within software development, this chapter focuses on human factors related to the software development team (Sawyer, 2004). These factors are applicable to three psychological perspectives, team behavior, team affect, and team cognition (S. Kozlowski & Ilgen, 2006). In this chapter, the first study investigates the behavioral perspective of teams, drawing on CAS theory. The second study investigates the effects of emotional contagion in SDT.

4.1. Explaining the Emergence of Team Agility: A Complex Adaptive Systems Perspective [7]

4.1.1. Introduction

Software development organizations need to respond to manifold challenges that include new customer requirements, market dynamics, mergers, and technological innovation (Börjesson & Mathiassen, 2005). Hence, software development organizations need to improve their reaction to changes, such as changing customer requirements. As a result, many organizations are increasingly adopting agile software development (ASD) as their development method (Serrador & Pinto, 2015; West & Grant, 2010). However, simply adopting agile methods, such as Scrum or XP, will not automatically lead to an agile organization (Conboy, 2009; Gregory et al., 2016). A corresponding example is the development organization for Visual Studio Online in Microsoft, which took four years to move from a waterfall-oriented organization to becoming a truly agile organization that releases new features into the cloud-based product in a three-week cycle[8]. Thus, there are different states of agility, but as of yet we lack a proper understanding of how they emerge. This study sheds light on this dilemma by identifying different dynamics that explain different emergent states of agility within software development teams.

Within the fields of software engineering and information systems development, there are differing views on the concept of agility. While many choose a methodical focus (Campanelli & Parreiras, 2015; Conboy, 2009), others focus on the corresponding practices (Diebold & Dahlem, 2014; Inayat et al., 2015) or adopt a dependency perspective in relation to knowledge, process, and resources (Strode, 2015). Given the complexity of software development (Meso & Jain, 2006), however, it is often carried out in teams (Sawyer, 2004). Yet, only a few studies focus on agility as a team phenomenon (e.g. Lee and Xia, 2010), and scholars have called for more research related to human or social factors (Dingsøyr et al., 2012; Dybå & Dingsøyr, 2008).

[7] This section is based on Werder and Maedche (2017b).

[8] http://thenewstack.io/visual-studio-online-microsofts-road-open-agile-development/

While previous research on agility often compared ASD with traditional or waterfall methods (Dybå & Dingsøyr, 2008), more recently there has been a stronger investigation into more mature agile teams (Dingsøyr et al., 2012; F. S. Silva et al., 2015). Many teams have already adopted ASD focusing on the method, and institutionalized methods such as Scrum help organizations to get a start. However, the adoption of a specific method (e.g. Scrum) cannot explain the difference between mature and immature agile teams (Gill et al., 2016). Rather, prior work suggests concepts independent of the method and more profoundly embedded in the team (Campanelli & Parreiras, 2015). The study suggests that team agility is an emergent phenomenon that develops and that evolves over time (Goldstein, 2000; S. W. J. Kozlowski & Chao, 2012).

When investigating emergence (i.e. the result of the process self-organization), scholars often rely on the theory of complex adaptive systems (CAS) (Alaa & Fitzgerald, 2013; Mittal, 2013). Non-linearity, emergence, and self-organization are major characteristics of CAS. Non-linearity refers to the relationship between the system components and the whole. When the relationship is non-linear, a small change in a component can lead to a larger change in the whole (McCarthy, Tsinopoulos, Allen, & Rose-Anderssen, 2006). The concepts of self-organization and emergence are often discussed together. While self-organization is described as a process, emergence is the result of such a process (Curşeu, 2006; McCarthy et al., 2006). Both characteristics define the trajectory of the CAS and therefore require further investigation. We need to understand the conditions leading to different emergent states in order to channel them accordingly (Goldstein, 2000; S. W. J. Kozlowski & Chao, 2012). Hence, the study investigates the broader picture that considers local, global, and contextual dynamics and their relationship to an emergent phenomenon (Mittal, 2013).

This work seeks to identify the *conditions of team dynamics that explain emergent states of team agility.* To this end, the research uses a multi-level perspective of CAS and considers team agility as an emergent state. The overall objective is to propose a framework that explains the empirical findings. Thus, the research objectives are: i) to provide an integrated summary of the literature of complex adaptive systems and agility, ii) to derive propositions and an initial framework from the integrated literature, and iii) to investigate the proposed framework empirically.

In order to identify the conditions of team dynamics, a multicase study is conducted, whose results will help practitioners to enhance their agility (Gregory et al., 2016). The identification of specific conditions guides practitioners along the evolution of their agility. The theoretical contribution is tripartite: First, while earlier research in the field of ASD focused on the method when using CAS as a theoretical lens (Alaa & Fitzgerald, 2013; Kautz, 2012; Meso & Jain, 2006), this work focuses on the SDT as the unit of analysis (cf. Moe, Dingsøyr, & Dybå, 2010; Sawyer, 2004). The focus on teams heeds the call for more research in the area of

social and human factors in agile software development (Dybå & Dingsøyr, 2008). Second, the study builds on a wider body of knowledge from team-focused research (Arrow, McGrath, & Berdahl, 2000; S. Kozlowski & Ilgen, 2006; Mathieu et al., 2008). Therefore, the study responds to calls to connect ASD research with existing streams from mature fields (Dingsøyr et al., 2008; Dybå & Dingsøyr, 2008). The study leverages the concepts of self-organization and emergence as key characteristics of CAS. CAS is an established theory explaining team-related phenomena (Arrow et al., 2000; Curşeu, 2006; S. W. J. Kozlowski & Chao, 2012; Mathieu et al., 2008). Third, the need for more research into mature agile teams is addressed by comparing them with less mature agile teams to better understand their differences (Dingsøyr et al., 2008). The definitions of team agility and emergent states are reviewed in order to suggest team agility as an emergent phenomenon. Initial conceptualizations to identify SDTs as CAS are conceptual contributions without empirical validation (e.g. Alaa and Fitzgerald, 2013; Kautz, 2012; Meso & Jain, 2006). Hence, there are differences between teams that recently adopted ASD and those that are more mature. This study seeks to understand the factors that enable self-organization within the team.

Following this introduction, the theoretical background presents ASD, CASs, and teams in subsection 2. Subsection 3 presents the conceptual development, which leads to four propositions and the proposal of a team agility framework. The research method in subsection 4 includes the sampling strategy, data collection, case description, and case analysis. In subsection 5, the empirical data is used to test the framework and report the findings on the local, global, and contextual systems. Subsection 6 discusses the findings. Finally, subsection 7 concludes the paper and points toward future research.

4.1.2. Theoretical Background

Complex Adaptive Systems

The theory of CAS stems from the idea that some systems are challenging to simulate (Holland, 1992). This idea was formalized in general system theory (e.g. Boulding, 1956) and extended with the notion of complexity. A CAS has three characteristics, i.e. evolution, aggregate behavior, and anticipation, and the evolving nature of such systems provides a key challenge to research (Holland, 1992). Although academics have applied the theory of CAS in different fields, such as control theory, economics, biological cells, and games, the systems share four distinct characteristics across disciplines (Holland, 2006). First, parallelism means that the agents work in parallel while also interacting with each other. Second, the fact that an individual agent's response depends on the communication received indicates the characteristic of conditional action. Third, agents represent modularity in the sense that the formulation of a response is subject to multiple decision points within the agent at which the agents assess the situation. Fourth, agents adapt and evolve over time as part of an ongoing learning process. The behavior of a CAS is attributed to the simultaneous and parallel actions of the system's agents (Choi, Dooley, & Rungtusanatham, 2001).

These dynamics can be attributed to different sub-systems and lead to new emergence of the system (Goldstein, 1999).

When discussing teams as a CAS, the latter's agents are the team members. The literature presents three relevant sub-systems with their own dynamics (Arrow et al., 2000). Here, the team consists of multiple systems, for example, the individual team members, the team itself, and larger systems such as the organization or community (Mathieu et al., 2008; McGrath, Arrow, & Berdahl, 2000). Consequently, the literature introduced local, global, and contextual dynamics that represent changes shaping the systems. While the local dynamics represent individual activities such as tasks and using resources or technology (McGrath et al., 2000), the global dynamics include the team's behavioral variables. The teams may respond differently to various team compositions, task designs, team norms, or compelling direction, i.e. a specification of the team's purpose (Wageman et al., 2005). The contextual dynamics influence the team's direction but are outside its immediate scope. Examples in this respect are management support, corporate incentives, and talent supply (McGrath et al., 2000).

Research into teams' effectiveness argues for the impact of CAS on their emergent states, such as team cognition or cohesion (Curşeu, 2006). More recently, seven concepts for a CAS in the context of information systems development were identified and suggest that these lead to emergent effects (Kautz, 2011). The theory of CAS has been applied to the context of ASD. In a conceptual work, scholars identify principles of CAS that they support with the principles and practices of ASD (Alaa & Fitzgerald, 2013). As a result, they suggest a contextualized conceptual framework of complex adaptive systems. The work highlights emergent attributes that require further exploration in ASD, e.g. diversity, pattern recognition, adaptation as a fit to environment, collaboration, and inter-connectivity.

While the conceptualization of CAS includes three sub-systems with three different dynamics affecting the group's shaping over time, we need to understand the characteristics that explain the system behavior of a CAS. Self-organization and emergence are two important characteristics of CAS. They are often jointly discussed, and there is an ongoing debate about their relationship. While some authors understand self-organization as an emergent phenomenon (Curşeu, 2006; Mittal, 2013), others advance the idea that emergence is the result of the process self-organization (Curşeu, 2006; McCarthy et al., 2006). A third characteristic of CAS is the aspect of non-linearity, i.e. the unpredictability of the whole in relation to a change of its parts. For example, a small change in a system component can lead to a significant change in the system as a whole, and vice versa. Non-linearity is a requirement for self-organization and hence for the development of novel or emergent outcomes (Goldstein, 1999). The following discusses the concepts of self-organization and emergence in greater depth.

Self-Organization

Self-organization is defined as "a process in a complex system whereby new emergent structures, patterns, and properties arise without being externally imposed on the system" (Goldstein, 1994). Here, it becomes apparent why the two concepts of self-organization and emergence are often discussed together. The definition suggests the authors' view of self-organization to be a process and therefore of emergence as the product. Self-organization can often be observed in nature, for example in the formation of flocking birds (Choi et al., 2001). Contrary to what some believe, mocking birds do not implement a predetermined plan or follow a lead bird when flying in formation. Rather, it is the result of self-organization whereby each bird derives its flying position from local information. Therefore, the resulting formation is a pattern or structure of emergence that can be observed. A pre-condition of self-organization is the system agent's autonomy (Maturana & Varela, 1980; Vidgen & Wang, 2009). Autonomy is "the extent to which a team [or individual] has considerable discretion and freedom in deciding how to carry out tasks" (Langfred, 2005, p. 514). Without autonomy, the agent is dependent on external stimuli and guidance for a sense of direction. An external influence would prohibit the emergence of new structures and patterns through a process of self-organization. As a result, some may suggest that self-organization only comes about through a lack of control. In fact, the opposite is the case: It is the result of local dynamics and the adaptation of agents that build a new configuration of the system (McCarthy et al., 2006). Within a new configuration, the connectivity needs to be moderate. While the highly connected system agents follow a structure that is too rigid, loosely coupled system agents lack structure and lead to recursive patterns (McCarthy et al., 2006).

Within teams, the development of a hierarchy in the form of emergent leadership is an example for one form of increasing order (Goldstein, 1999). Self-organization can only occur when the system has more energy on the inside than the pressure external forces exert on it (Anderson, 1999). Hence, SD team members have to be motivated and enthusiastic about their work, which helps them to cope with negative effects for example caused by uncertainty and stress. Larger teams have more energy, as they have more team members to counter negative effects. Moreover, the team members' characteristics and their decision process require consideration as both influence the energy within the system. In contrast to regular teams that only execute the team task, self-organizing teams also monitor and manage their development process and progress (Hackman, 2002).

A self-organized system is a "system that starts with its parts separate (so that the behavior of each is independent of the others' states) and whose parts then act so that they change towards forming connections of some type" (Ashby, 1962, p. 266). Independent behavior of the system's agents describes a common practice for teams: Professionals with the ability to act on their own are drawn together to follow a common

goal. Often, these team members have complementary skill sets that help them to adapt and form connections with the goal of achieving such a goal.

Emergence

The close relationship between self-organization and emergence is mirrored in the definition of emergence. Emergence is "the arising of new, unexpected structures, patterns, properties, or processes in a self-organizing system" (Goldstein, 1994). A more granular definition is offered by McCarthy et al. (2006), who define emergence as "the manifestation of new process characteristics due to the collective behavior of the agents, as opposed to the individual behavior of each agent (Anderson, 1999; Holland, 1995; Kauffman, 1995; Waldrop, 1992)" (p. 444). While the former definition provides a broader view, the latter limits it to a new process characteristic. In either case, the definition describes the rise of something new because of collective behavior within a system. Hence, emergence offers a way to experiment with and explore a new configuration and therefore makes it possible to capture a configuration that is different from the previous one. Only through this manifestation is it possible to investigate and determine the outcome and use it as a feedback mechanism. If needed, proper adaptations can lead to a more desirable configuration. Over time, such adaptations with interim states help the team to evolve, which results in an emergent capability (McCarthy et al., 2006). Such an emergent capability helps the team to navigate in the quickly changing software development environment, where the challenging environment, in the form of technological innovation and new customer requirements, pressures the team to change (Börjesson & Mathiassen, 2005). The team is required to act and react quickly in such an uncertain environment. The uncertainty theory describes decision making under uncertain conditions (e.g. Gilboa, 2009).

Within teams, emergence is a phenomenon that "is a pattern of behavior, a coherent structure or a state between individuals" (Curşeu, 2006, p. 251). Often, team-related literature refers to the term "emergent state," which is applied to many theoretical frameworks (e.g. Ilgen et al., 2005; Kozlowski & Ilgen, 2006). Emergent states can manifest themselves in different forms within a team, such as team trust, team cognition, or team affect. Emergent states are defined as "constructs that characterize properties of a team that are typically dynamic in nature and vary as a function of team context, input, processes and outcomes" (Marks et al., 2001, p. 357). While some may suggest emergence to be a positive phenomenon, it may also have negative effects. Shitstorms in social media are an example of an emergent phenomenon with dominantly negative effects. Most people would agree that such an emergence has led to a worse state of affairs. The same applies to the concept of emergent states. The adaptation to a new emergent state can lead to a negative result. The results of an adaptation can be lower team trust if a team member shares misleading information.

Emergent states within a team that is conceptualized as a CAS reside in the team and are influenced by local, global, and contextual dynamics (Ilgen et al., 2005; S. Kozlowski & Ilgen, 2006). While an emergent state is the outcome of self-organization, within team theoretical frameworks they are one instantiation of a team process. Self-organization and emergence are also facets of adaptability (McCarthy et al., 2006). Moving through space-time, the CAS is adapting itself and its behavior to existing conditions. Hence, this adaptation is one of a system's capabilities. However, the process of adaptation through which the system seeks to improve its fit to the given conditions is self-organization.

Team Agility

The agile manifesto is the foundation of ASD (Beck et al., 2001). The manifesto provides a set of principles that guide software developers and SDTs in adopting agile methods. ASD has many advantages over traditional software development (cf. Dybå & Dingsøyr, 2008). One of the main benefits is the embracement of customer changes and therefore an increase in the customer relationship in comparison with a document-driven approach (Sillitti, Ceschi, Russo, & Succi, 2005). In a similar vein, ASD prevents negative effects for customers by increasing understanding and communication between them and the developers (Ceschi, Sillitti, Succi, & Panfilis, 2005). Moreover, environmental factors, such as new customer requirements, market dynamics, mergers, and technological innovation (Börjesson & Mathiassen, 2005), drive the software development organizations to adopt an agile method. The need to deliver the products in shorter timeframes, while providing additional value, also leads to the adoption of such development methods (Baskerville, Ramesh, & Levine, 2003). In order to cope with the increased need to deliver software rapidly, specific practices, such as parallel development, increased release frequency and customer involvement have been suggested (Baskerville et al., 2003). Besides external factors influencing the adoption of ASD, developers positively perceive the adoption of agile practices, resulting in the recommendation of its adoption (Melnik & Maurer, 2002, 2005). Depending on the focus of the agile adoption, different maturity levels are derived (Gill et al., 2016).

In order to enhance the understanding and measurement of ASD, literature often refers to the concept of agility. Such a concept is also presented in other reference disciplines, such as business research and supply chain management (e.g. Hoek et al., 2001). In the field of manufacturing research, a reference agility model has been proposed (Sharifi & Zhang, 1999). The agility model suggests agility drivers lead to agility by forming agile capabilities such as responsiveness, competency, flexibility, and speed. Given that flexibility is a key aspect of agility, the work by Volberda (1996) is an important source of research into organizational agility. He defines organizational flexibility in terms of the range of management capabilities and the speed at which the latter can be used to enhance the ability to control the organization and to increase the management capacity to control (Volberda, 1996).

Building on various definitions from the field of IS and other reference disciplines, the term agility has been defined for the information systems development field to rely on two fundamental concepts, namely flexibility and leanness (Conboy, 2009). The flexibility of an information systems development method refers to the reaction and response to change but also to the initiation of change. A flexible information systems development method needs to align with its parts and its environment. The leanness of an information systems development method aims to provide additional value that customers perceive along with the economy, quality, and simplicity of the outcome. Thus, information systems development agility can be defined as "the continual readiness of an information systems development 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" (Conboy, 2009, p. 340). Hence, team agility is defined as the continual readiness of an SDT 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 concept of team agility supports the definition of emergent states as put forward by Marks, Mathieu, & Zaccaro (2001). Team flexibility refers to the capability of a team "to rapidly or inherently create change, embrace change and learn from change" (Conboy, 2009, p. 336). Creating, embracing, and learning are dynamic elements that can vary based on a team's function, as mentioned by Marks et al. (2001). Earlier the concept of leanness was introduced. A team is lean when it focuses on the customers and their perceived value. The focus on customers' perceived value also changes dynamically with other elements (Marks et al., 2001). The elements mentioned by Marks are context, input, processes, and outcomes. Here, changing customer objectives leads to a shift in their valuation of things. Alternatively, technological changes alter the way the team wants to achieve such customer value. Consequently, this study understands team agility as one form of an emergent state.

4.1.3. Conceptual Development

The input–process–state–output framework (Collins et al., 2013) builds a starting point for the conceptual development. Therefore, this study focuses on the forming stage and investigates the relationship between input factors and an emergent state (Ilgen et al., 2005). Each input factor is conceptualized based on prior literature and the corresponding system's description (Ilgen et al., 2005; Mathieu et al., 2008; McGrath et al., 2000). As mentioned above, team agility is the emergent state identified in the context of ASD teams. Consequently, a better understanding of the relationship between the team as a CAS and its emergent state of team agility is needed. This study proposes that the team's self-organizing characteristics mediate this

forming relationship. Self-organization requires that the team and its members be autonomous (Maturana & Varela, 1980; Vidgen & Wang, 2009).

The local system of a CAS refers to the individual team member (Mathieu et al., 2008; McGrath et al., 2000). Therefore, the local system is subject to different dynamics represented by the individuals' activities, such as using resources, tasks, or technologies (McGrath et al., 2000). First, within the system, the individual team member is one type of resource. Particularly in knowledge management, the experience gained by an individual or their collectives is a critical resource that builds up over time. An example of the importance of knowledge is the knowledge-based theory of the firm by Nonaka and Takeuchi (1995), which presents knowledge as a key resource for providing a strategic advantage to the firm. The theory is based on the resource-based view of the firm and points toward knowledge as an inimitable key resource (Kozlenkova, Samaha, & Palmatier, 2014). Hence, an individual's experience reflects an important resource within the local system. Prior experience in the form of task related knowledge increases the individual's self-efficacy (Latham, Winters, & Locke, 1994) and therefore, their independence in decision making. Consequently, experienced team members have more autonomy than less experienced team members do. It is suggested that the individual's experience enables autonomy.

Second, the task is an example of an individual's activity in the local system. In the work context, an individual's task is closely linked to its role. In other words, a role pre-defines certain tasks and expectations regarding the individual who has such a role. Given the expectations to complete various tasks enforced by the role, the explicit documentation and communication of such a role is important. Such communication helps the individual to understand the existing expectations of herself and is often referred to as job clarity (Salas, Rozell, Mullen, & Driskell, 1999). An individual who is clear about her job and the expectations toward her can act autonomously. The person is able to make rational and informed decisions. Hence, job clarity enables the team members' autonomy.

Third, the use of technology is another element in the local system. Prior conceptualization in information systems suggests that the team members' connectivity indicates local dynamics within the CAS (Alaa & Fitzgerald, 2013). In today's digital environment, access to technology, such as communication and collaboration tools, defines the team member's connectivity. Examples are globally distributed teams, where the team members are spread out around the globe (Sarker & Sarker, 2009). Thus, without access to technology, a team member is not able to communicate effectively with others. Access to technology is critical for a team member in order to be flexible in her communication and collaboration with others. This flexibility of a team member relates to her autonomy.

An individual's experience, job clarity, and technology access are relevant factors for a positive influence on the local system. Hence, the following proposition can be formulated:

Proposition 1 (P1): *For agile software development teams, the higher a) individuals experience, b) job clarity, and c) technology access are, the higher is the team's autonomy.*

The global system refers to the team (Mathieu et al., 2008; McGrath et al., 2000). The team's behavioral variables indicate different dynamics within the global system (McGrath et al., 2000). Such dynamics are not only the sum of the local dynamics but reflect the network, patterns, and links of the local system (Arrow et al., 2000). First, teams naturally form a network, as they comprise multiple team members (nodes) connecting with each other by means of their interactions (edges). Patterns emerge and can be steered through incentives and the team task itself. While incentives on an individual basis may lead to conflicts within the team, it can be difficult to associate incentives in an organizational system with an individual's work. Hence, a team's incentive can ensure goal interdependency (Z.-X. Zhang et al., 2007). While a common goal is a requirement for a team, such a goal may not be part of the organization's incentive structure. However, established goal interdependence within the team ensures that all team members work together toward a unifying goal and influences the team's orientation. Such unity and an achievable orientation help the team's autonomy. Hence, clear common goals enhance the team's orientation and therefore, its autonomy.

Second, the team task provides a common pattern and a link between the team members. Complexity is an important characteristic of the team task (Wood, 1986). Task complexity is especially important in the context of software development (Darcy, Kemerer, Slaughter, & Tomayko, 2005). While prior studies in the area of software development investigated software complexity (R. D. Banker, Davis, & Slaughter, 1998) or data complexity (Rajiv D. Banker & Slaughter, 2000), the role of task complexity has been studied in other context (e.g. Argote et al., 1995). However, it has been suggested that task complexity inhibits the adoption of ASD (Svensson & Höst, 2005). Consequently, low task complexity enhances the team's ability to use redundancies. When the team task is less complex, there is room for redundant functions within the team and therefore for allowing the team to self-organize (Moe, Dingsøyr, Dybå, & Ict, 2008). Lower task complexity positively influences the team's autonomy, as less complex tasks are easier to understand. A correct understanding is important in order to make an informed decision. Hence, less task complexity enhances the team's autonomy.

Third, the objective of serving the user's needs helps to build a natural link between the team members. While ASD tends to focus on functionality, its extension with and the importance of user research have been suggested (Brhel et al., 2015). User research enables the team to develop usable software and helps them to

determine future requirements that will influence upcoming development cycles (Grudin, 1991). Extracting future requirements and understanding the user's needs helps the team to develop a joint team orientation. This team orientation allows the team to act autonomously during the development process. Hence, the degree of user research relates to the team autonomy. Hence, goal interdependence, team task complexity, and user research are relevant factors in the global system. Therefore, the study proposes:

Proposition 2 (P2): *For agile software development teams, a) the higher goal interdependence, b) the lower team task complexity, and c) the higher user research are, the higher is the team's autonomy.*

The contextual system of a CAS is the organization and its environment (Mathieu et al., 2008; McGrath et al., 2000). Contextual dynamics nest within this system and influence the team's emergent state (Ilgen et al., 2005). In return, however, the team is unable to influence contextual dynamics, which are often organizational decisions such as development length. An organization-wide decision to adopt an organizational heartbeat pre-defines the development length. A heartbeat synchronizes existing product management and development processes (Vlaanderen, Jansen, Brinkkemper, & Jaspers, 2011). For example, establishing a clear development length forces re-occurring patterns within the SDT across releases. A release's development length typically reflects the frequency of such a heartbeat. A shorter development length of a release means a higher number of releases and a larger number of synchronization points. Such synchronization points help the team to learn and evolve on their own because of their autonomous behavior.

In a similar vein, management support has a positive influence on teams' self-organization. Hence, management support within the organization is important to develop the capability of self-organization. These organizational decisions are external forces that prevent the project team from developing autonomy (Moe et al., 2008), e.g. when they lack management support or the development length is unclear. This suggests that management support and development length are associated with the team's capability to self-organize. Therefore, the study formulates:

Proposition 3 (P3): *For agile software development teams, a) the higher management support, the higher the team's self-organization and b) the shorter the development length the higher the team's autonomy.*

Environmental factors and individual factors influence the teams' self-organization (Hoda, Noble, & Marshall, 2010). While self-organization is a process, emergence is the product (Curşeu, 2006; McCarthy et al., 2006). Teams' self-organization is an evolutionary process (Goldstein, 1999). As a result, an emergent state within the team can be observed. Here, team agility is the emergent phenomenon resulting from self-organization. Prior work also suggests the effect of team processes on teams' emergent states (Collins et al., 2013). A pre-condition of self-organization is the system agent's autonomy (Maturana & Varela, 1980). While definitions of agility vary, all seem to agree on the element of flexibility (Conboy, 2009; G. Lee &

Xia, 2010). Hence, agility is often referred to as the responsiveness to change (G. Lee & Xia, 2010). Prior work has shown that autonomy has a positive effect on software development response efficiency (G. Lee & Xia, 2010). Given those findings, the study proposes:

Proposition 4 (P4): *For agile software development teams, autonomy positively influences team agility.*

I summarize all propositions in the team agility framework (see Figure 16).

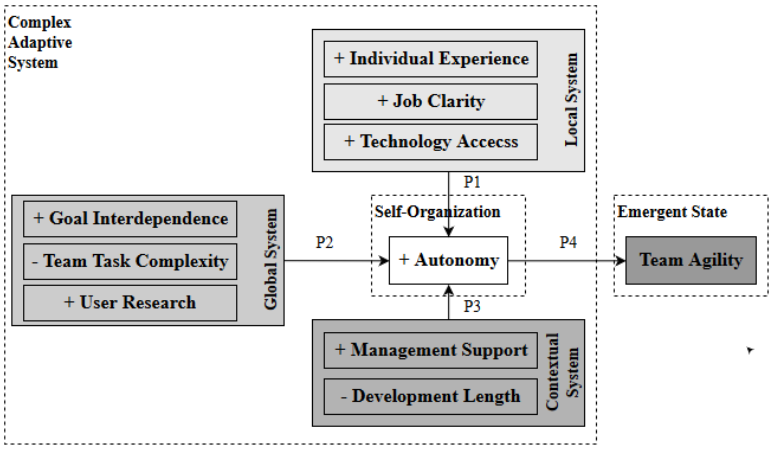

Figure 16 – Proposed Team Agility Framework.

4.1.4. Research Method

In the following, the research method is described along with the research design, data collection, and analysis (Dubé & Paré, 2003). Given the phenomenon's complexity and its multiple systems, a holistic multicase study is the applied research approach (Yin, 2008). Furthermore, the team agility framework is tested using a positivist case study.

Research Design

A multicase study using theoretical sampling logic was selected as the main research design (Eisenhardt, 1989). Companies are actively approached to identify cases that vary in terms of their agility. Three states of agility are investigated, i.e. starting the transition to agility, being in the middle of the agile transition, and being more mature with agile development. Through the varying degrees of agility as the emergent state, differences in self-organization and CAS can be observed. The differences help to expand the knowledge compared with a single case design. An introductory discussion between the researchers and the case organization determines whether a case is suitable. Both sides commit resources as part of the discussion, and the degree of agility is assessed (i.e. to determine the applied development method and the time of its introduction). The later in-depth analysis of the cases confirmed the initial assessment.

Moreover, relying on multiple cases increases the external validity of the findings (Yin, 2008). Given the complexity, software is typically developed in teams (Sawyer, 2004). Hence, the unit of analysis is the ASD team. In total, the study encompasses three case teams from three different organizations. Within each team, at least four different perspectives are obtained: a) management or Mgt. (e.g. product manager, portfolio manager or general manager), b) technical supervisor or Tec. Sv. (e.g. development team leader or head of development), c) developer or Dev. (e.g. junior engineer or senior engineer), and d) designer or Des. (e.g. interaction designer, usability engineering or designer). Investigating at least four perspectives helps to identify differences between the roles but also to identify common patterns across the team. Table 29 presents an overview of the 16 interviews and the different roles of the interviewees. Each team's interviews were conducted on the same day. Team ALPHA also served as the pilot case, leading to some adjustment and refinement of the semi-structured questionnaire. In response to this team's participation, the cases received actionable recommendations that help to take action against the identified status quo. While the first authors conducted all the interviews, key deliverables such as case reports and interview guidelines (see Appendix E) are the subject of a discussion within the author team. All teams received the results, and two of three teams were open to an in-person feedback session led by both researchers. The feedback session includes a presentation of the case report's results and allows the case organization to make additional comments.

Case Description

Team ALPHA is a small software development organization with fewer than 25 employees. The founder and owner runs the organization. The main product helps sport associations to manage their business. The organization values sports and serves many sports clubs in Europe. Its development focuses on a core product that assists in the management and operation of sports clubs. The core SDT encompasses 11 team members; seven of them are based remotely in Russia, while four work at the organization's headquarters in Germany. The remote team members have been working for the company for many years. Overall, the team is very specialized with little cross-functionality. The development length of the investigated release is greater than 12 months. Although the organization is mostly located in Germany, the customer market stretches beyond Germany's boarders to include France, Austria, and South Africa as key markets. The team only recently adopted ASD and focuses on evolutionary elements during the development, such as early delivery and welcoming change. They also use basic agile practices, such as sprint-meeting and a backlog. However, the team's use of agile practices has not been ritualized yet.

Team BETA is a medium-sized software development organization. The organization, a subsidiary of a publicly traded holding, has fewer than 100 employees and focuses on the development of administration add-ons for collaboration platforms. The organization also develops software in the area of business process

management, mobility, and security. Here, the team develops an email management system with enhanced security measures. The SDT has 11 members, five of whom are developers. The team has two product managers, a designer, and three quality assurance members. All members are located at the same German site. The team does not extend functional boundaries. The development length of the release investigated is six months. Customers are globally dispersed at locations in USA, Canada, India, and Germany. The development length is 12 months. The team adopted Scrum as their leading agile method and has learned to improve it since its introduction two years ago. The team collaborates through face-to-face meetings and tracks their progress through working software. They adopted sprints with a 4-week cycle.

Team GAMMA describes a larger medium-sized organization with fewer than 400 employees. The organization is a public entity and part of a larger holding. The organization focuses on the financial services sector. Its objective is to develop innovative and intelligent solutions tailored to the individual user's needs. The organization offers products for controlling, product management, value-based product consulting, and a back-end library. The product investigated is a dashboard application that helps to visualize financial information. The SDT has seven members; five of them are developers, one is a scrum master, and the other is a product owner. While four developers work close to the customer in Frankfurt area, three members are based at the organization's headquarters in Germany. Cross-functionality within the team is low. The organization is a subsidiary of a larger international firm, and the customer base is mainly German speakers. Team GAMMA is the team with the highest agile maturity, having adopted Scrum and used it for the past three to four years. In addition, they focus on their people and reflect on their prior experience to improve their agile practices. The team adopts a 2-week cycle and grooms their backlog, i.e. the continuous update of their backlog based on recent discoveries.

Preparation and Data Collection

Publicly available information (e.g. through websites) is analyzed as a first step in the preparation of the case study. This information helps to form an understanding of the business context and products. In addition, this information reveals organizational information, such as team members and responsibilities. There is particular focus on the preparation of the semi-structured interviews. The development of an interview guideline enhances the main data collection method. During the development of the questions, suggestive forms are avoided. Such guidelines ensure that permission is received to record the interview, ask pre-defined questions, provide room for notes, and a one-page survey at the end. The survey captured demographic information related to the individual and the team. Information related to the individuals included the years of experience in usability engineering/interaction design, traditional and agile software development. Team related information included the team size, team member's roles, and their location.

The investigators' skills have proved to be a critical factor in carrying out case study research (Yin, 2008) and while conducting prior multicase studies. In addition, a separate preparation interview with a practitioner helps to improve the interview guidelines. The practitioner has many years of experience in software development and has a management role in an international software development organization. This helps to prepare the investigator and train the interviewer for data collection in the field. It also provides another means to evaluate the interview guideline. The data sources and their use are an important step to assure the results' reliability and validity (Benbasat, Goldstein, & Mead, 1987), and Table 29 provides an overview. For participation in the case study, interviews with at least four different roles are required in order to get a comprehensive understanding of team-level perspectives. Prior communication and alignment between the case company and the researchers assured the correct mapping of the company's internal roles to one of the four perspectives. Key decisions, along with the case study's purpose, data collection, and interview questions, are documented in a case study protocol prior to the collection of data for the initial case (Yin, 2008). A similar investigation of the cases increases the findings' reliability.

Table 29 – Overview of Data Sources, Types, and their Use in the Analysis.

Data Sources	Types of Data	Use in Analysis
Semi-structured interviews (ALPHA; 254 min.; 36,075 words) (BETA; 354 min.; 50,483 words) (GAMMA; 339 min.; 51,483 words)	Interviews with experts (7 in total): ALPHA with 1 developer and 1 designer; BETA with 2 developers and 1 designer; GAMMA with 1 developer and 1 designer	Investigate the software development process Identify local and global system factors and their effects
	Interviews with management (4 in total): ALPHA with 1 manager; BETA with 1 manager; GAMMA with 2 managers	Understand management perspectives on the phenomenon Identify contextual system factors and their effects
	Interviews with technical supervisors (5 in total): ALPHA with 1 technical supervisor; BETA with 2 technical supervisors; GAMMA with 2 technical supervisors	Investigate the maturity of the software development process Identify global and contextual system factors and their effects
Quantitative data	Survey (16 in total): individual and team-related information from all interviewees	Support and cross-check accounts from interviews
Archival data	Organization's website: organizational and product-related information Emails: prior email exchange and tool screenshots	Support and cross-check accounts from interviews

The data collection relies on multiple sources and triangulates documentary evidence, such as websites, emails, and screenshots, with interviews and observations, as well as survey data (Yin, 2008). The use of multiple sources also increases the validity of constructs used and allows for triangulation (Yin, 2008). However, the key data collection method is semi-structured retrospective interviews (Schultze & Avital, 2011). The use of laddering interviews helped to extract antecedents and their impact. Laddering interviews use comparisons and contrasts to identify patterns that are subsequently understood in greater depth through

repeated inquiry related to the how and the why of such patterns (Schultze & Avital, 2011). The interviews were conducted in German and English, following the prepared interview guidelines. As a result, the CAS, its characteristics, and team agility were assessed. The basis for such characteristics is the conceptual team agility framework. Each interview lasted approximately 60 minutes, ranging between 50 and 67 minutes. The interviewer took additional notes before and after the interview. Each case contains interviews with at least four team members. Table 29 presents an overview of the different cases and their interview length. All interviews were conducted during face-to-face meetings at the company's main location. In order to maintain honest responses from the interviewees, a new room had to be requested in which the interviews of the pilot case could be conducted. Given that the initial room had been in front of the CEO's office, separated only by glass, this new setup avoided any potential influence on the interviewees. The transcripts fed the case study database using NVivo to support the coding process.

Data Analysis

The data analysis includes three steps: preliminary data analysis, within-case analyses and cross-case analysis. The first step uses preliminary data analysis techniques and tools (Dubé & Paré, 2003). Using a pre-defined set of codes, a systematic extraction of interview data helps the later analysis. The initial list of codes included different factors as the result of a literature study on agile teams that also informed the framework. A coding sheet including the codes name, a definition from literature, and example text from the interview data that matches the code, documents the codes. Nevertheless, while the coding is performed, existing codes require adjustment, and new codes emerge (Seaman, 1999). For those cases where a statement does not match any existing code, a new code emerges. This process lead to three tables with 237, 184, and 234 coded text segments. The final list of codes, including their empirical observations, presents the relationship between the conceptual and the empirical level (see Table 30). In addition, field notes document informal information exchanges and non-verbal information. The use of data displays during the early stages, but also during later analysis stages assisted the researchers to visualize and discuss different interim results.

Second, within-case analysis for each case for each case. The number of words identified for each code indicates the importance of each code: If a factor is not important or has no influence, the interviewee will have little to say. In addition, such descriptive data along with the various roles and cases was analyzed. While they only provide an indication, the information certainly requires completion with qualitative data from other sources and a content analysis. Given the assessment of each case with different team members in different roles, rich case reports were generated. The use of multiple roles and different sources of evidence allows a triangulation of findings. Each case concludes with a case report that discusses background information of the company, its current status, and future recommendations. Each report builds

on a logical chain of evidence (Yin, 2008), taking also contextual information into consideration. The presentation of such reports helps the evaluation of the findings and recommendations.

Third, the data was sequentially aggregated in order to conduct a cross-case analysis. In an initial step, all data was extracted into a spreadsheet that presents all coded quotes. The quotes are a grouped list based on the combination of factors and interviewees. In other words, this step considered the full set of codes (number of interviewees times the number of codes). All quotes of a given group present an initial summary. The next step is to aggregate all the summaries for a given code and to structure them along the initial framework. A cross-case comparison assessed the replicability and variance of individual factors (Yin, 2008). Here, the influence of such factors and use of CAS as a theoretical lens, which helps to structure the findings, was investigated. Additional codes are considered, as there might be some control factors. Moreover, the findings are compared with factors and effects from existing literature. Interim results are discussed between the authors and presented to other scholars in order to strengthen the findings.

Table 30 – Overview of Data Structure.

System	Codes	Observations
Local system	*Individual experience*	Years of experience in usability engineering / interaction design, software development and agile software development
	Job clarity	Task separation and delegation; separating product owner and development team; Product Manager manages customer contacts and is responsible for one product; tasks vary widely
	Technology access	Free and flexible to choose; easy access; higher investments need to be justified
Global system	*Goal interdependence*	Missing goal interdependence; either individual or organizational goals
	Team task complexity	High complexity; historicity of the product; unused functionality within the product
	User research	Access: incomplete access to users; often link to an intermediary (e.g. management) or a specific event (e.g. training, annual workshops) Knowledge: incomplete knowledge about users; based on own experience and role dependent
Contextual system	*Management support*	Varying from high support through involvement to low support with a focus on basic workflows
	Development length	Time of the development for the release; delta between start date and end date
CAS	*Self-organization*	Varying extent of autonomy and ability to make informed decisions, degree of team orientation and focus within the team

4.1.5. Findings

The teams studied in this article indicated different emergent states of team agility. The following presents the within-case analyses of each case. Thereafter, commonalities and differences are identified across the team. The analysis helps to investigate conditions that lead to different emergent states of team agility.

Within-Case Analyses of Teams as Complex Adaptive Systems

Team ALPHA: Little Experience and Unclear Expectations

Team ALPHA has the least agile experience with 1.5 years per team member on average. While the technical supervisor is experienced with agile methods, most team members have little experiences and therefore, rely on the supervisor's experience. Besides relying on individual's experience, a team can specifically assign agile roles to the team member to enhance their role clarity. Here, on the contrary, the assignment of two roles, such as scrum master and developer, to a single individual indicates less clarity of the team member's job. This constellation leads to the abatement of the earlier role for the sake of the latter due to partly conflicting expectations. When expectations are not clearly defined, team members have little control over their tasks. The technical supervisor describes the neglect as follows: *"Previously, we received tickets, and we did the most urgent task first. But honestly, many tasks were discarded as a result"*. This suggests, when work packages come in as needed without a clear structure or area of responsibility, but rather as a means to convert needs into features, the team member has only limited information, which prevents an informed decision. In addition, management considers developers as coders instead of problem solvers: *"OK, the design is more in our area of responsibility, and I would express the development mainly as coding. We coordinated the dialogues"*. While some roles are reduced to their basic tasks, other team members struggle to understand each other's role within the team. As a result, an imbalance of responsibilities and expectations occurs, and a single role becomes dominant. In this scenario, other team members are very dependent on such a dominant role, which limits their autonomy according to the technical supervisor: *"[The General Manager] was always in charge of everyone; his responsibilities are only increasing, and his time becomes limited. Now, he delegates tasks. That is fine. Thus, it cannot get out of control within his remit. However, who conserves the consistency of the 'big picture' of the product and is responsible for this? This is one burden over time. There is no product management with the capability to think holistically".* The access technology is easy and the developers are flexible to choose the technology they deem appropriate. While most development applications are open source software, new technology can be acquired when the need occurs.

Evidence suggests that interdependent goals are either informal or allocated on a different level. While there are no formal goals for the team, the technical supervisor reports that customer satisfaction serves as an informal consensus across team members: *"There is no bonus structure here. In addition, I do not receive any direct bonus either. So are there any potential conflicts within the goals or interests between individuals? I do not think that is the case. I think everyone has the objective to satisfy the customer, whether it is verbalized or not"*. While an informal common goal exists, it is not reflected within the application and the unnecessary complexity makes it difficult for the new team members to understand the software in its

entirety. As a result, this complexity is not masked through a simple UI. Rather a developer indicated that large parts of the software's functionality are not used. The main information from and about users stem either from own experiences or are acquired through customer interactions during trainings and conferences. Given these infrequent and situation feedback mechanisms, the team has little incentive to act upon them.

Management support of user-related concerns is superficial. While ad hoc support from management is prominent, the technical supervisor reports that teams lack sustainable solutions, especially in times of crisis. As a result, budgets related to the concerns of users are modest and lead to a delay or deferral of investments. Management postpones the investments into user-related needs, such as usability, until another version of their product: *"Well, we invested some money at the beginning. However, rules were established at a later point in time, and as a result, no one thought about the design anymore. So, we delayed the design parts and reserved it for [software version n+1], resulting in the fact that no decision has been made in this direction for a long period of time."* For the current release, the available person-days provide a natural limit on their efforts and rely on management to prioritize tasks accordingly. The team develops more than 12 months on a release and therefore, requires the most time to develop a new release.

Within Team ALPHA, the extent of autonomy and self-organization is low. Since developers are seen as coders, general management takes most decisions. When the company used to be smaller, the General Manager was deeply involved in the development efforts of the product. At present, given the growth of the company, the General Manager's other duties have increased. At the same time, the delegation of authority and responsibilities from the General Manager do not synchronize with the company's growth. Thus, the team has limited autonomy, which challenges a mature adoption of agility. In addition, the General Manager makes many decisions alone, which often leads to an opaque decision-making process. Combined with the lacking authority, the team is not able to make informed decisions. An increased dependency on the General Manager and therefore, a deficient ability to develop new structures or patterns within the team is observed. This suggests a lower extent of team agility.

Team BETA: Challenges of Complexity and the Value of User Research

Team BETA has 2.2 years of agile experience per team member on average. The technical supervisor reports that as long as roles are clearly defined and assigned, one team member can work on multiple projects with rather positive effects through the convergence and the combination of efforts by both project teams. However, in the case of an important role, such as the product manager, one team member is solely responsible for one product. Management suggests that new technology also enhances the coverage of developers' needs. While simple chat applications have been available in the past, the introduction of social

platforms enriches the interactions between developers by providing additional meta-knowledge about each other.

There is no financial goal interdependence for the team, but a flexible portion of the salary relies on the overall firm performance. In addition, team members' goals are driven by an informal goal interdependence, as management expressed: *"We have feedback talks where we discuss the current performance of the team and whether everything works fine. Hence, naturally I think my success is also measured by the team assessment"*. However, the interviews suggest that all of the goals, regardless of their subject, need careful determination. Inadequate goals can have negative effects on the team's process. Management shares prior experience in which the defined goals were not challenging for the team: *"I personally had the impression that it becomes irrelevant at the end of the year. If I hadn't changed the goals, we would have delivered decently, and there is no point in it".*

Implementing complex changes depend on very specialized individuals and are therefore less autonomous. This results in a delayed implementation of such changes or generally led to time constraints due to such dependency, as management suggests when confronted with externally caused changes: *"We didn't have the time. These would have been profound structural changes, and there was no time".* Moreover, the limited redundancies that result from specialization also decrease oversight, resulting in complexity presented to the user, as described by the designer: *"There are so many field options that [the user] sees, and I think that 90% of the members or users do not need such functionality. There is also too much information for the user. Initially we had to provide this functionality. This functionality has to go somewhere. It would be nice, when we could arrange it according to some logic through which the likelihood increases that the user finds it. This is for those cases where the user does not check the manual, which happens most of the time".* According to management, increased product complexity is often a legacy, i.e. a result of the product's historicity. Often, developers and designers who implemented older features are no longer part of the team. Consequently, when such a feature requires a change, the team has to learn about the structure of such a feature again, management indicates: *"In fact, a lot of options were needed once within the past 20 years, and after the fact there was a realization that the functionality was barely used. Consequently, the functionality did not evolve. However, it was still kept in the product. In this case, when the original developer left the company, no one else had any know-how about this functionality. Over time, there have been a couple of these functionalities".* Such learning efforts are often seen as unnecessary investments in old features and are stigmatized with the notion of legacy. Therefore, the team is less motivated to build redundancies for such features either due to an expected timely removal or due to an expected stability.

User research helps the team to develop an orientation that enables autonomous behavior. User research requires the means to both gather information about users, i.e. access to the user base, and store and

accumulate information about the user, i.e. knowledge available about the user base. The team's user access allows the team to iterate ideas, concepts, and problems quickly with users of the software. Such a direct link avoids miscommunication that can result from introducing intermediaries. Evidence suggests that user access improves the team's ability to interact swiftly with users and extract requirements. The expected results in the form of additional feedback clarify user needs and help the team to act autonomously to develop better software. However, it is often difficult for the companies to get access to users as expressed by a developer: *"Yes, in terms of user friendliness, they will surely have many suggestions. However, often in larger organizations this is not supported, and it becomes very difficult for us to get an appointment, as those businessmen surely have other tasks, especially within the administration area".* Depending on the user base, the term "user" may lead to some irritation, and the use of roles is preferred. For example, when someone is an administrator within the customer's organization, naming him a user might be considered offensive. Using the role helps the team to understand the user's perspective and further shapes the team orientation: *"A customer survey is a great thing. For us, the customer is the administrator who decides that his company needs such a product. From our point of view, the administrator is the customer, as he orders the product for his firm, needs them, and in fact he is not the end-user who actually works with the product. Thus, it is not the end-customer who actually works with the product".* Besides good user access, the team should also encompass sound knowledge about its user base. The sheer existence of user access does not imply a good understanding of the user's tasks and context. This knowledge will help to enhance the lean aspect of agility, streamlining activities and limiting them to those that drive the perceived user value and therefore increase the team's orientation. A negative effect of little or no user knowledge is extra development efforts. Such efforts are caused by the team's disorientation and result in a mismatch of user needs with the designed solution as expressed by a technical supervisor: *"Absolutely, this often occurs in a painful way, probably with your first customer, if the requirements were not exact but loosely defined and not questioned afterwards. So you give [the software] to the customer, and then you realize, 'Oh, the technological landscape is different from ours.' There is an additional layer, and [the software] does not fit their environment and will not work".*

The initial interest and willingness of management to address user-related concerns quickly fade away when the team requires actions or the allocation of budgets, as mentioned by a technical supervisor: *"[Top management] is interested in improvements that are also communicated as the need to become more user-friendly. The fact that this also requires the availability of resources is not necessarily seen and supported"*. The team appreciates receiving initial support; however, when funding is required, progress slows down. Therefore, the BETA case builds on current planning activities by putting more emphasis on user-related concerns for their future development efforts. The team's release cycle is aligned to the annual release cycle of changing formal requirements related to financial transactions.

The findings suggest a higher extent of autonomy and self-organization in Team BETA. After the team had made the decision to adopt ASD, the technical supervisor reports experienced difficulties and challenges that follow the introduction of a new paradigm. However, the supervisor has reported positive effects experienced during the past year. While the team presents a higher extent of agility because of increased autonomy and self-organization, challenges remain. Management reports that there are still some traditional views, e.g. starting with the development of the back-end and continuing with the front-end. Yet, techniques such as prototyping and sketches that work well within agile environments suggest a different order. They identify and acknowledge the problem. Management is committed to addressing these challenges in the future. While they have already taken the first steps in this direction, more time will have to pass before the team has truly incorporated the idea.

Team GAMMA – Open-minded Management and Short Development Cycles

Team GAMMA's team members have the most experience with ASD (3.75 years on average). The team members in Team GAMMA are less dependent on each other and are rather autonomous. The team defines roles clearly and establishes a contractor/client relationship within the organization. As a result, team members clearly assign themselves to one of the two perspectives, hence increasing role clarity for that member. Ensuring this is a management responsibility: *"We are considering this as a management task as we want to establish a principal agent situation, where some people are responsible for ensuring the attainment of customer requirements and economic success. Taken as a whole, it is the separation of the product owner and the development team".* While the assignment of two projects to one team member is beneficial, the assignment of two roles to one team member results in a conflict. In the latter scenario, the team member's performance and contribution decrease, as explained by a developer: *"I assumed a twofold role at this point, 50% | 50%, based on restricted thinking. This means that it is not possible without one [scrum master], and [interests] collide. We tried it, but I noticed that I could not fulfil the scrum master role. It doesn't work and is not justified for the team and leads to worse results".* The designer positively mentioned that a clear separation of tasks helps the team members to understand their contribution and the contribution of each member with whom they are interacting. In addition, this meta-knowledge helps the team members to enhance efficiency, as they know when to contact a colleague for further information in order to make an informed decision.

While there are no team related financial incentives, the organization measures its organizational success. When goals are achieved, employees receive a reward. Thus, team members and the team as a whole work thrive to achieve success to which they have only limited influence. The team members report the application to be highly complex in the backend, yet they seek to provide a simple and easy to use interface. The team reports that they have a good and continuous access to the user. The access to the user can vary with the

team member's role, but regular workshops, occasional on-site visits and accessibility through phone allows the team to build substantial knowledge of their user base. This is important, as the technical supervisor realizes that both, the benefit of user knowledge and obstructions or the lack thereof, is only visible after the fact: *"[Usability] was not a major topic in the old product version. We thought we had understood the customer. Only after the fact do you even realize that you did not. In the new product that we are currently working on, we used prior prototypes and interviews with recordings to consider the whole task and in the work environment."*

Instead, management focuses on the core workflow with clear and visible benefits to the team's operations and its organization, as mentioned by management and designer: *"It is a question whether I integrate [usability] into an already existing [software] and how I can integrate it into a grown software with the budget that the customer has already paid for. I can certainly change smaller things, but I cannot change the basic workflow again. The other thing is that, when you start doing something really new, management is pretty engaging, so then it has to be delivered".* Such prioritization helps the team to understand management's expectations, and therefore the team can adapt appropriately. Knowing the internal budgetary limitations, the team can find an alternative approach to fund such initiatives and efforts through specific customers. Management suggests: *"There are budget restrictions where I say that I won't invest a few hundreds of thousands [of euros] in the current product line so that the customer gets along better. For that, he should pay money for the next product line, which has great usability".* The team has the most experience with agile software development and the shortest release cycle. The team releases a new version every six months. Team GAMMA also follows the Scrum method.

The team mentioned the benefits of accessing users directly. The feedback from users enables the team to align the product vision with the users' needs. Hence, the technical supervisor reported that they consider such early feedback to be a valuable source of information: *"This approach was beneficial as we did not say what the customers had to do, but we developed a vision that was subsequently adjusted in collaboration with the customers to meet the customers' expectations exactly".* Such collaboration with the user is a result of their high extent of autonomy and their ability to self-organize. The team's high extent of autonomy leaves them to self-organize according to the feedback from the users. Such feedback allows the team to enhance their understanding of the perceived user value and therefore increase their extent of agility. The interviews indicated that the value of such feedback is different in each team. As Team GAMMA developed a new application, the technical supervisor pointed out that they had the chance to be more open to new ideas as part of their development process. Overall, the findings suggest a relationship between the self-organization of the team and team agility.

Cross-Case Analysis: Explaining Different States

The within-case analyses show that all three teams have different emergent states of team agility. As a result, an overview of the three cases is presented (see Table 31). With the exception of technology access and team task complexity, the team's antecedents vary across the cases. The *individual's experience* is one of the team's critical resources. The team members' experience related to ASD varies. Individual experience can either facilitate autonomy, as in the case of GAMMA, but it can also prevent autonomy, as indicated by team ALPHA. Hence, when team members gain experience they are less dependent on external stimuli and thus, increase their autonomy. *Job clarity* requires that the individual understands her role and tasks associated with that role in order to manage expectations and to gain autonomy. In an environment where roles and responsibilities are not clearly defined, team members struggle to discretely decide on their tasks (e.g. ALPHA). When it becomes clear what is expected of each role, people have more freedom on how to meet such expectations as seen in case GAMMA. A team member does not have to be assigned to a single project. Rather, when resources are scarce, a team member can work on two or more projects, while still having more freedom over the execution of their tasks (e.g. BETA). *Technology access* enhances a team member's flexibility. Given the ubiquity of technological advances within the software industry, easy access to new technology seems to be a mandatory requirement for developers in order to perform well. New technology also enhances the coverage of developers' needs. While technology access is important, findings did not suggest any differences between the teams. Rather it seems to be a boundary condition for software development in general. In summary, an individual's experience, job clarity, and access to technology characterize the local system; yet, the teams had a good access to technology.

While team task complexity remains stable within the global system, the findings indicate differences in the goal interdependence and user research across the teams. A formal *goal interdependence* that would enhance the leanness of the team is absent in all cases. Yet, the findings suggest organizational goals that influence team member rewards and therefore, reduce the teams' dependency on external stimuli (BETA and GAMMA). Team ALPHA does not have a rewards structure that targets the enhancement of joint efforts. *Team task complexity* inhibits team autonomy by introducing chaos and limiting their ability to create redundancies. Teams that develop highly complex products are less autonomous. All three teams report high task complexity, suggesting team task complexity to be a boundary condition applicable to development tasks. The access to and knowledge of users through *user research* help the team to becoming more independent in their task fulfillment. When the team has easy access to users and has already build a substantial knowledge base about its users, they develop a joint vision and are less dependent on other functions to clarify requirements and needs, as in the case of BETA and GAMMA. Contrary, ALPHA depended heavily on the expertise and network of the general manager and therefore, has less freedom in

structuring their tasks. Hence, the team's goal interdependence, team task complexity, and user research reflect the local system; yet, all teams have a high team task complexity.

The contextual system, represented by the management support and development length, varies across the three teams. *Management support* plays an important role in defining boundary conditions that allow the team to self-organize. Within Team ALPHA, support by management is rather superficial, making it increasingly difficult for the team to discretely decide on means to accomplish their tasks. While team BETA receives moderate management support, team GAMMA benefits from true management commitment. Thus, the team and management have a joint vision and understanding of the overall goals and accomplishments, which the team seeks to pursue rather independently. The *development length* resides in the organizational system and influences the team orientation. The organization seeks to introduce a heartbeat that establishes clear deadlines synchronizing the work at critical milestones. The development length observed varies from six months to more than 12 months. The teams with a shorter development length have smaller cycles in which they can adapt and self-organize. Hence, decisions and experiences that only occur at the end of such a cycle happen more frequently. This increased frequency of feedback helps the team to gather more experience and enhance their team orientation. Therefore, it allows the team to self-organize over time. Hence, it is important that management support responds to user-related needs and to development length.

Table 31 – Case Comparison.

Team	**ALPHA**	**BETA**	**GAMMA**
Individual Experience	↓	→	↑
Job Clarity	↓	→	↑
Technology Access	↑	↑	↑
Goal Interdependence	↓	→	→
Team Task Complexity	↑	↑	↑
User Research	↓	→	↑
Management Support	↓	→	↑
Development Length	↓	→	↑
Autonomy	↓	→	↑
Team Agility	↓	→	↑

4.1.6. Discussion

The paper proposes, empirically tests, and extends a team agility framework. This subsection starts by presenting the contribution. A discussion of the theoretical and practical implications follows, before pointing out the study's key limitations.

Theoretical Contribution

The study suggests three theoretical contributions. First, this study takes a different perspective by adopting the team as the unit of analysis, whereas prior research focused on the development method (cf. Alaa & Fitzgerald, 2013; Kautz, 2012; Meso & Jain, 2006). Adopting a team-level perspective helps to understand the nature of the agile phenomenon. This study responds to calls for more research related to human and social aspects (Moe et al., 2010). One perspective toward agility and the adoption of ASD is the process-oriented view, which seeks to understand the steps leading to an agile development process. As the phenomenon of agility is also expanding to other areas, this can certainly be an important step. However, the agile manifesto puts a great deal of emphasis on the importance of individuals and their interactions, suggesting a psychological and social phenomenon.

Second, following calls for more theory-based research (cf. Dybå & Dingsøyr, 2008), the study uses the theory of CAS. Using CAS as a theoretical lens, the study facilitates an understanding of the characteristics of self-organization and emergence within the context of SDT. Following the representation of local, global, and contextual system, their influence on the self-organization of the team is determined (p1-3). The results suggest that the local system is associated with the self-organization of the team (p1). Here, findings support job clarity (p1a) and individual experience (p1b), which help the team to withstand outside pressure (Anderson, 1999). When an individual knows the expectations toward her and her team members, the team is more likely to self-organize. Similarly, the more experience the individual team members have, the more likely they are to be autonomous, supporting the self-organization of the team. Contrary to the proposition, the study does not find support for p1c (technology access). The individual's access to technology rather seems to be a mandatory condition captured by the studies' context of software development.

The results also suggest that global system relates to the self-organization of the team (p2). The study finds that goal interdependence (p2a) and user research (p2c) influence the self-organization by assisting the teams' monitoring and managing their own processes (Hackman, 2002). Goal interdependence and user research help the team to identify their common goals. Such a goals become the common denominator that drives the team's and the individual's autonomy and therefore improves their self-organization. Contrary to the suggestion, the study does not find support for an influence by team task complexity (p2b). The context seems to predefine the team task complexity. Software development seems to be inherently complex and perceived as such by the developers.

Contextual system can restrict or enable self-organization within the team (p3). The findings suggest an association between management support (p3a) and self-organization as management support can strengthen

the forces of self-organization within the team and prevent external forces from limiting self-organization (Anderson, 1999). On the one hand, the level of support is very important, i.e. not only a verbal commitment but also the corresponding actions and financial support should follow. On the other hand, simply articulating the management's stance toward certain issues already increases transparency, and hence the team can take appropriate actions that are in line with management expectations. Thus, it is necessary to avoid spending resources on objectives that will not find support from management. Both strengthen the autonomy of the team and thus, relate to their self-organization. In addition, a shorter development length (p3b) within the team increases the feedback cycles and learning mechanisms. Both are important for the team to enable team orientation and self-organization.

Higher extent of self-organizations helps the team to improve its agility (p4). First, the team needs to have the corresponding autonomy that is fundamental to self-organization (Moe et al., 2008). Team members need to receive the information necessary to make informed decisions. Second, the team members need to have the freedom to make mistakes and learn from those mistakes as an ongoing journey toward self-organization and self-regulation (McCarthy et al., 2006). Those cases resemble the highest extent of team agility. The findings also suggest boundary conditions, such as team task complexity and technology access. However, the context seems to dominate these boundary conditions.

Our third contribution is the investigation of differences between immature and mature agile teams (cf. Dingsøyr et al., 2008). The study identifies boundary conditions that influence the team's self-organization and therefore, their agility. Such conditions help to understand the characteristics of mature agile teams and explain how autonomy can be achieved (Moe et al., 2008). Moreover, these characteristics help immature teams to identify areas of improvement when progressing on their agile journey.

Practical Implications

The study has two practical implications. First, practitioners receive a list of influential characteristics. Practitioners can leverage the characteristics in the form of a checklist in order to assess the status quo of SDTs. As a result, they can investigate weak characteristics and strengthen others in order to increase the agility of the team. This is especially useful for teams struggling to move from a process-centered view of agility toward a mature and cost-effective state of agility. Second, practitioners need to be aware of the importance of a common and joint objective. Goal interdependence is one means to formally embed such objectives into an organization. They can also develop organically by adopting user research. For those cases where the organization lacks the expertise to conduct user research, consulting agencies, or experts can extend the team's expertise. They also provide another view onto the issues and challenges faced, allowing the organization to benefit from their experience and expertise.

The research also has two limitations. First, this work seeks to understand different states of team agility and suggests the means to achieve them. Therefore, the study focuses on the direct relationships of the different systems toward team agility. Hence, future research could investigate the relationships between the systems. Second, while there are potential replications of the concepts to other agile teams, e.g. in new product development, the context of this study does not allow to generalize to such teams. Hence, future research could replicate aspects of this study to teams beyond agile SDT.

4.1.7. Conclusion

This paper proposes, tests, and extends a team agility framework. It contributes to the body of literature by developing and investigating theory-based propositions. Contrary to prior studies, this work identifies agility as a team-level phenomenon. Starting with a review of extant literature on CASs and ASD in the context of teams, theory-based propositions have been formulated. Agile SDTs are one form of a CAS. The study focuses on the characteristics leading to self-organization and suggests team agility as an emergent phenomenon within SDTs. Consequently, this work helps to understand the characteristics of self-organizing teams. In this study, teams are understood as one form of a CAS. Furthermore, team agility is an emergent phenomenon that evolves over time, and it is the result of a process of self-organization.

The CAS comprises three systems, i.e. the local, global, and contextual system (Holland, 1992). In the local system, the study finds job clarity and individual experience to enhance self-organization. Within the global system, goal interdependence and user research improve self-organization. Within the contextual system, management support and development length can help the team to self-organize. Technology access and team task complexity are mandatory conditions within the context of SDT.

4.2. Emotional Leaders in Open Source Development Teams - Examining Collaboration Structure and Function[9]

4.2.1. Introduction

In today's highly dynamic and complex environment, it is absolutely critical to establish high-performance software development teams (Burke et al., 2006). Seeking to recruit the most suitable team members for specific tasks independent of their location (Kirkman, Rosen, Tesluk, & Gibson, 2004), virtual teams have been suggested. These teams are increasingly popular, as they rely on advanced collaborative technologies and web-based software repositories in order to communicate and coordinate their work. Open source software (OSS) development communities represent a specific form of virtual teams that benefit strongly from the technology-empowered ability to work virtually (B. Fitzgerald & Feller, 2001). In a professional environment, management relies on collaboration and work practices for achieving high performance (Posthuma, Campion, Masimova, & Campion, 2013) that cannot be generically applied to OSS. While businesses can easily rely on task-focused leadership practices (e.g. initiating structure) due to their power structure, leadership in virtual OSS teams cannot. Instead, leadership relies on the employment of emotional displays for achieving cohesion and high performance (Burke et al., 2006; Kahai, Carroll, & Jestice, 2007; Rajah, Song, & Arvey, 2011). An emotional display is an emotional cue given towards the team and its members that is not felt; often, emotional displays are employed in order to influence the team member's emotional response and collective emotions (Humphrey, Pollack, & Hawver, 2008; van Kleef et al., 2009; G. Wang & Seibert, 2015). Hence, there is a need to better understand the effect of OSS team leader's emotional displays on the team's emotional displays and to consider possible conditional effects on this relationship (Charlier, Stewart, Greco, & Reeves, 2016).

In addition, the investigation of team emotions enjoys increasing interest by the academic community (Barsade & Knight, 2015; Collins et al., 2013; van Kleef et al., 2009). Team emotions are known to play an important role in work teams and help to predict different team processes and outcomes. Examples are the effect of team emotions on team's coordination behavior (Homan et al., 2016), cohesiveness and responsibility of the team members (Magee & Tiedens, 2006), and team performance (Knight, 2015; van Kleef et al., 2009). Yet, factors influencing the team emotion and possible conditional factors that explain such influence are not well investigated (e.g. Døjbak Håkonsson et al., 2016; Homan et al., 2016; Magee and Tiedens, 2006). Prior conceptual work suggests influential team members, such as team leader to have an effect on the team's emotions. The few examples that investigate team leader's emotions are limited to positive emotions (e.g. Van Kleef et al., 2009; Neff et al., 2014). In addition, emotions are subject to contagious mechanisms, in which the leader takes an important role (Barsade, 2002; T. Sy et al., 2005).

[9] This section is based on Werder and Maedche (2017a)

However, many of the studied emotional cues (e.g. facial expressions or speech differences) are not available in virtual settings, such as OSS development, as they often rely on written communication (Kahai et al., 2007). Therefore, analyzing textual information exchanged during collaborative work within OSS development teams is particularly interesting for the analysis of emotional contagion. In addition, leaders in an OSS development context lack a formal authority structure. Their authority is the result of an emergent process within the project team (Pescosolido, 2002) and therefore, limits the likelihood of mimicking processes caused my formal hierarchy and authority (Charlier et al., 2016; Yoo & Alavi, 2004).

The increasing open access to big data sources enables scholars to rely on secondary data for the evaluation of theoretical models (e.g. George et al., 2014). While empirical studies on virtual teams in general (e.g. Peters and Karren, 2009) and OSS teams in particular (e.g. Stewart & Gosain, 2006) capture only perceptions of the team, big data from OSS repositories allows us to analyze productive data that was not created for research purposes. On the one hand, prior studies focusing on virtual teams investigate concepts such as trust (Jarvenpaa, Shaw, & Staples, 2004; Peters & Karren, 2009; Staples & Webster, 2008), empowerment (Kirkman et al., 2004), conflict (Kankanhalli, Tan, & Wei, 2007), and collaboration (Bjørn & Ngwenyama, 2009; Kahai et al., 2007; Peters & Manz, 2007). On the other hand, OSS research investigates team diversity (Vasilescu, Posnett, et al., 2015), social structures (Bird, Pattison, D'Souza, Filkov, & Devanbu, 2008), social influence (P. V. Singh & Phelps, 2013), or innovation (Liu, Hull, & Hung, 2016). While few exceptions investigate emotions in OSS development projects (Guzman, Azócar, & Li, 2014; Jurado & Rodriguez, 2015), emotions in virtual teams received little attention in prior research. When investigating the role of emotions within teams, virtual teams lack emotional cues based on face-to-face interactions (Derks, Fischer, & Bos, 2008). However, prior empirical studies often rely on physical presence when investigating team emotions (Homan et al., 2016; van Kleef et al., 2009). Due to the importance of OSS within the software industry (B. Fitzgerald & Feller, 2001) and its specific characteristics of virtual work and emergent leadership, more research in needed to better understand the influencing role of emotions (van Kleef, 2009).

Given the importance of emergent leaders as managers of team emotions (Pescosolido, 2002), we investigate the contagious effect of team leader's emotional displays on the team's emotional displays in the context of OSS development. These teams collaborate virtually and therefore, do not receive emotional cues based on physical presence. However, many of the existing explanations for emotional contagion rely on physical presence (e.g. afferent feedback, or mimicry) (Barsade, 2002). Given the lack of physical emotional cues in virtual teams, we rely on the Emotions-As-Social-Information (EASI) model in order to investigate the contagious effect (van Kleef, 2009). The literature presents mixed results for the positive and negative effects of emotional contagion (Collins et al., 2013). Therefore, we lack an understanding of the emotional

contagion effect in OSS teams in general, and of the conditional effect of team collaboration in particular. Hence, we investigate positive and negative emotions within the teams and extend existing models by two conditional factors, team collaboration structure and team collaboration function, following a structural functionalism view from sociology (Durkheim, 1897; Macionis, 2013). We suggest that both properties of team collaboration influence the effect of emotional contagion from the leader towards the team. Thus, we address the following research questions:

- *What is the effect of emergent OSS leader's emotional display on the OSS team's emotional display and does this effect differ between positive and negative emotions?*

- *What is the moderating effect of collaboration structure and collaboration function on emotional contagion?*

These questions guide the development of our research model. We empirically test our derived research model using 999 projects from the open source software repository GitHub10. Therefore, we rely on secondary data, benefitting from recent advancements suggesting the use of big data sources in management research, team research and information systems research (G. George et al., 2014; S. W. J. Kozlowski et al., 2015; Müller et al., 2016). Our emotional cues are based on textual and numeric data. In contrast to primary data, this data was not created for the sole purpose of scientific analysis. Therefore, we do not rely on perceptions, but on real comments by OSS teams. The study compares three models: the first model tests the direct effect of the control variables, the second model includes the main effects of the independent variables in the test, and the third model adds the interaction effects.

Our research extends the literature on emergent leadership and emotional contagion by two boundary conditions: team collaboration structure and team collaboration function. We find that team collaboration structure interacts with positive and negative team leader's emotional displays. Therefore, our study also contributes to empirical research related to the contagious effect of team leader positive and negative emotions on the team emotions (Gooty, Connelly, Griffith, & Gupta, 2010). Drawing on the EASI model, we broaden the application of the model towards the context of virtual teams and OSS development. While prior literature often focused on the investigation of either positive or negative emotions, presenting mixed results, our study provides further clarification. We find that in virtual teams, team members broadly adopt both emotional tendencies of their leaders, an effect that increases for teams with denser collaboration structure. The research contributes to the important role of emotions for software development team leaders

[10] GitHub is an open source software repository, as it allows developers to host their software source code on the internet and benefit from GitHub's version control system. The service is accessible through the website https://github.com.

and managers alike. Practitioners learn about the importance of team collaboration structure when regulating their emotional expressions to others.

4.2.2. Theoretical Background and Hypothesis Development

Collaboration in virtual software development teams is different from other work teams. Work teams often rely on face-to-face interaction. For example in Scrum, the most popular development method in practice with applications beyond agile software development (R. G. Cooper & Sommer, 2016), daily face-to-face interaction is explicitly embedded through daily stand-up meetings (Schwaber & Sutherland, 2013). Occasional face-to-face meetings are known to increase the effect of team empowerment (Kirkman et al., 2004). Yet, in virtual teams all collaboration is technology mediated (Derks et al., 2008; Faraj, Kudaravalli, & Wasko, 2015). While different collaboration means are available, OSS teams tend to use the asynchronous text-based communication embedded in the software repository. The team communication focuses on textual cues and therefore, lacks many additional informational cues, such as facial expression, social context, or change of speech tone (Barsade, 2002; Faraj et al., 2015). Particularly, researchers investigating emotional contagion tend to focus on these additional non-verbal cues for measuring emotional expression (Barsade, 2002; van Kleef et al., 2009). These cues or affordances can be interpreted by an observer in different ways and through different mechanisms (van Kleef, 2009). However, little research investigates conditions of emotional contagion in virtual teams in general and virtual teams leader-follower relationship in particular (Barsade, 2002). For example, individual and group positive affect enhances the individual's information seeking behavior (Neff et al., 2014).

As a result of the participant's autonomous and independent behavior, OSS teams lack a formal authority structure (Maturana & Varela, 1980; Vidgen & Wang, 2009). The team's structure is an emergent result of a self-organizing process (Curşeu, 2006; Goldstein, 1994). This may suggest that self-organization results from a lack of control. To the contrary, the structure is a result of the agents adaptation leading to new configurations of the system (McCarthy et al., 2006). Particularly, the concept of leadership provides new challenges towards team collaboration within OSS teams. The leadership literature suggests a direct effect of the leader's emotions on the followers performance (G. Wang & Seibert, 2015). Yet, the primitive processes and afferent feedback mechanisms leading to emotional mimicking could play a different role in a self-selected environment in contrast to a postulated environment (Barsade, 2002; van Kleef, 2009). In addition, primitive processes and afferent feedback are not applicable to virtual environments. Little research has investigated the effect of emotional contagion in virtual teams and possible conditional factors. While we understand that communication is an important factor for OSS team effectiveness (Stewart & Gosain, 2006), we have yet to understand the influence and conditions of leader's emotions on the team's emotions in general, and during virtual team collaboration in particular (Barsade, 2002).

Research Model for Emotional OSS Team Leadership

For our research model, we draw on emotional contagion theory (Hatfield, Cacioppo, & Rapson, 1993) and social network theory (Granovetter, 1985; Travers & Milgram, 1969). Our model hypothesizes that emotions are contagious (Barsade, 2002) and a team leader can influence the emotions of the team by providing corresponding emotional displays. Following social network theory, we suggest that this contagious effect is stronger for virtual teams that collaborate more frequent and more diverse. We explain this by understanding emotional display as social information (cf. EASI) that distribute within the team (van Kleef, 2009).

In order to describe individual emotions and team emotions, researchers refer to different models (Parrott, 2001; Plutchik, 1980; Russell, 1980). In general, the literature distinguishes discrete emotional models from dimensional models. While discrete models list individual emotions (e.g. Plutchik & Kellerman, 1980), such as joy and anger, dimensional models focus on dimensions that apply to discrete emotions to differing extents (e.g. Russell, 1980), such as valence and arousal. Following prior studies we use eight discrete emotions as suggested by Plutchik's wheel of emotions (Jurado & Rodriguez, 2015; Plutchik, 1980). Therefore, we define positive emotional displays as the extent to which a team or team leader reflects joy, trust, surprise and anticipation. In a similar vein, we define negative emotional displays as the extent to which a team or team leader reflects sadness, disgust, fear and anger. In our model, we distinguish between the team leader's emotions and the team members' emotions. Since the team leader has no formal position within OSS development, we define the leader based on six characteristics (Bird et al., 2008). A leader delivers outstanding quality, quantity and broad commits to a project, has a high social status, is in the inner circle of the development team, has extensive and broad discussions, and comments a lot on the project.

Researchers have investigated the role of social cues for technology-mediated communication and non-verbal communication (Kearney et al., 2009; Sproull & Kiesler, 1986). Such cues are predominantly relevant for virtual teams, as they lack many of the otherwise available social cues indicated by facial expression or body language. Therefore, we investigate the structure and function of team interaction through collaboration. The concepts of social structure and social function have been introduced through structural functionalism in sociology (Durkheim, 1897; Macionis, 2013). While social structure is often referred to as emergent patterns related to individual's actions, social function is referred to as actions that contribute to such structures' maintenance (Macionis, 2013). We adopt these concepts to team collaboration and understand a team's collaboration structure as the extent of collaboration density (Yang & Tang, 2004). We understand a team's collaboration function as the extent of collaboration messages and feedback (Kahai et al., 2007).

Hypotheses development for main effects

Emotional contagion theory serves as an explanation for the spread of emotions between two individuals (Hatfield et al., 1993). This research stream attracted a wide range of scholarly interest. While initially, research was limited to dyadic relationships, the topic gained interest by team-level researchers (e.g. Barsade, 2002; Kelly and Barsade, 2001; Sy et al., 2005). The theory suggests that emotions are contagious. While many scholars agree with this idea, three competing underlying explanations related to the process evolved (Barsade, 2002). First, primitive, non-conscious mimicry can be the cause of spontaneous behavior, e.g. in the form of facial expression or changes in speech tone. Second, afferent feedback and mimicking behavior precedes the following emotional reaction, caused by physiological feedback of muscles. Third, emotions reflect social information and hence, play a role in the understanding of a situation that causes the individual to respond in a socially acceptable manner. The latter is supported by the social functioning of emotions and their informative role during social events (Keltner & Haidt, 1999). Building on this idea, the emotions as social information (EASI) model has been developed (van Kleef, 2009). EASI suggests that observer's affective reactions and observer's inferences explain the relationship between an emotional expression and the observer's behavior. Further, social relational factors and observer's information processing are expected to moderate this relationship.

In line with prior literature, this study understands emotions as social information (Homan et al., 2016; Keltner & Haidt, 1999; van Kleef, 2009). In the context of virtual teams, this view is most applicable. Given the technology-mediated collaboration in virtual teams, their communication lacks emotional expressions, such as facial expressions and changes in speech tone, that are part of prior studies (e.g. Faraj et al., 2015). Therefore, all emotional information is text-based and requires interpretation by the team members. Despite the lack of physical cues, text-based communication is equally emotional as people develop compensatory mechanisms, e.g. making emotions more explicit (Derks et al., 2008). Prior research on leadership suggests that leaders should be aware of this effect and use emotions in order to cause a desired response by their followers (Humphrey et al., 2008; Rajah et al., 2011). Leaders of virtual teams then rely on the technology-mediate communication to express their emotions. Therefore, team members will use this information and, similar to the primitive contagion and afferent feedback mechanisms (Kelly & Barsade, 2001), mimic the emotional response. This allows the team member to share the emotions of the team leader and enable the person to blend in and become more alike other team members (G. Wang & Seibert, 2015).

Some researchers suggest different effects for positive and negative emotions. For example, prior research argues for a positive effect of positive psychology (e.g. Gable & Haidt, 2005), particularly in the context of leadership (e.g. Avolio et al., 2004). Others suggest improved performance for negative emotions, particularly in the context of negotiation (van Kleef et al., 2009). Following our earlier argumentation, we

suggest emotions in virtual teams are used as social information. As a result, team members are more likely to conform to existing emotions presented by the leader in order to move towards a group consensus and gain approval of others (Cialdini & Trost, 1998). Therefore, the team converges on their emotional experience and form a consensus (Barsade & Knight, 2015). An opposing emotional reaction is likely to create dissonance that alienates the individual (Matz & Wood, 2005). Hence, the team member will rather conform than confront to the emotion. We formulate two hypotheses for emotional contagion in virtual teams:

H1a: *A positive association exists between the OSS team leader's positive emotional display and the OSS team members' positive emotional display.*

H1b: *A positive association exists between the OSS team leader's negative emotional display and the OSS team members' negative emotional display.*

Hypotheses development for interaction effects

Our first research question relates to the direct contagious effect of emotions as social information. Emotions from team leaders influence the team member's emotions in a virtual environment. Prior empirical research related to software development teams benefits from a social perspective and identifies interesting findings through in-depth analysis of individual agent's communication structures that a broader quantitative approached would not be able to identify (Yang & Tang, 2004). Empirical evidence from leadership research suggests the importance of a functional view towards team communication in the context of virtual teams (Charlier et al., 2016). Hence, our model includes the structural-functional view through two additional constructs: team collaboration structure and team collaboration function.

Team collaboration structure: When contagion takes place, team collaboration structure is an important element for the distribution of emotions (de Choudhury & Counts, 2013). Therefore, we suggest it is a relevant factor for the relationship between team leader emotional displays and team member emotional displays. Social network theory suggests density as an important factor for the exchange of information and knowledge within social networks (Granovetter, 1985). For instance, the density of a network can affect the reach of an organizational actor when communicating corporate tweets (Helms & Werder, 2013), or when assessing the applicability of social media for various organizational functions (Werder, Helms, & Jansen, 2014). Therefore, actors in a dense network benefit from more first-level connections and even more second-level connections and therefore, enhance the effect of the small-world property of such networks (Watts & Strogatz, 1998). Research from social psychology presents empirical evidence for the emotional nature of connections between actors (Magee & Tiedens, 2006). Therefore, we suggest that team collaboration

structure is a boundary object for the contagious effect of positive and negative emotions (G. Wang & Seibert, 2015). Consequently, we hypothesize that team collaboration structure strengthens the effects of both relationships from team leader's emotional displays on the team's emotional displays:

H2a: *The association between the OSS team leader's positive emotional display and the OSS team members' positive emotional display is stronger with increased density of the team collaboration structure.*

H2b: *The association between the OSS team leader's negative emotional display and the OSS team members' negative emotional display is stronger with increased density of the team collaboration structure.*

Team collaboration function: The functional role of team collaboration is more difficult given the ambivalent results. Some researchers suggest that the frequency of information can influence the observers perception of the emotion and inferences (Kacmar, Witt, Zivnuska, & Gully, 2003; van Kleef, 2009). Particularly, leaders frequently use emotions in order to influence others affective states (Kelly & Barsade, 2001; G. Wang & Seibert, 2015). Studies on virtual game teams suggest the benefit of non-verbal communication on team performance (Leavitt, Keegan, & Clark, 2016). On an individual-level, the frequency of positive emotions has an effect on the individual's perception of positive affect (e.g. happiness) (Lyubomirsky, King, & Diener, 2005). Yet, studies supporting the importance of leader's frequent emotional display on an individual-level, find non-significant results on the group level (G. Wang & Seibert, 2015). In addition, while some researchers may suggest the negative impact of frequent information caused by information overload (Eppler & Mengis, 2004), prior research on development teams investigates this concern for information stemming from external sources (Sawyer, Guinan, & Cooprider, 2010). The study suggests communication profiles that benefit from shielding the team against an external information overload. While the prior results are mixed, we concur with the idea of teams maintaining team viability through a reinforcement process (Niedenthal & Brauer, 2012). Frequent exposure to positive or negative emotions would reinforce the contagious effect from the team leader. Therefore, we suggest frequency to strengthen the effects of both relationships from team leader's emotional display on the team's emotional display:

H3a: *The association between the team leader's positive emotional display and the team members' positive emotional display is stronger with increased frequency of the team collaboration function.*

H3b: *The association between the team leader's negative emotional display and the team members' negative emotional display is stronger with increased frequency of the team collaboration function.*

Figure 17 depicts the research model suggesting a direct positive effect of emotional contagion from the OSS team leader on the OSS team. The effect applies to positive and negative emotional display, and will be stronger for cases with dense OSS team collaboration structure and high OSS team collaboration function. Additionally, we include important control variables, such as lines of code, project team size, project productivity, and project duration.

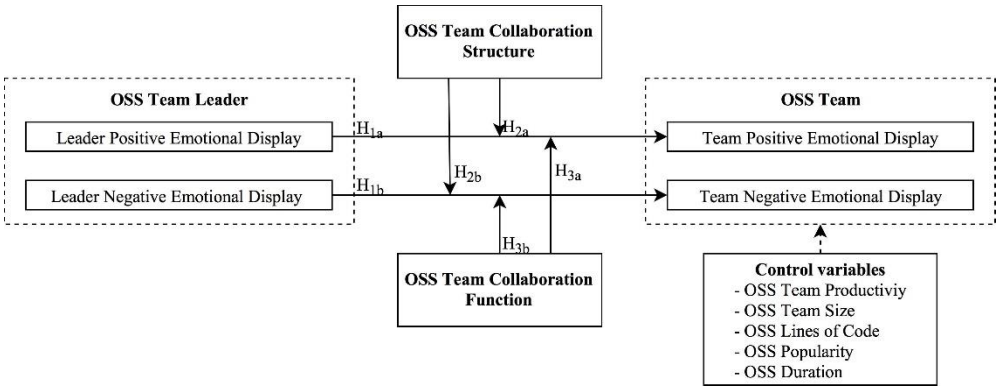

Figure 17 – Research Model for Emotional Contagion in OSS Teams.

4.2.3. Research Method

Team research often opposes challenges related to the collection of appropriate sample sizes and representative responses from multiple team members (S. W. J. Kozlowski et al., 2015). Hence, we opt for secondary data including 999 projects within the domain of software development teams[11]. Secondary data is often of high quality (Blumberg, Cooper, & Schindler, 2011) and allows us to tap into a larger data source. While examples in other contexts benefitting from Big Data have been mentioned (S. W. J. Kozlowski et al., 2015), we investigate OSS development teams. We mine an OSS repository as a Big Data source storing textual and numerical information. In order to extract emotional cues that are representative for the teams and their leader's emotions, we mine textual information exchanged during collaborative work within the OSS development projects.

Specifically, the GitHub OSS repository allows development teams to collaborate online and maintain a single code repository for each software project. Key features are committing code, requesting changes to be committed by code maintainers (pull request), and raising issues. For the analysis of emotional cues based on textual information, OSS teams are particularly suitable. OSS teams work in virtual settings and therefore, embed the documentation and communication of project decisions within an online repository. This gives us a unique opportunity to investigate the role of emotional displays in work teams. While the data is generated in GitHub, we use Ghtorrent as our main data source. Given the access restrictions of

[11] The extracted data can be accessed at https://madata.bib.uni-mannheim.de/189/.

GitHub, we complement the information from Ghtorrent through data from the GitHub API. Ghtorrent is a software repository for research purposes that archives GitHub data in a complex data format12. The Ghtorrent project hosts a history of multiple terabytes of GitHub data. Therefore, Ghtorrent provides researchers with a unique data source in order to answer research questions that are challenging to answer without this valuable data source.

Sampling and Data Collection

Overall, we follow an established Big Data analysis process as we see great value in the information stored in the OSS repository. Hence, our procedure includes five steps (Blumberg et al., 2011). First, we sample our data from the software repository. Second, we explore the data and investigate potential relationships. Third, we modify and transform the data, where needed and plausible, e.g. through the creation of new factors. Fourth, we develop a model that explains the relationships. Fifth, we test the model's accuracy.

We sample the data and define clear inclusion and exclusion criteria before describing the sampling process. We include projects with more than 50 comments in total. Prior studies suggest that this criteria also excludes most personal projects (Kalliamvakou et al., 2014). The programming language and starting year of a project do not inform any inclusion criteria. We exclude projects that are either inactive, incomplete or redundant. Following prior studies, we consider projects inactive when they have less than 2 commits per month, less than 24 commits in total, less than 6 participating developers and less than 6 months of activity (Vasilescu et al., 2015). A project is incomplete when we could not access all commits or when >= 25% of committer information where inaccessible. A similar rule applied to the absence of >= 25% of LOC (Lines of Code) as a frequent performance indicator. In addition, we remove duplicates from the dataset. We extract the data from Ghtorrent and aggregated them on a monthly basis (Guzman et al., 2014; Vasilescu et al., 2015). In order to draw logical conclusions on the project team-level, we aggregate and summarize the data per project. Applying the above mentioned selection criteria present us with 1.121 projects to include in our data analysis.

Following a missing values analysis and outlier's detection, we reduce the data to 999 complete cases for further analysis. Outliers are detected using post hoc analysis for both equations, predicting team positive and negative emotional displays. We calculate Mahalanobis distance (Mahalanobis, 1936), Cook's distance (Cook, 1977), and Leverage Distance (Läuter, 1985). When two of three tests suggested a case as an outlier, the case was removed from the data. Table 32 presents descriptive statistics of this dataset.

[12] http://ghtorrent.org/ - ghtorrent stores its JSON data in a MongoDB and its meta-data in a MySQL database.

Variables

All variables are grounded in prior literature. The data is either extracted from the databases (e.g. programming language), or processes the results thereof (e.g. number of developers or team emotions). We sample all variables on a monthly basis for the selected projects. Consequently, the variables take either an average score (e.g. in the case of the developers), or a sum score (in the case of the duration), depending on the plausible aggregation of a variable.

Table 32 – Descriptive Statistics of Final Dataset (N=999).

Variable	Unit	Aggregation	Average	SD	Min	Max
Team Positive Emotional Display	Team Emotions	(Average/Month)	0.029	0.953	-1.273	4.924
Team Negative Emotional Display	Team Emotions	(Average/Month)	0.026	0.950	-1.291	6.095
Team Leader Positive Emotional Display	Team Leader Emotions	(Average/Month)	-0.032	0.840	-1.687	4.682
Team Leader Negative Emotional Display	Team Leader Emotions	(Average/Month)	-0.044	0.839	-1.476	4.322
Productivity	Merged and closed pull requests & closed issues	(Average/Month)	-0.085	0.561	-0.507	3.790
Team Size	Developers	(Average/Month)	42.275	68.885	5.000	748.842
Number of Comments	Comments	(Average/Month)	8.861	8.936	0.025	58.318
Density	Edges	(Average/Month)	0.221	0.188	0.001	0.950
Lines of Code	Lines of Code	(Average/Month)	15210	34687	-61623	299516
Subscribers	Subscribers	(Average/Month)	1521	3830	0	32503
Duration	Month	(Sum)	21.780	16.712	1.000	83.000

Emotions play an important role within teams and organizations (Barsade & Gibson, 2007; Barsade & Knight, 2015). We distinguish between emotions shared across the team and emotions held by the team leader. For both, emotional scores are calculated for individual team members as a mean value and on for the project team as a mean function (Barsade & Knight, 2015). A developer is considered a team member when he/she commits to the projects. We calculated the scores on a monthly basis from the extracted comments related to issues and pull requests. While different packages exist (Appendix A), we rely on the Syuzhet R Package to extract scores for eight discrete emotions based on Plutchik's wheel of emotions and store these on a monthly basis for each project (Jockers, 2016; Plutchik, 1980). Equation 1.1 shows the emotion as calculated for each team member and Equation 1.2 shows the calculation for the entire team. We measure joy, trust, surprise and anticipation to indicate positive emotional displays (Cronbach's α = .95) and we measure sadness, disgust, fear and anger to indicate negative emotional displays (Cronbach's α = .90). Forming the factors using regression method leads to values that include negative scores.

(1.1) $Team\ member\ emotion\ display\ (TME) = \frac{\sum_{i=1}^{C} Emotion(Comments[i])}{C}$; *Where C is the number of comments.*

(1.2) $Team\ emotion\ display = \frac{\sum_{i=1}^{S} Emotion(TME[i])}{S}$ *; Where S is the project team size.*

Leadership has been emphasized by prior research within teams and organizations. We identify the team leader of an OSS team based on established characteristics (Bird et al., 2008). Hence, a team leader is identified depending on the number of commits, eigenvector centrality in the social graph, betweenness centralization in the social graph, degree of team members in the graph, and whether they were the first committer within the project. Centrality, betweenness, and degree are indicators for the team leader's influence within the team. This is consistent with prior research that suggests emergent leaders send more messages compared to their team members (Yoo & Alavi, 2004). Analogue to team member emotion display, Equation 1.3 presents the calculation of team leader emotion display. We measure the team leader positive emotional displays (Cronbach's α = .86) and team leader negative emotional displays (Cronbach's α = .78) analogue to the team emotional display. Negative scores result from the factor formulation.

(1.3) $Team\ leader\ emotion\ display = \frac{\sum_{i=1}^{C} Emotion(Comments[i])}{C}$.

Team collaboration structure: The connections between individuals and their communication influence knowledge flows and social interactions within project teams. Here, the developers and contributors of the project are considered the nodes of the network. Any written communication exchange, e.g. through the commenting on each other's issues, commits or pull requests indicated a communications exchange that we consider the edges of each projects social network. Analogue to Totterdell et al. (2004), we measure project network density as the division of the sum of edges over the sum of potential ties within the network based on the number of nodes (see Equation 1.4). In contrast to degree, this study relies on the density as a measure for project team's collaboration structure, as it is agnostic to the project team size. We measured the scores on a monthly basis and formed a monthly average per project.

(1.4.) $Team\ Collaboration\ Structure = \frac{2*E}{S*(S-1)}$ *; Where E is the number of edges in the graph.*

Team collaboration function: The information exchanged within the context of emotional analysis are social information (van Kleef, 2009). The frequency of information exchanged within the team can influence the observers perception of emotion and therefore the observers inferences (Kacmar et al., 2003). Analogue to prior studies, we measure the frequency of information exchanged through the number of comments on issues, commits and pull requests within a team (Yoo & Alavi, 2004). Equation 1.5 shows the calculation of the collaboration function. We measured the scores on a monthly basis and formed a monthly average per project.

(1.5.) $Team\ Collaboration\ Function = \sum_{i=1}^{S}(C[i])$.

Control variables: We include important co-variates in our model that could influence the dependent variable, such as lines of code, project team size, project productivity, and project duration. While demographics, such as age, previous job experience, and experience in the current role are relevant for non-virtual leadership (Walter & Scheibe, 2013), these information are suppressed in virtual teams and therefore, do not play a role for the emergence of leadership in virtual teams (Yoo & Alavi, 2004). The project team size is often considered a team characteristic that influences the team processes (de O. Melo et al., 2013; Wageman et al., 2005). Here, we measure the project team size excluding the team leader. The popularity of a project is another characteristic we control for (Cosentino, Luis, & Cabot, 2016) by measuring the number of subscribers[13]. Prior studies on leadership suggest an effect between emotions and team productivity (Burke et al., 2006). Therefore, we control for team productivity (Cronbach's $\alpha = .90$) measured by the number of merged and closed pull requests, and the number of issues closed. The project duration captures the temporal dimension of the projects, that has been identified as an influential factor when investigating emergent leadership (Yoo & Alavi, 2004).

Prior studies suggest the measurement of quality by the number of bugs (Maruping, Zhang, et al., 2009). For the identification of bugs, user tagging has been suggested as a successful mechanism in prior studies (Vasilescu, Yu, et al., 2015). Hence, we analyze the comments and consider the following words to indicate the existence of bugs: "bug, fix, flaw, error, failure, fault, defect, glitch, conflict and reproduce". Yet, the data was incomplete and missing values exceeded 70% of the cases. Therefore, we did not use this measure in our analysis. However, we control for Lines of Code as an often-used quality indicator. While this measure is sometimes used with misconceptions (Rosenberg, 1997), it is a predictor of software faults (Radjenović, Heričko, Torkar, & Živkovič, 2013). Team dispersion can be an influential factor on team processes within software development teams (Ågerfalk et al., 2005). Using an optional indicator in conjunction with open street maps API (Haklay & Weber, 2008), less than 10% of the projects included sufficient (i.e. more than 70%) geographic information of their members. Given the small sample and incomplete data, we are unable to account for this variable.

Measurement Validation

We formed the factors using the regression method in order to preserve the correlations. All factors receive sufficient KMO scores >0.7 (Kaiser–Meyer–Olkin) indicating an appropriate sample size (Hutcheson & Sofroniou, 1999). Using Kolmogorov-Smirnov (KS) and Shapiro-Wilk tests for normality we find that all variables significantly differ from normality ($p<.001$). A visual inspection suggests a power-law distribution. Testing all variables for a power-law distribution (Clauset, Shalizi, & Newman, 2009) indicates

[13] We also measured the number of people that watch a project and the number of forks that have been created. Due to incomplete data, limited responses, and because of our missing value analysis, we excluded these single items from further analysis.

that this distribution is most likely (Table 33). Except for team leader negative emotions and project duration, all variables, pass >90% of the KS tests (Lai, 2016). For team leader negative emotions and project duration, a power law distribution is still the most likely distribution in comparison to exponential and lognormal distributions. During our analysis, we conduct different regressions that do not assume normality of the data. We present the correlations in Table 34. We use the Fornell-Larcker criterion to assess the discriminant validity (Fornell & Larcker, 1981). All diagonal entries exceed the horizontal and vertical entries in the correlation matrix and therefore, pass this test.

Table 33 – Psychometric Quality Assessment.

Construct	KS Score	α	xmin	Cronbach's alpha
Team Positive	0.98	4.859	0.097	.953
Team Negative	0.99	5.059	0.147	.896
Leader Positive	0.99	3.594	0.120	.863
Leader Negative	0.60	2.529	0.101	.783
Productivity	0.96	2.484	0.072	.904
Team Size	0.99	3.117	5.667	Single Item
Collaboration Function	0.96	4.405	0.593	Single Item
Collaboration Structure	0.98	8.497	0.065	Index
Quality	0.95	2.104	43.400	Single Item
Popularity	0.99	4.203	9.000	Single Item
Project Duration	0.89	8.117	1.000	Single Item

Note: KS Score = Number of significant KS test statistics over the number of tests performed (2500). A value >0.90 indicates a power-law distribution.

Table 34 – Intercorrelation Matrix.

		1	2	3	4	5	6	7	8	9	10	11
1	TED Positive	.882										
2	TED Negative	.752**	.768									
3	TLE Positive	.372**	.190**	.716								
4	TLE Negative	.115**	.307**	.328**	.604							
5	Productivity	-.104**	-.066*	-0.002	.091**	.890						
6	Team Size	-.312**	-.284**	-0.008	0.046	.440**	-					
7	CF	.557**	.546**	.053*	0.040	0.011	-.254**	-				
8	CS	.677**	.608**	.061*	-.070*	-.237**	-.433**	.540**	-			
9	Lines of Code	-0.025	-0.019	-.055*	-.065*	.241**	.071*	.072*	-.087**	-		
10	Popularity	-.185**	-.150**	0.025	.100**	.389**	.567**	-.123**	-.307**	0.015	-	
11	Duration	-.375**	-.341**	.057*	.117**	.292**	.444**	-.326**	-.464**	-0.013	.291**	-

Note: TLE = Team Leader Emotional Display; TED = Team Emotional Display; CF = Team Collaboration Function; CS = Team Collaboration Structure. **.p at 0.01 level (1-tailed); *. p 0.05 level (1-tailed); Absolute values for the correlation coefficient are significant above 0.046 (0.072) on a 5% (1%) level, Bold values represent the average variance extracted (AVE),

4.2.4. Data Analysis and Results

In order to estimate our model, we employ multivariate regression analysis using ordinary least squares (OLS) and seemingly unrelated regression (SUR). As we formulate two central models with potential correlations of the errors, we conduct a seemingly unrelated regression (SUR) analysis (Zellner, 1962). When testing our directional hypothesis, we employ one-tailed tests (Blumberg et al., 2011; Pelled, Eisenhardt, & Xin, 1999). An analysis of multi-collinearity of all variables indicates no sign of concern14. We use the centered scores of all variables for interpretation purposes in Table 35. We confirm the assumption of independent error using Durbin-Watson test. We find good Durbin-Watson values for the nested models explaining team positive emotions (1.919) and the nested models explaining team negative emotions (1.934). Visual analysis of the histogram and P-P plots confirm normally distributed errors. Scatterplot of standardized residuals suggests homogeneity of variance and linearity. We estimate three models, controls-only, main effects, and the interactions (Table 35).

First, we analyze the controls only model to assess whether these have a significant effect on the dependent variables, team positive emotional display (Model 1a) and team negative emotional display (Model 1b). Results support prior expectations identifying project productivity, project team size and project duration as significant covariates of both, team positive emotional display and team negative emotional display. Non-significant effects of some controls might be attributed to the fact that no model has assessed their effects simultaneously. Second, we add main effects of the independent variables and moderators. When estimating team emotions, we add the corresponding team leader emotional display, team collaboration function and team collaboration structure as main effects. Model 2a and model 2b show significant main effects. Leader positive emotional display (β =.351, p <.001), team collaboration function (β =.282, p <.001), and team collaboration structure (β =.486, p <.001) have strong effects on the team positive emotional display. In a similar vein, leader negative emotional display (β =.352, p <.001), team collaboration function (β =.293, p <.001), and team collaboration structure (β =.452, p <.001) have strong effects on the team negative emotional display.

Third, we evaluate additional effects of interactions between team collaboration function and the corresponding leader emotional display, as well as the team collaboration structure and the corresponding leader emotional display (positive or negative). We find a significant interaction effects for leader emotional display with the team's collaboration structure for both equations. When predicting team positive emotional display, we find an interaction effect of team leader positive emotional display and team collaboration structure (β =.126, p <.001). Also, we find an interaction effect of team leader negative emotional display

[14] Variables VIF scores for all model are <1.95. A concerning threshold is >10 (Bowerman & O'connell, 1990, Myers 1990). The tolerance scores are >0.5, while values <0.2 indicates potential problems (Menard 1995)

and team collaboration structure when predicting team negative emotional display (β =.190, p <.001. There is no significant interaction effect between team collaboration function and the team leader's emotional display. However, the main effects remain significant in model 3a and model 3b.

Table 35 – Seemingly Unrelated Regression.

Variable	Team positive emotional display			Team negative emotional display		
	Model 1a (controls)	Model 2a (main effects)	Model 3a (interaction)	Model 1b (controls)	Model 2b (main effects)	Model 3b (interaction)
TLE positive		.351 (.019) ***	.328 (.020) ***		-	-
TLE negative		-	-		.352 (.021) ***	.354 (.021) ***
TLE positive x team CF			.059 (.032)			-
TLE positive x team CS			.126 (.022) ***			-
TLE negative x team CF			-			.041 (.036)
TLE negative x team CS			-			.190 (.025) ***
Team CF		.282 (.028) ***	.278 (.028) ***		.293 (.030) ***	.263 (.029) ***
Team CS		.486 (.026) ***	.486 (.026) ***		.452 (.028) ***	.496 (.028) ***
Productivity	.151 (.058) **	.067(.040)	.059 (.039)	.194 (.058) ***	.065 (.043)	.078 (.041)
Team Size	-.481 (.091) ***	-.042 (.063)	-.055 (.062)	-.494 (.092) ***	-.054 (.068)	-.071 (.065)
Quality	-.067 (.056)	.016 (.038)	.025 (.038)	-.066 (.057)	.026 (.042)	.032 (.040)
Popularity	-.024 (.063)	.016 (.043)	.028 (.042)	.014 (.064)	.024 (.047)	.054 (.044)
Duration	-.293 (.031) ***	-.088 (.023) ***	-.085 (.022) ***	-.271 (.032) ***	-.084 (.025) ***	-.078 (.023) ***
Constant	-.006 (.029)	.041 (.019) *	.031 (.019)	-.004 (.029)	.043 (.021) *	.053 (.020) **
Number of observations	999	999	999	999	999	999
R-square	.173	.624	.638	.149	.551	.593
R-square increase		.451 ***	.014 ***		.402 ***	.042 ***
Adjusted R-square	.169	.621	.635	.144	.548	.589

Note: TLE = Team Leader Emotional Display; CF = Team Collaboration Function; CS = Team Collaboration Structure. The table reports standardized β coefficients and the standardized errors. *Significance level of 5%, **Significance level of 1%, *** Significance level of 0.1%. R-square increase significance was tested using Likelihood Ratio Test for nested Models.

When further analyzing the interaction effects, we can employ the slope difference test comparing slopes of different values of the interaction variable (Aiken & West, 1991). Consequently, we can better understand the nature of the interaction effects. Table 36 presents the results for both interaction effects. The effect of team leader's positive emotional display on the team positive emotional display increases with higher values of team collaboration structure. While less dense teams have a rather flat slope (.234, p<.001), dense teams are best predicted using a steep slope (.478, p<.001). In a similar vein, the effect of team leader's negative emotional display on the team negative emotional display increase with higher values of team collaboration

structure. While less dense teams have a flat slope (.164, p<.001), dense teams follow a steep slope (.614, p<.001). An overview of all tested hypothesis presents Table 37.

Table 36 – Slope Difference Test.

Interaction term	Value of interaction variable			
	Mean-1SD	Mean	Mean+1SD	DV
TLE positive x team CF	Not analyzed due to non-significant interaction effect.			
TLE positive x team CS	.234 (.053) ***	.356 (.025) ***	.478 (.061) ***	Team positive Emotions
TLE negative x team CF	Not analyzed due to non-significant interaction effect.			
TLE negative x team CS	.164 (.049) ***	.389 (.035) ***	.614 (.066) ***	Team negative emotions

Note: SD = standard deviation; TLE = Team Leader Emotional Display; CF = Team Collaboration Function; CS = Team Collaboration Structure; *Significance level of 5%, **Significance level of 1%, *** Significance level of 0.1%; Values are the conditional effect scores, values in parentheses present the standard errors.

Table 37 – Overview of Results.

Hypothesis	Relationship	Results
H1a	TLE positive -> TED positive	Supported
H1b	TLE negative -> TED negative	Supported
H2a	Moderation of CS on TLE positive -> TED positive	Supported
H2b	Moderation of CS on TLE negative -> TED negative	Supported
H3a	Moderation of CF on TLE positive -> TED positive	Not supported
H3b	Moderation of CF on TLE negative -> TED negative	Not supported

Note: TLE = Team Leader Emotional Display; TED = Team Emotional Display; CF = Team Collaboration Function; CS = Team Collaboration Structure

Additional Tests

We calculate the Likelihood ratio test statistic for SUR and the corresponding OLS models. For models 1a/b and 2a/b, results indicate no significant differences. We find a significant difference for models 3a/b (χ^2 = 9.705; p<.01), providing further support for the use of SUR analysis. Durbin-Wu-Hausmann test is used as a robustness check, indicating whether variables are exogenous[15]. Results suggest that variables are exogenous, providing further support for the use of SUR analysis.

A model test including the three-way interactions suggests a three-way interaction effect when predicting team negative emotional display (β =-.048, p <.1). While, the SUR model does not offer significantly better predictions, as assessed using likelihood ratio test, it suggests negative effects of extensive collaboration function within dense collaboration structure. We also test the cross-emotional effects of negative leader emotional display on team positive emotional display and vice versa. While the SUR model does not significantly improve, we find significant effects. When predicting team positive emotional display, leader

[15] Test statistic: hausman=10.997, df=12; p=.529

negative emotional display (β =.054, p <.05) and the interaction of leader negative emotional display with density (β =.090, p <.01) have an effect. Contrary, team leader positive emotional display has little effect on the team negative emotional display (β =.050, p <.1).

Following prior studies (Ramasubbu, Bharadwaj, & Tayi, 2015), we investigate the effect on productivity and quality as common dependent variables of team and project performance. Therefore, we calculate a delta score of leader emotional display by subtracting the negative emotional display score from the positive emotional display score. Both delta scores are used to split the 999 cases into equally large sets of positive vs negative teams (we disregard the median case). Using a Likelihood ratio test to compare OLS regression results of positive emotional scores with negative emotional scores, we find significantly better predictions for the negative emotional scores across all three models for quality and productivity scores[16].

We also test for reverse causality by calculating all three models using team leader emotions as the dependent variables and team emotions as the independent variables. The resulting predictions perform worse across all models. The best prediction results from model 2a' with an adjusted R^2 value of .234. Model 3 finds no significant interactions. These results suggest that team leader emotion effecting team emotion is a better predictive model than vice versa and therefore, support our developed theory.

4.2.5. Discussion

Despite prior research suggesting an emotional contagion effect within teams (Barsade, 2002; Hatfield et al., 1993; van Kleef, 2009), empirical evidence in the context of virtual teams is lacking. Drawing on emotional contagion theory, we investigate the emotional effect of team leader's positive and negative emotional displays on the corresponding team emotional displays in the context of OSS teams. Additionally, to the best of our knowledge, this study is the first to investigate collaboration function and collaboration structure, extending existing research in this domain that neglects both collaboration properties thus far. Using a large sample of virtual teams, we test our proposed model and extend existing empirical findings

The analysis of the main effects shows significant associations between team leader positive emotional displays and the team emotional displays. In addition, team collaboration structure and team collaboration function have a significant direct effect. These associations remain significant following the inclusion of interaction effects. The interaction of collaboration function and team leader positive emotional displays is

[16] When predicting productivity or quality, we use an adjusted model 3. We remove productivity or quality and leader emotional display scores from the list of independent variables. We calculate the moderation effects for the relationship between team emotional display and productivity or quality. Predicting productivity, we find equation one (team positive emotional display) to result in an adjusted R^2 value of .178 and equation two (team negative emotional display) to result in an adjusted R^2 value of .391. Predicting quality, we find equation one to result in an adjusted R^2 value of .036 and equation two to result in an adjusted R^2 value of .130.

no significant predictor for the team's positive emotional displays. However, the interaction between team collaboration structure and team leader positive emotional displays is significantly associated with team positive emotional displays. The findings support our theoretical explanations that despite the absence of physical cues, emotions in the form of social information are contagious (van Kleef, 2009). During the exchange of these social information, we find collaboration structure interacts with the team leader emotional displays when predicting team emotional displays. However, the collaboration function does not significantly interact with the team leader emotional displays.

The analysis of the complementary models using negative emotional displays show similar results. While the moderating role of team collaboration structure is stronger when predicting team negative emotional displays, all models of positive emotional displays explain more variance of the team emotional displays compared to the corresponding model of negative emotional displays. We explain these results with the general tendency towards optimism (Sharot, 2012) and empirical research on personality suggesting that positive emotions correlate with optimism and negative emotions correlate with pessimism (Marshall, Wortman, Kusulas, Hervig, & et al, 1992). The analysis of cross-emotional affects lends further support for this explanation. Prior conceptual work also suggests stronger emotional sharing effects for positive emotions in contrast to negative emotions (Walter & Bruch, 2008). Our results are consistent with experimental studies on leadership using student teams, suggesting that leaders transmit emotions to their subordinates (T. Sy et al., 2005).

When investigating the interaction effects of the team collaboration structure, the data supports our hypothesized relationships for positive and negative emotional displays. Therefore, the effects of the team leader's emotional displays on the team's emotional displays increase with denser collaboration structures. These findings are supported by research on social networks that articulate the importance of network density for the spread of knowledge and information within a network (Granovetter, 1985). Often, these social networks are subject to a small-world property, indicating that limited degrees are required to connect two random nodes within the same network (Watts & Strogatz, 1998). This property helps to exchange social information quickly to the entire network. Upon arrival, the information are interpreted by the observer who wants to conform rather than confront, and therefore mimics the emotional surrounding (Cialdini & Trost, 1998).

Another interesting finding is the non-significant interaction effect between team leader emotional displays and the team collaboration function. Therefore, higher frequencies of collaborative communication do not help a leader to increase the contagious effect of their emotional displays. We suggest that an increased frequency of emotional displays are interpreted as insincere or inauthentic (Humphrey et al., 2008). In other words, an increased frequency of communication during collaboration suggests a rather coerced approach

to the emotional expression. This is supported by prior studies suggesting that emotional expression is most successful under authentic conditions (Avolio et al., 2004).

Our results suggest that emotional conformance rather relies on the number of participants expressing an emotional tendency as indicated by the collaboration structure, than the strength of the signal as indicated by the collaboration function. In order to recognize these emotional tendencies, a team member needs to be able to monitor the emotional behavior and to be able to act upon it (Derks et al., 2008; Mayer & Salovey, 1995). Research on the closely related concept of collective empathy highlights the role of trust as an antecedent of collective empathy (Akgün, Keskin, Cebecioglu, & Dogan, 2015). This study offers a possible explanation for our results, as a coerced act decreases trust and therefore, the affective empathy of the team members. Yet, trust and empathy are vital factors for high performance. This idea is also supported by our additional analyses. The negative effect of the three-way interactions suggests that the excessive expression of emotions will lead to negative effects.

Overall, our study follows prior calls for more data-driven research (G. George et al., 2014; S. W. J. Kozlowski et al., 2015; Müller et al., 2016), providing novel and interesting empirical results by using big data to investigate emotional contagion within OSS Teams. We find an effect of a team leader's emotional displays on the team's emotional displays, which holds true for positive and negative emotions alike. Given the differing effects of positive and negative emotions in the context of team behavior (e.g. Gable & Haidt, 2005; Lyubomirsky et al., 2005), team leaders need to be aware of these mechanisms. The correct management of emotions helps team leaders to influence team coordination behavior or team performance (Barsade & Knight, 2015; Homan et al., 2016). In addition, the study points out the beneficial effects of dense team structures (Macionis, 2013) onto emotional contagion within OSS teams.

Our research shows managers the importance of emotions within work teams. While prior research suggests the importance in formal organizations, this study shows the applicability towards virtual OSS teams. Therefore, we join other scholars by highlight the role of emotions and emotional capabilities for managers (Akgün, Keskin, Byrne, & Gunsel, 2011). In addition, we suggest that the emotional contagion effect is stronger for teams with higher density. Therefore, managers are urged to increase diverse networking and diverse collaboration within the team, so that all team members are better connected with each other's.

4.2.6. Limitation and directions for future research

While our study investigates interesting research avenues and offers multifaceted contributions, some limitations require future investigation. Our focus is the emotional contagion in the context of OSS development teams. Future studies could investigate the applicability of our results in other context, such as

development organizations with formal authority structure or emergent teams in not for profit organizations. Given the context of OSS, we benefit from a larger dataset and do not rely on perceived measures. Yet, assessing the replicability of our results to other context should be investigated.

In addition, further investigation towards the awareness of team leaders and team members about emotional contagion can spur further interesting insights. Given the emergence of leadership in OSS teams, the team leaders may not hold such a position in other facets of life and hence, would be less experienced in managing emotions. Given the motivational setting of OSS teams, the active management of emotions in order to influence team members could harm the trust relationships and therefore, diminish team effectiveness (Akgün et al., 2015). In addition, scholars should investigate the influence of personality profiles on the management of emotions. These personality profiles might differ based on the context. While OSS development team leaders can be expected to be intrinsically motivated, personality profiles in for-profit organizations might differ, as suggested by a study investigating the personality profiles of business professionals (Noll et al., 2012).

Other limitations that future studies need to consider concern our methodical approach. First, our data processing requires further testing. While we build on existing algorithms used in prior research, compare them, and adapt them to our context, future studies can use sophisticated learning algorithms to reduce errors even further. Second, existing data can be extended with further data sources. We relied on Ghtorrent and GitHub to extract our data; yet, future research can tap into other data sources or extend data collection to include survey data. Third, other evaluation methods need to be employed to test the causality. While we base our model on prior literature to support our causal inferences and test the model for reverse causality, our econometric tests are not able to assess any purported flow effects. These flow effects remain a challenge to future investigations.

5. Discussion

5.1. Summary of the Findings

This thesis investigates different challenges relating to software development in general and ASD in particular. It benefits from a methodical pluralism in order to identify human factors and design principles in order to enhance ASD. The derived subquestions articulate different gaps at the intersection of three literature streams. Gap 1 relates to the methodical integration of ASD and UCD. Prior research suggests benefits owing to the complementary characteristics of both methodical approaches (Fox et al., 2008). Gaps 2 and 4 relate to calls for more research into the human factors in ASD (Moe et al., 2008) and emotions as a regulating mechanism for coordinating behavior (Homan et al., 2016). Research gap 3 investigates three research streams when exploring early activities in ASD teams by benefiting from UCD practices and ideas.

Study 1 investigates the state-of-the-art in UCD and ASD. As a result of a systematic literature review, it derives generic principles for a user-centered ASD approach. An initially high-level coding scheme, distinguishing processes, practices, social and people-related aspects, and technologies/artifacts dimensions are enriched with lower-level elements using a data-centric approach. Synthesizing the identified and analyzed 83 relevant publications, five generic principles are derived: 1) separate product discovery and product creation, 2) iterative and incremental design and development, 3) parallel interwoven creation tracks, 4) continuous stakeholder involvement, and 5) artifact-mediated communication. Thus, the study contributes to the body of knowledge on ASD by 1) providing a broad overview of existing literature related to UCASD, 2) developing an analysis framework (in the form a coding system) for further studies beyond former classifications, and 3) deriving UCASD principles relating to practices and processes.

Study 2 investigates agility as an emergent phenomenon at the team-level. Using CAS theory, it captures multiple influencing levels of SDTs and their interplays with self-organization and emergence. Analyzing data from three agile SDTs in different contextual environments, each participating with four or more mutually exclusive roles, the results suggest that self-organization is a key process in team agility. While contextual factors often restrict self-organization, the findings suggest factors that enhance team autonomy. The findings help practitioners to improve the cost-effectiveness ratio of their team's operations. The theoretical contributions result from the development and test of theory-grounded propositions and the investigation of mature agile development teams.

Study 3 proposes PDISC, a method for software product discovery. A systematic review is conducted to extract design requirements and method fragments from the literature. The method fragments describe early activities and are documented using PDDs. Collectively, such method fragments form a method database that is used to develop PDISC. Expert interviews provide a subsequent evaluation and help to improve the

method. The results suggest activities, deliverables, and their relationships for the discovery of software products. While the study identifies sets of challenges and recommendations relating to the discovery of software products, three distinct principles are derived. The principles help the software product team to create a viable, desirable, and feasible product vision.

Study 4 draws on contagion theory and social network theory to propose a model that accounts for the effects of team leader emotional display on team member emotional display for positive and negative emotional displays. The study suggests that team collaboration structure and team collaboration function influence the effects of team leader emotional display. Using data from 999 open source projects, a set of initially developed hypotheses for positive and negative emotional displays are tested jointly using seemingly unrelated regression. The results suggest that the emotional contagion process applies to virtual teams and identifies significant moderation effects for the team collaboration structure. In addition, this effect increases for higher levels of team collaboration structure, investigating positive and negative emotional displays. The implications for research and practice show the importance of collaboration structure for the conformance and convergence of team emotions.

Thus, this thesis identifies different human factors and methodical design principles for that improve ASD. Study 2 identifies human factors on the individual-level, team-level, and organizational level as antecedents of team agility. These relationships are influenced by team autonomy. In addition, Study 4 suggests the importance of emotions, leadership, and collaboration in development teams. Studies 1 and 3 identify principles improving ASD through the integration of UCD and through the clarification of early activities during product discovery.

5.2. Theoretical Contributions

The results offer some interesting findings and various theoretical contributions to information systems development, software engineering, and team research.

5.2.1. Exploring the Integration of ASD and UCD

ASD has evolved towards an established research domain, drawing great interest from the software engineering and IS communities. The research has focused on the adoption of ASD (e.g. Gandomani & Nafchi 2015; Gill et al., 2016) and on the effects of particular practices (e.g. Strode, 2015). While few studies have investigated the extension of ASD (Fox et al., 2008), Study 1 develops clear principles along processes and practices when extending ASD with UCD. While three process-related principles and two practice-related principles can be contextualized in future research, the findings suggest a lack of evidence for the development of principles relating to the technologies and artifacts dimension. Study 1 spurs a vivid

exchange of research ideas between the ASD and UCD research community. On the one hand, ASD research benefits from the evidences gathered and ideas generated in the UCD community generally, and the involvement of users during the development process in particular. On the other hand, UCD research benefits from findings and suggestions from the ASD community generally, particularly the ability to quickly respond to changes. Study 1 also develops a coding system that helps future scholars to investigate more fine-grained effects in this nomological net.

5.2.2. Supporting Early Steps in ASD

Advancements on the central activities surrounding ASD have come from practice and research (Schwaber & Sutherland, 2013). Examples are studies on the introduction of ASD (e.g. Gandomani & Nafchi, 2015), the scalability and maturity of ASD (e.g. Gill et al., 2016), and the performance impacts of ASD (e.g. G. Lee & Xia, 2010). However, we lack a better understanding of the early activities conducted during the product discovery phase that ought to precede the development. Study 3 contributes by specifying a meta-model that provides clear guidance on the structures and sequences of activities required in the product discovery phase. This study is also the first to develop clear principles that ensure viability, desirability, and feasibility in the resulting product vision. These principles are validated through a subsequent expert evaluation. The study contributes an overview of the challenges and benefits of a product discovery phase in software production.

5.2.3. A New Perspective on Agility

While prior research into ASD and agility explains the concept, its antecedents, and implications, much of the work did not fully benefit from theory-driven research (Dybå & Dingsøyr, 2008). Especially the need to benefit from mature theory of reference disciplines has been mentioned (Dingsøyr & Dybå, 2012; Dybå & Dingsøyr, 2008). Drawing on CAS theory, Study 2 investigates team agility as an emergent phenomenon. While prior studies investigate team agility as a behavioral team process (e.g. G. Lee & Xia, 2010; Tripp et al., 2016), this study is the first to consider team agility as an emergent process. Further, autonomy level is an important explanation for the emergence of different agile states. The model was tested using a multicase study with three ASD teams of different maturity, meaningfully extending empirical results.

5.2.4. Understanding Emotional Contagion in Virtual OSS Teams

While prior research on emotional contagion in teams exist (Barsade, 2002; Hatfield et al., 1993; van Kleef, 2009), we still lack empirical proof relating to virtual teams. Study 4 draws on emotional contagion theory to investigate the effects of team leader emotional displays on OSS team emotional displays. In contrast to other studies that have focused on positive or negative emotions, Study 4 investigates the effects of positive and negative emotions simultaneously. Further, extending prior research in this domain, it is the first to

analyze the effects of team collaboration structure and team collaboration function on emotional contagion. The derived model is tested using a large sample of virtual teams, extending extant empirical proof.

Cumulative research into emotion extraction and different algorithms that can be applied have resulted in quality differences (Jurado & Rodriguez, 2015; Mohammad & Turney, 2013). Study 4 investigates two prominent algorithms for emotion extraction. Following a comparison, the Syuzhet R package suggests higher quality. Based on prior research, domain-specific properties of textual discourses are considered. These domain-specific properties are addressed through additional data processing and data cleaning in order to achieve results that are more accurate. Future research benefits from these findings and the domain-specific adaptations of the extraction process.

In addition, Study 4 contributes to leadership's roles in IS research. Studies have suggested leadership's influences on team cooperation and collaboration (Flouri & Berger, 2010) as well as on knowledge-sharing (J. Lee et al., 2014; Srivastava et al., 2006). Yet, the IS research lacks empirical evidence for emotions' effects in software development teams. Study 4 is one of the first studies to investigate emotional contagion in IS research. Further, empirical data provides evidence for the model, contributing to empirical research relating to software development teams.

5.2.5. Methodical Contributions: Big Data as a Valuable Data Source for IS Researchers

Recently, the use of big data in management research (G. George et al., 2014), team research (S. W. J. Kozlowski et al., 2015), and IS research (Müller et al., 2016) has been suggested. While capturing new empirical data has the benefit of being tailored to a research project, big data can answer new questions with high precision. Study 4 provides a great example of the use of big data from an open source repository in order to investigate emotional contagion in software development teams. Thus, a unique dataset could be accessed to helps answer a question that would have been difficult to answer otherwise. Projects of relevance to the analysis have also been identified. However, big data also requires researchers to carefully consider the questions they want to answer using the dataset.

5.3. Practical Contributions

The agile manifesto was introduced by practitioners in order to change and improve the ways development is conducted (Beck et al., 2001). The principles behind the agile manifesto started the journey away from an execution-oriented and production-oriented notion of software development (e.g. Brooks, 1975). ASD has since been on the rise, with different instantiations (e.g. Scrum, XP) and impacts beyond software development (e.g. new product development). Becoming agile is now on the agenda of managers for themselves (Forbes Coaches Council, 2016) and their organizations (Bazigos et al., 2015; McKinsey Digital,

2015, 2016). Overall, the thesis offers practical contributions along the human and methodical aspects of ASD.

Human aspects. The agile manifesto clearly suggests the preference for individuals and interactions. This thesis derives different benefits along human and behavioral aspects for practice. While agility is a desirable state, its achievement is challenging. Reports from industry suggest that it takes teams approximately four years to become truly agile (Bisson, 2015). The findings suggest different conditions that apply to teams with higher agility levels. In addition, the work distinguishes conditions at the individual, team, and organizational levels. The findings can help practitioners to improve the cost-effectiveness ratio of their team's operations. Further, the results suggest that some conditions (e.g. development time) can be defined at an organizational level and can therefore benefit from standardization. Simultaneously, other conditions (e.g. user research) are best managed flexibly across teams, so that each team finds the most effective setting.

Moreover, the importance of emotions in software development has been emphasized. In ASD teams, emotional cards are a suggested tool for team members to express their emotions (Pahuja, 2014). Further, the regulation of negative emotions has been identified as a key skill for development managers (Radley, 2014). The results can help practitioners to manage their and their team's emotions. The thesis suggests the importance of team density for the spread of emotions in the team. Given the importance of emotional intelligence in leadership (Forbes Coaches Council, 2016) and the importance of emotions in organizational change (Weathington, 2016), these findings have larger practical implications.

Methodical aspects. While interactions and individuals are valued more than tools and processes, the latter still play a key role in ASD. Particularly, the use of more lightweight practices and techniques are being adopted in practice. This thesis' methodical contributions include different principles that provide practitioners with simple yet meaningful guidelines. On the one hand, I suggest principles that relate to the process and practice dimensions when integrating UCD and ASD. On the other hand, I derive principles that assure the validity, desirability, and feasibility of a product vision. A product vision is the result of early activities that jointly form the product discovery phase. The discovery phase raises challenges for practitioners in the software industry (Cagan, 2012; Torres, 2016). Thus, I document a meta-model that helps practitioners to understand which activities can be conducted in this early discovery phase.

5.4. Limitations and Future Research

While this thesis addresses key research issues, each study has limitations; these have already been articulated in each study. I will now make some general remarks and will suggest broader research directions for future research. These research directions are articulated along human factors and methodical design principles.

Human aspects. While Studies 2 and 4 adopt a variance theoretical perspective, seeking to investigate differences in effects in the data, they do not investigate longitudinal effects. While this might be the most challenging task for future research, it is a promising research avenue. Both key concepts from Studies 2 and 4 (team agility and emotions) change over time. We need to better understand these changes over time in order to advance understandings of and theory-building around software development teams. Thus, future research could take an event-driven and process-oriented view of team dynamics (Marks et al., 2001) such as team agility and team emotions. In addition, the cross-sectional data collected does not allow the establishment of causality, as suggested by the research models. While the literature supports the interpretation of the results, any implied causality results from the adopted theoretical position (Gregor, 2006).

Methodical aspects. Studies 1 and 3 take a methodical perspective and investigate the extension of ASD. While both studies benefit from secondary data to derive their guidelines, such data focuses on software development and design. Yet, the integration of development and design, as well as the discovery of new products, can be applied to other contexts. Thus, the studies can be replicated in different contexts in order to increase the results' generalization (Johns, 2006). Moreover, the thesis investigates challenges of and benefits in the corresponding literature domains in order to derive meaningful design principles. Yet, the studies do not investigate the direct impacts of the extension of ASD with UCD or the incorporation of a product discovery phase on outcome measures such as product success. Thus, future studies can investigate the effects of integrating UCD and ASD, and the effects of introducing a formal product discovery phase on software product success (Gruner & Homburg, 2000).

6. Conclusion

Many software development organizations adopt the agile philosophy in one way or another (Beck et al., 2001). While prior research has already contributed to the procedural adoption of agile software development (ASD), teams struggle to become truly agile (Bisson, 2015; Gregory et al., 2016). This thesis extends the literature on ASD along design principles and human factors. Methodical extensions include the integration of ASD with user-centered design (UCD) and the exploration of an early phase pre-development that has been termed *product discovery*. The investigations on human aspects consider team agility as an emergent phenomenon and investigate emotional contagion in OSS development teams.

Thus, the thesis contributes to the literature by providing methodical extensions of ASD, answering calls in the literature (Fox et al., 2008; Larusdottir et al., 2017). First, this thesis explores the integration of ASD and UCD. Study 1 derives generic principles along practices and processes, contributing to a fruitful exchange of ideas and benefits between two research communities. Second, Study 3 explores early steps in software development by developing the meta-model PDISC, which provides clarity about the early pre-development activities and evaluates three principles that assure a product vision's desirability, feasibility, and viability.

Further, the thesis contributes by following a theory-driven approach to investigating human aspects of ASD in response to prior calls from scholars (Dingsøyr & Dybå, 2012; Dybå & Dingsøyr, 2008; van Kleef, 2009). First, Study 2 draws on complex adaptive systems theory in order to investigate team agility as an emergent phenomenon. The study explains different team agility levels through individual, team, and organizational factors of ASD teams. Second, Study 4 tests emotional contagion theory in the context of virtual OSS development teams. It finds that team collaboration structure enhances the contagion effect of a team leader's emotional display on the team's emotional display.

References

Abelein, U., Sharp, H., & Paech, B. (2013). Does involving users in software development really influence system success. *IEEE Software*, *30*(6), 17–23. http://doi.org/10.1109/MS.2013.124

Abras, C., Maloney-Krichmar, D., & Preece, J. (2004). User-centered design. In W. S. Bainbridge (Ed.), *Berkshire Encyclopedia of Human-Computer Interaction* (Vol. 2, pp. 763–768). Great Barrington, MA, USA: Berkshire Encyclopedia of Human-computer Interaction Front Cover William Sims Bainbridge Berkshire Publishing Group LLC, 2004 - Computers - 958 pages 1 Review This encyclopedia, edited by the deputy director of the National Science Foundation's Division o. http://doi.org/10.1.1.94.381

Acuña, S. T., Gómez, M., & Juristo, N. (2009). How do personality, team processes and task characteristics relate to job satisfaction and software quality? *Information and Software Technology*, *51*(3), 627–639. http://doi.org/10.1016/j.infsof.2008.08.006

Adikari, S., McDonald, C., & Campbell, J. (2009). Little Design Up-Front: A Design Science Approach to Integrating Usability into Agile Requirements Engineering. In *13th International Conference, HCI International* (pp. 549–558). San Diego, CA, US: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-02574-7_62

Ågerfalk, P. J., Fitzgerald, B., Holmström, H., Lings, B., Lundell, B., & Conchúir, E. Ó. (2005). A framework for considering opportunities and threats in distributed software development. *Proc. International Workshop on Distributed Software Development, Paris, France: Austrian Computer Society*, (August), 47–61. http://doi.org/10.1.1.93.5181

Aiken, L. S., & West, S. G. (1991). *Multiple regression: Testing and interpreting interactions*. London, UK: SAGE Publications.

Akgün, A. E., Keskin, H., Byrne, J. C., & Gunsel, A. (2011). Antecedents and Results of Emotional Capability in Software Development Project Teams*. *Journal of Product Innovation Management*, *28*(6), 957–973. http://doi.org/10.1111/j.1540-5885.2011.00845.x

Akgün, A. E., Keskin, H., Cebecioglu, A. Y., & Dogan, D. (2015). Antecedents and consequences of collective empathy in software development project teams. *Information & Management*, *52*(2), 247–259. http://doi.org/10.1016/j.im.2014.11.004

Alaa, G., & Fitzgerald, G. (2013). Re-Conceptualizing Agile Information Systems Development as a Complex Adaptive System. *Emergence: Complexity & Organization*, *15*(3), 1–20.

Alzoubi, Y. I., & Gill, A. Q. (2014). Agile global software development communication challenges: A systematic review. In *Proceedings of the pacific Asia Conference on nformation Systems (PACIS)* (p. Paper 20). AISeL.

Ames, A. L. (2001). Users first! An introduction to usability and user-centered design and development for technical information and products. In *IPCC 2001. Communication Dimensions. Proceedings IEEE International Professional Communication Conference (Cat. No.01CH37271)* (pp. 135–140). Piscataway, NJ, USA: IEEE. http://doi.org/10.1109/IPCC.2001.971558

Amoako-Gyampah, K. (1997). Exploring users' desire to be involved in computer systems development: An exploratory study. *Computers in Human Behavior*, *13*(1), 65–81. http://doi.org/10.1016/S0747-5632(96)00030-1

Amoako-Gyampah, K., & White, K. B. (1993). User involvement and user satisfaction: an exploratory contingency model. *Information & Management*, *25*(1), 1–10. http://doi.org/10.1016/0378-7206(93)90021-K

Anderson, P. (1999). Perspective: Complexity Theory and Organization Science. *Organization Science*, *10*(3), 216–232. http://doi.org/10.1287/orsc.10.3.216

Ardito, C., Buono, P., Caivano, D., Costabile, M. F., & Lanzilotti, R. (2013). Investigating and promoting UX practice in industry: An experimental study. *International Journal of Human Computer Studies*, *72*(6), 542–551. http://doi.org/10.1016/j.ijhcs.2013.10.004

Argote, L., Insko, C. A., Yovetich, N., & Romero, A. A. (1995). Group Learning Curves: The Effects of Turnover and Task Complexity on Group Performance1. *Journal of Applied Social Psychology*, *25*(6), 512–529. http://doi.org/10.1111/j.1559-1816.1995.tb01765.x

Arnowitz, J. (2013). Taking the fast RIDE. *Interactions*, *20*(4), 76. http://doi.org/10.1145/2486227.2486243

Arrow, H., McGrath, J. E., & Berdahl, J. (2000). *Small groups as complex systems: Formation, coordination, development, and adaptation*. Thousands Oaks, CA, USA: SAGE Publications.

Ashby, W. R. (1962). Principles of the Self-Organizing Dynamic System. In H. Von Foerster & J. G. W. Zopf (Eds.), *Principles of Self-Orga- nization: Transactions of the University of Illinois Symposium* (pp. 255–278). London, UK: Pergamon Press. http://doi.org/10.1080/00221309.1947.9918144

Avolio, B. J., Gardner, W. L., Walumbwa, F. O., Luthans, F., & May, D. R. (2004). Unlocking the mask: a look at the process by which authentic leaders impact follower attitudes and behaviors. *The Leadership Quarterly*, *15*(6), 801–823. http://doi.org/10.1016/j.leaqua.2004.09.003

Balijepally, V., Mahapatra, R., & Nerur, S. (2006). Assessing Personality Profiles of Software Developers in Agile Development. *Communications of the Association for Information Systems*, *18*(1), Article 4.

Balijepally, V., Mahapatra, R., Nerur, S., & Price, K. H. (2009). Are two heads better than one for software development? The productivity paradox of pair programming. *MIS Quarterly*, *33*(1), 91–118.

Banker, R. D., Davis, G. B., & Slaughter, S. a. (1998). Software Development Practices, Software Complexity, and Software Maintenance Performance: A Field Study. *Management Science*, *44*(4), 433–450. http://doi.org/10.1287/mnsc.44.4.433

Banker, R. D., & Slaughter, S. a. (2000). The Moderating Effects of Structure on Volatility and Complexity in Software Enhancement. *Information Systems Research*, *11*(3), 219–240. http://doi.org/10.1287/isre.11.3.219.12209

Barksdale, J. T., & McCrickard, D. S. (2010). Concept mapping in agile usability. In *Proceedings of the 28th of the international conference extended abstracts on Human factors in computing systems - CHI EA '10* (p. 4691). New York, NY, USA: ACM Press. http://doi.org/10.1145/1753846.1754212

Barksdale, J. T., & McCrickard, D. S. (2012). Software product innovation in agile usability teams: an analytical framework of social capital, network governance, and usability knowledge management. *International Journal of Agile and Extreme Software Development*, *1*(1), 52. http://doi.org/10.1504/IJAESD.2012.048302

Barksdale, J. T., Ragan, E. D., & McCrickard, D. S. (2009). Easing team politics in agile usability: A concept mapping approach. *Proceedings - 2009 Agile Conference, AGILE 2009*, 19–25. http://doi.org/10.1109/AGILE.2009.57

Baroudi, J. J., Olson, M. H., & Ives, B. (1986). An empirical study of the impact of user involvement on system usage and information satisfaction. *Communications of the ACM*, *29*(3), 232–238. http://doi.org/10.1145/5666.5669

Barsade, S. G. (2002). The Ripple Effect: Emotional Contagion and Its Influence on Group Behavior. *Administrative Science Quarterly*, *47*(4), 644. http://doi.org/10.2307/3094912

Barsade, S. G., & Gibson, D. E. (2007). Why Does Affect Matter in Organizations? *Academy of Management Perspectives*, *21*(1), 36–59.

Barsade, S. G., & Knight, A. P. (2015). Group Affect. *Annual Review of Organizational Psychology and Organizational Behavior*, *2*(1), 21–46. http://doi.org/10.1146/annurev-orgpsych-032414-111316

Baskerville, R., Ramesh, B., & Levine, L. (2003). Is internet-speed software development different? *IEEE Software*, *20*(6), 70–77.

Batra, D., Xia, W., VanderMeer, D., & Dutta, K. (2010). Balancing agile and structured development approaches to successfully manage large distributed software projects: a case study from the cruise line industry. *Communications of the Association for Information Systems*, *27*(1), 379–394.

Bazigos, M., Smet, A. De, & Gagnon, C. (2015). *Why agility pays*. *McKinsey Quarterly*.

Beath, C., & Orlikowski, W. (1994). The contradictory structure of systems development methodologies: deconstructing the IS-user relationship in information engineering. *Information Systems Research*, *5*(4), 350–377. http://doi.org/Doi 10.1287/Isre.5.4.350

Beck, K. (1999). *Extreme programming explained: embrace change*. Amsterdam, NL: Addison-Wesley Longman.

Beck, K., Beedle, M., Bennekum, A. van, Cockburn, A., Cunningham, W., Fowler, M., … Thomas, D. (2001). Principles behind the Agile Manifesto. Retrieved October 7, 2016, from http://agilemanifesto.org/principles.html

Begier, B. (2011). Working with Users to Ensure Quality of Innovative Software Product despite Uncertainties. In N. T. Nguyen, C. Kim, & A. Janiak (Eds.), *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 6591 LNAI, pp. 337–346). Daegu, KR: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-20039-7_34

Benbasat, I., Goldstein, D. K. D., & Mead, M. (1987). The Case Research Strategy in Studies of Information Systems. *MIS Quarterly*, *11*(3), 369–386.

Benigni, G., Gervasi, O., Passeri, F. L., & Kim, T.-H. (2010). USABAGILE_Web: A Web Agile Usability Approach for Web Site Design. In D. Taniar, O. Gervasi, B. Murgante, E. Pardede, & B. O. Apduhan (Eds.), *Computational Science and Its Applications – ICCSA 2010* (pp. 422–431). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-12165-4_34

Benington, H. D. (1983). Production of large computer programs. *Annals of the History of Computing*, *5*(4), 299–310.

Bertholdo, A. P. O., da Silva, T. S., de O. Melo, C., Kon, F., & Silveira, M. S. (2014). Agile Usability Patterns for UCD Early Stages. In A. Marcus (Ed.), *Design, User Experience, and Usability. Theories, Methods, and Tools for Designing the User Experience. DUXU 2014. Lecture Notes in Computer Science* (pp. 33–44). Cham, CH: Springer International Publishing Switzerland. http://doi.org/10.1007/978-3-319-07668-3_4

Bevan, N. (2009). International Standards for Usability Should Be More Widely Used. *Journal of Usability Studies*, *4*(3), 106–113. http://doi.org/10.1.1.177.1356

Beyer, H. (2010a). User-Centered Agile Methods. *Synthesis Lectures on Human-Centered Informatics*, *3*(1), 1–71. http://doi.org/10.2200/S00286ED1V01Y201002HCI010

Beyer, H. (2010b). *User-Centered Agile Methods*. *Synthesis Lectures on Human-Centered Informatics* (Vol. 3). http://doi.org/10.2200/S00286ED1V01Y201002HCI010

Beyer, H., Holtzblatt, K., & Baker, L. (2004). An Agile Customer-Centered Method: Rapid Contextual Design. In C. Zannier, H. Erdogmus, & L. Lindstrom (Eds.), *Extreme Programming and Agile Methods - XP/Agile Universe 2004* (pp. 50–59). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-540-27777-4_6

Bird, C., Pattison, D., D'Souza, R., Filkov, V., & Devanbu, P. (2008). Latent social structure in open source projects. In *Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of software engineering - SIGSOFT '08/FSE-16* (p. 24). New York, New York, USA: ACM Press.

http://doi.org/10.1145/1453101.1453107

Bisson, S. (2015). Microsoft's Road to Open Agile Development. Retrieved March 11, 2016, from http://thenewstack.io/visual-studio-online-microsofts-road-open-agile-development/

Bjorn-Andersen, N. (1988). Are "Human factors" Human? *The Computer Journal*, *31*(5), 386–390. http://doi.org/10.1093/comjnl/31.5.386

Bjørn, P., & Ngwenyama, O. (2009). Virtual team collaboration: building shared meaning, resolving breakdowns and creating translucence. *Information Systems Journal*, *19*(3), 227–253. http://doi.org/10.1111/j.1365-2575.2007.00281.x

Blau, B., & Hildenbrand, T. (2011). Product Line Engineering in Large-Scale Lean and Agile Software Product Development Environments - Towards a Hybrid Approach to Decentral Control and Managed Reuse. In *2011 Sixth International Conference on Availability, Reliability and Security* (pp. 404–408). IEEE. http://doi.org/10.1109/ARES.2011.66

Blomkvist, S. (2005). Towards a Model for Bridging Agile Development and User-Centered Design. In *Human-Centered Software Engineering — Integrating Usability in the Software Development Lifecycle* (pp. 219–244). Dordrecht, NL: Springer Netherlands. http://doi.org/10.1007/1-4020-4113-6_12

Blumberg, B., Cooper, D. R., & Schindler, P. S. (2011). *Business Research Methods* (3rd ed.). Mcgraw-Hill.

Bodker, K., Kensing, F., & Simonsen, J. (2009). *Participatory IT Design - Designing for Business and Workplace Realities*. Cambridge, MA, USA: The MIT Press.

Bodker, S. (1996). Creating Conditions for Participation: Conflicts and Resources in Systems Development. *Human-Computer Interaction*, *11*(3), 215–236. http://doi.org/10.1207/s15327051hci1103_2

Boehm, B. W. (1988). A spiral model of software development and enhancement. *Computer*, *21*(5), 61–72. http://doi.org/10.1109/2.59

Boehm, B. W. (2000). Requirements that Handle IKIWISI, COTS, and Rapid Change. *Computer*, *33*(7), 99–102.

Boehm, B. W. (2011). Some Future Software Engineering Opportunities and Challenges. In *The Future of Software Engineering* (pp. 1–32). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-15187-3_1

Boehm, B. W., Rombach, H. D., & Zelkowitz, M. V. (2005). *Foundations of Empirical Software Engineering*. (B. W. Boehm, H. Rombach, & M. Zelkowitz, Eds.)*The Legacy of Victor R. Basili*. Berlin, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/3-540-27662-9

Boell, S. K., & Cecez-Kecmanovic, D. (2015). On being "systematic" in literature reviews in IS. *Journal of Information Technology*, *30*(2), 161–173. http://doi.org/10.1057/jit.2014.26

Boivie, I., Gulliksen, J., & Göransson, B. (2006). The lonesome cowboy: A study of the usability designer role in systems development. *Interacting with Computers*, *18*(4), 601–634. http://doi.org/10.1016/j.intcom.2005.10.003

Börjesson, A., & Mathiassen, L. (2005). Improving software organizations: agility challenges and implications. *Information Technology & People*, *18*(4), 359–382. http://doi.org/10.1108/09593840510633329

Botzenhardt, A., & Meth, H. (2011). Cross-Functional Integration of Product Management and Product Design in Application Software Development: Exploration of Success Factors. In *Thirty Second International Conference on Informations Systems* (p. Paper 10). Shanghai, PRC: AISeL.

Boulding, K. E. (1956). General Systems Theory-The Skeleton of Science. *Management Science*, *2*(3), 197–

208.

Bourdieu, P. (1977). *Outline of a Theory of Practice*. Cambridge, MA, USA: Cambridge University Press.

Braun, C., Wortmann, F., Hafner, M., & Winter, R. (2005). Method construction - a core approach to organizational engineering. In *Proceedings of the 2005 ACM symposium on Applied computing - SAC '05* (p. 1295). New York, New York, USA: ACM Press. http://doi.org/10.1145/1066677.1066971

Brereton, P., Kitchenham, B. a., Budgen, D., Turner, M., & Khalil, M. (2007). Lessons from applying the systematic literature review process within the software engineering domain. *Journal of Systems and Software*, *80*(4), 571–583. http://doi.org/10.1016/j.jss.2006.07.009

Brhel, M., Meth, H., Maedche, A., & Werder, K. (2015). Exploring Principles of User-Centered Agile Software Development: A Literature Review. *Information and Software Technology*, *61*(1), 163–181. http://doi.org/10.1016/j.infsof.2015.01.004

Brinkkemper, S. (1996). Method engineering: engineering of information systems development methods and tools. *Information and Software Technology*, *38*(4), 275–280.

Brinkkemper, S., Saeki, M., & Harmsen, F. (1999). Meta-modelling based assembly techniques for situational method engineering. *Information Systems*, *24*(3), 209–228.

Brooks, F. P. J. (1975). *The Mythical Man-Month: Essays on Software Engineering*. Boston, MA: Addison-Wesley.

Broschinsky, D., & Baker, L. (2008). Using Persona with XP at LANDesk Software, an Avocent Company. In *Agile 2008 Conference* (pp. 543–548). Toronto, ON, CA: IEEE. http://doi.org/10.1109/Agile.2008.91

Brown, J., Lindgaard, G., & Biddle, R. (2008). Stories, Sketches, and Lists: Developers and Interaction Designers Interacting Through Artefacts. In *Agile 2008 Conference* (pp. 39–50). Toronto, ON, CA: IEEE. http://doi.org/10.1109/Agile.2008.54

Brown, J. M., Lindgaard, G., & Biddle, R. (2011). Collaborative Events and Shared Artefacts: Agile Interaction Designers and Developers Working Toward Common Aims. In *Agile Conference 2011* (pp. 87–96). Salt Lake City, UT, USA: IEEE. http://doi.org/10.1109/AGILE.2011.45

Budwig, M., Jeong, S., & Kelkar, K. (2009). When user experience met agile. In *Proceedings of the 27th international conference extended abstracts on Human factors in computing systems - CHI EA '09* (pp. 3075–3084). Boston, MA, USA: ACM Press. http://doi.org/10.1145/1520340.1520434

Burke, C. S., Stagl, K. C., Klein, C., Goodwin, G. F., Salas, E., & Halpin, S. M. (2006). What type of leadership behaviors are functional in teams? A meta-analysis. *The Leadership Quarterly*, *17*(3), 288–307. http://doi.org/10.1016/j.leaqua.2006.02.007

Butt, S. M., Ahmad, W. F. W., & Rahim, L. (2014). Handling tradeoffs between agile and usability methods. In *2014 International Conference on Computer and Information Sciences (ICCOINS)* (pp. 1–6). Kuala Lumpur, MY: IEEE. http://doi.org/10.1109/ICCOINS.2014.6868450

Buxton, B. (2007). *Sketching User Experiences: Getting the Design Right and the Right Design*. San Fransisco, CA, USA: Morgan Kaufmann.

Caballero, L., Moreno, A. M., & Seffah, A. (2014). Persona as a Tool to Involving Human in Agile Methods: Contributions from HCI and Marketing. In S. Sauer, C. Bogdan, P. Forbrig, R. Bernhaupt, & M. Winckler (Eds.), *Human-Centered Software Engineering. HCSE 2014* (pp. 283–290). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-662-44811-3_20

Cagan, M. (2007, September). Product Discovery. Retrieved March 2, 2015, from http://www.svpg.com/product-discovery

Cagan, M. (2012). Time-Boxing Product Discovery. Retrieved September 2, 2017, from

http://svpg.com/time-boxing-product-discovery/

Cajander, Å., Larusdottir, M., & Gulliksen, J. (2013). Existing but Not Explicit - The User Perspective in Scrum Projects in Practice. In P. Kotzé, G. Marsden, G. Lindgaard, J. Wesson, & M. Winckler (Eds.), *Human-Computer Interaction – INTERACT 2013* (pp. 762–779). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-40477-1_52

Campanelli, A. S., & Parreiras, F. S. (2015). Agile methods tailoring – A systematic literature review. *Journal of Systems and Software*, *110*, 85–100. http://doi.org/10.1016/j.jss.2015.08.035

Campos, P., & Nunes, N. (2007). Towards useful and usable interaction design tools: CanonSketch. *Interacting with Computers*, *19*(5–6), 597–613. http://doi.org/10.1016/j.intcom.2007.05.006

Carlsson, S. a., Henningsson, S., Hrastinski, S., & Keller, C. (2011). Socio-technical IS design science research: Developing design theory for IS integration management. *Information Systems and E-Business Management*, *9*(1), 109–131. http://doi.org/10.1007/s10257-010-0140-6

Carter, J. (1999). Incorporating standards and guidelines in an approach that balances usability concerns for developers and end users. *Interacting with Computers*, *12*(2), 179–206. http://doi.org/10.1016/S0953-5438(99)00011-9

Carvalho, C. R. M. de. (2010). MEX experience boards: a set of agile tools for user experience design. In *Proceeding IHC '10 Proceedings of the IX Symposium on Human Factors in Computing Systems* (pp. 213–216). Belo Horizonte, BR: Brazilian Computer Society.

Ceschi, M., Sillitti, A., Succi, G., & Panfilis, S. De. (2005). Project management in plan-based and agile companies. *IEEE Software*, *22*(3), 21–27.

Chamberlain, S., Sharp, H., & Maiden, N. (2006). Towards a framework for integrating agile development and user-centred design. In S. Chamberlain, H. Sharp, & N. Maiden (Eds.), *Extreme Programming and Agile Processes in Software Engineering* (pp. 143–153). Berlin, DE: Springer Berlin Heidelberg.

Charlier, S. D., Stewart, G. L., Greco, L. M., & Reeves, C. J. (2016). Emergent leadership in virtual teams: A multilevel investigation of individual communication and team dispersion antecedents. *Leadership Quarterly*, *27*(5), 745–764. http://doi.org/10.1016/j.leaqua.2016.05.002

Cho, L. (2009). Adopting an Agile Culture. In *Agile Conference 2009* (pp. 400–403). Chicago, IL, USA: IEEE. http://doi.org/10.1109/AGILE.2009.47

Choi, T. Y., Dooley, K. J., & Rungtusanatham, M. (2001). Supply networks and complex adaptive systems: Control versus emergence. *Journal of Operations Management*, *19*(3), 351–366. http://doi.org/10.1016/S0272-6963(00)00068-1

Chow, T., & Cao, D.-B. (2008). A survey study of critical success factors in agile software projects. *Journal of Systems and Software*, *81*(6), 961–971. http://doi.org/10.1016/j.jss.2007.08.020

Cialdini, R. B., & Trost, M. R. (1998). Social Influence: Social Norms, Conformity, and Compliance. In *The Handbook of Social Psychology: 2-Volume Set* (pp. 151–192).

Clauset, A., Shalizi, C. R., & Newman, M. E. J. (2009). Power-Law Distributions in Empirical Data. *SIAM Review*, *51*(4), 661–703. http://doi.org/10.1137/070710111

Clegg, C., Axtell, C., Damodaran, L., Farbey, B., Hull, R., Lloyd-Jones, R., … Tomlinson, C. (1997). Information technology: a study of performance and the role of human and organizational factors. *Ergonomics*, *40*(9), 851–871. http://doi.org/10.1080/001401397187694

Cloyd, M. H. (2001). Designing user-centered Web applications in Web time. *IEEE Software*, *18*(1), 62–69.

Cockburn, A. (2006). *Agile software development: the cooperative game* (2nd ed.). Upper Saddle River, NJ, USA: AddisonWesley Professional.

Cohen, S. G., & Bailey, D. E. (1997). What Makes Teams Work: Group Effectiveness Research from the Shop Floor to the Executive Suite. *Journal of Management*, *23*(3), 239–290. http://doi.org/10.1177/014920639702300303

Cohn, M. (2004). *User Stories Applied - for agile software development*. Boston, MA: Addison-Wesley Professional.

Cole, M. (1998). *Cultural Psychology: A Once and Future Discipline*. Boston, MA, USA: Harvard University Press.

Collins, A. L., Lawrence, S. A., Troth, A. C., & Jordan, P. J. (2013). Group affective tone: A review and future research directions. *Journal of Organizational Behavior*, *34*(S1), S43–S62. http://doi.org/10.1002/job.1887

Conboy, K. (2009). Agility from First Principles: Reconstructing the Concept of Agility in Information Systems Development. *Information Systems Research*, *20*(3), 329–354. http://doi.org/10.1287/isre.1090.0236

Constantine, L. L. (2002). Process agility and software usability: Toward lightweight usage-centered design. *Information Age*, *8*(8), 1–10. http://doi.org/10.1.1.22.2229

Constantine, L. L., & Lockwood, L. A. D. (2002). Usage-centered engineering for Web applications. *IEEE Software*, *19*(2), 42–50. http://doi.org/10.1109/52.991331

Cook, R. D. (1977). Detection of Influential Observation in Linear Regression. *Technometrics*, *19*(1), 15. http://doi.org/10.2307/1268249

Cooper, A. (1999). *The Inmates are Running the Asylum*. Indianapolis, AN, USA: Sams - Pearson Education.

Cooper, A., Reimann, R., Cronin, D., & Noessel, C. (2014). *About Face: The Essentials of Interaction Design*. Indianapolis, IN, USA: John Wiley & Sons.

Cooper, R. G. (2011). *Winning at New Products: Creating Value Through Innovation* (4th ed.). New York, NY, USA: Basic Books.

Cooper, R. G., & Kleinschmidt, E. J. (1994). Determinants of timeliness in product development. *The Journal of Product Innovation Management*, *11*(5), 381–396. http://doi.org/10.1016/0737-6782(94)90028-0

Cooper, R. G., & Sommer, A. F. (2016). From Experience: The Agile–Stage-Gate Hybrid Model: A Promising New Approach and a New Research Opportunity. *Journal of Product Innovation Management*, *33*(5), 513–526. http://doi.org/10.1111/jpim.12314

Cosentino, V., Luis, J., & Cabot, J. (2016). Findings from GitHub. In *Proceedings of the 13th International Workshop on Mining Software Repositories - MSR '16* (pp. 137–141). New York, New York, USA: ACM Press. http://doi.org/10.1145/2901739.2901776

Costabile, M. F. (2001). Usability in the Software Life cycle. In S. K. Chang (Ed.), *Handbook of Software Engineering and Knowledge Engineering - Volume I: Fundamentals* (pp. 179–192). Singapore, SGP: World Scientific Publishing Co. http://doi.org/10.1142/9789812389718_0010

Curşeu, P. L. (2006). Emergent states in virtual teams: a complex adaptive systems perspective. *Journal of Information Technology*, *21*(4), 249–261. http://doi.org/10.1057/palgrave.jit.2000077

da Silva, B. S., Aureliano, V. C. O., & Barbosa, S. D. J. (2006). Extreme designing. In *Proceedings of VII Brazilian symposium on Human factors in computing systems - IHC '06* (pp. 101–109). Natal, RN, BR: ACM Press. http://doi.org/10.1145/1298023.1298038

da Silva, T. S., Martin, A., Maurer, F., & Silveira, M. (2011). User-Centered Design and Agile Methods: A Systematic Review. In *Agile Conference 2011* (pp. 77–86). Salt Lake City, UT, USA: IEEE.

http://doi.org/10.1109/AGILE.2011.24

da Silva, T. S., Silveira, M., & Maurer, F. (2011). Best practices for integrating user-centered design and agile software development. In *10th Brazilian Symposium on Human Factors in Computing Systems and the 5th Latin American Conference on Human-Computer Interaction* (pp. 43–45). Porto de Galinhas, PE, BRl: Brazilian Computer Society. http://doi.org/10.1.1.224.645

da Silva, T. S., Silveira, M. S., de O. Melo, C., & Parzianello, L. C. (2013). Understanding the UX Designer's Role within Agile Teams. In A. Marcus (Ed.), *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 8012 LNCS, pp. 599–609). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-39229-0_64

da Silva, T. S., Silveira, M. S., & Maurer, F. (2013). Ten Lessons Learned from Integrating Interaction Design and Agile Development. In *Agile Conference 2013* (pp. 42–49). Nashvi: IEEE. http://doi.org/10.1109/AGILE.2013.11

Damodaran, L. (1996). User involvement in the systems design process-a practical guide for users. *Behaviour & Information Technology*, *15*(6), 363–377. http://doi.org/10.1080/014492996120049

Darcy, D., Kemerer, C., Slaughter, S., & Tomayko, J. (2005). The structural complexity of software an experimental test. *IEEE Transactions on Software Engineering*, *31*(11), 982–995.

de Choudhury, M., & Counts, S. (2013). Understanding affect in the workplace via social media. In *Proceedings of the 2013 conference on Computer supported cooperative work - CSCW '13* (p. 303). New York, New York, USA: ACM Press. http://doi.org/10.1145/2441776.2441812

de O. Melo, C., S. Cruzes, D., Kon, F., & Conradi, R. (2013). Interpretative case studies on agile team productivity and management. *Information and Software Technology*, *55*(2), 412–427. http://doi.org/10.1016/j.infsof.2012.09.004

Derks, D., Fischer, A. H., & Bos, A. E. R. (2008). The role of emotion in computer-mediated communication: A review. *Computers in Human Behavior*, *24*(3), 766–785. http://doi.org/10.1016/j.chb.2007.04.004

Detweiler, M. (2007). Managing UCD within agile projects. *Interactions*, *14*(3), 40. http://doi.org/10.1145/1242421.1242447

Diebold, P., & Dahlem, M. (2014). Agile Practices in Practice - A Mapping Study. In *18th International Conference on Evaluation and Assessment in Software Engineering* (p. Paper 30). London, UK: ACM. http://doi.org/10.1145/2601248.2601254

Dingsøyr, T., & Dybå, T. (2012). Team effectiveness in software development: Human and cooperative aspects in team effectiveness models and priorities for future studies. *2012 5th International Workshop on Co-Operative and Human Aspects of Software Engineering, CHASE 2012 - Proceedings*, (7465), 27–29. http://doi.org/10.1109/CHASE.2012.6223016

Dingsøyr, T., Dybå, T., & Abrahamsson, P. (2008). A preliminary roadmap for empirical research on agile software development. In *Proceedings - Agile 2008 Conference* (pp. 83–94). Toronto, ON, CA: IEEE. http://doi.org/10.1109/Agile.2008.50

Dingsøyr, T., Nerur, S., Balijepally, V., & Moe, N. B. (2012). A decade of agile methodologies: Towards explaining agile software development. *Journal of Systems and Software*, *85*(6), 1213–1221. http://doi.org/10.1016/j.jss.2012.02.033

Døjbak Håkonsson, D., Eskildsen, J. K., Argote, L., Mønster, D., Burton, R. M., & Obel, B. (2016). Exploration versus exploitation: Emotions and performance as antecedents and consequences of team decisions. *Strategic Management Journal*, *37*(6), 985–1001. http://doi.org/10.1002/smj.2380

Dorairaj, S., & Noble, J. (2013). Agile Software Development with Distributed Teams: Agility, Distribution

and Trust. In *Agile Conference 2013* (pp. 1–10). Nashville, TN, USA: IEEE. http://doi.org/10.1109/AGILE.2013.7

Dubé, L., & Paré, G. (2003). Rigor in information systems positivist case research: current practices, trends, and recommendations. *MIS Quarterly*, *27*(4), 597–635.

Düchting, M., Zimmermann, D., Nebe, K., Duechting, M., Zimmermann, D., & Nebe, K. (2007). Incorporating User Centered Requirement Engineering into Agile Software Development. In J. A. Jacko (Ed.), *Human-Computer Interaction. Interaction Design and Usability* (pp. 58–67). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-540-73105-4_7

Durkheim, E. (1897). étude de sociologie. In *Le suicide* (pp. 65–95).

Dybå, T., & Dingsøyr, T. (2008). Empirical studies of agile software development: A systematic review. *Information and Software Technology*, *50*(9–10), 833–859. http://doi.org/10.1016/j.infsof.2008.01.006

Dybå, T., & Dingsøyr, T. (2015). Agile Project Management: From Self-Managing Teams to Large-Scale Development. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering* (Vol. 2, pp. 945–946). Florence, IT: IEEE. http://doi.org/10.1109/ICSE.2015.299

Ebert, C. (2006). Understanding the product life cycle: Four key requirements engineering techniques. *IEEE Software*, *23*(3), 19–25.

Eisenhardt, K. M. (1989). Building Theories from Case Study Research. *Academy of Management Review*, *14*(4), 532–550. http://doi.org/10.5465/AMR.1989.4308385

Ekman, P., & Davidson, R. J. (1994). *The nature of emotion: Fundamental questions. Series in affective science.* New York, NY, USA: Oxford University Press.

Engesser, H., Claus, V., & Schwill, A. (1993). *Duden Informatik: ein Sachlexikon für Studium und Praxis*. Mannheim, DE: Dudenverlag.

Eppler, M. J., & Mengis, J. (2004). The Concept of Information Overload: A Review of Literature from Organization Science, Accounting, Marketing, MIS, and Related Disciplines. *The Information Society*, *20*(5), 325–344. http://doi.org/10.1080/01972240490507974

Evnin, J., & Pries, M. (2008). Are You Sure? Really? A Contextual Approach to Agile User Research. In *Agile Conference 2008* (pp. 537–542). Toronto, ON, CA: IEEE. http://doi.org/10.1109/Agile.2008.81

Faraj, S., Kudaravalli, S., & Wasko, M. (2015). LEADING COLLABORATION IN ONLINE COMMUNITIES. *Management Information Systems Quarterly*, *39*(2), 393–412. http://doi.org/Article

Feiner, J., & Andrews, K. (2012). Usability Reporting with UsabML. In M. Winckler, P. Forbrig, & R. Bernhaupt (Eds.), *Human-Centered Software Engineering* (pp. 342–351). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-34347-6_26

Felker, C., Slamova, R., & Davis, J. (2012). Integrating UX with scrum in an undergraduate software development project. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education - SIGCSE '12* (p. 301). Raleigh, NC, USA: ACM Press. http://doi.org/10.1145/2157136.2157226

Ferre, X., Juristo, N., & Moreno, A. M. (2005). Framework for Integrating Usability Practices into the Software Process. In F. Bomarius & S. Komi-Sirviö (Eds.), *Product Focused Software Process Improvement* (pp. 202–215). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/11497455_17

Ferre, X., & Medinilla, N. (2007). How a Human-Centered Approach Impacts Software Development. In *12th International Conference, HCI International* (pp. 68–77). Beijing, PRC: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-540-73105-4_8

Ferreira, J., Noble, J., & Biddle, R. (2007a). Agile Development Iterations and UI Design. In *Agile Conference 2007* (pp. 50–58). Washington, DC, USA: IEEE. http://doi.org/10.1109/AGILE.2007.8

Ferreira, J., Noble, J., & Biddle, R. (2007b). Up-Front Interaction Design in Agile Development. In *Agile Processes in Software Engineering and Extreme Programming* (pp. 9–16). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-540-73101-6_2

Ferreira, J., Sharp, H., & Robinson, H. (2010). Values and Assumptions Shaping Agile Development and User Experience Design in Practice. In A. Sillitti, A. Martin, X. Wang, & E. Whitworth (Eds.), *Agile Processes in Software Engineering and Extreme Programming* (pp. 178–183). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-13054-0_15

Ferreira, J., Sharp, H., & Robinson, H. (2012). Agile Development and User Experience Design Integration as an Ongoing Achievement in Practice. In *Agile Conference 2012* (pp. 11–20). Dallas, TX, USA: IEEE. http://doi.org/10.1109/Agile.2012.33

Fisher, K. G., & Bankston, A. (2009). From Cradle to Sprint: Creating a Full-Lifecycle Request Pipeline at Nationwide Insurance. In *Agile Conference 2009* (pp. 223–228). Chicago, IL, USA: IEEE. http://doi.org/10.1109/AGILE.2009.72

Fitzgerald, B. (2012). Software Crisis 2.0. *Computer*, *45*(4), 89–91. http://doi.org/10.1109/MC.2012.147

Fitzgerald, B., & Feller, J. (2001). Guest Editorial Open source software: investigating the software engineering, psychosocial and economic issues. *Information Systems Journal*, *11*(4), 273–276. http://doi.org/10.1046/j.1365-2575.2001.00109.x

Flouri, K., & Berger, H. (2010). Agile Development – Scrum Adopted in Practice but not in Principle. In *Proceedings of the UK Academy for Information Systems Conference 2010* (p. Paper 21). Oxford, UK: AISeL.

Forbes Coaches Council. (2016). 13 Leadership Skills You Didn't Need A Decade Ago That Are Now Essential. Retrieved September 2, 2017, from http://www.forbes.com/sites/forbescoachescouncil/2016/12/13/13-leadership-skills-you-didnt-need-a-decade-ago-that-are-now-essential

Fornell, C., & Larcker, D. F. (1981). Evaluating Structural Equation Models with Unobservable Variables and Measurement Error. *Journal of Marketing Research*, *18*(1), 39. http://doi.org/10.2307/3151312

Fox, D., Sillito, J., & Maurer, F. (2008). Agile Methods and User-Centered Design: How These Two Methodologies are Being Successfully Integrated in Industry. In *Agile Conference 2008* (pp. 63–72). Toronto, ON, CA: IEEE. http://doi.org/10.1109/Agile.2008.78

Freeman, P., & Hart, D. (2004). A science of design for software-intensive systems. *Communications of the ACM*, *47*(8), 19. http://doi.org/10.1145/1012037.1012054

Frishammar, J., Florén, H., & Wincent, J. (2011). Beyond managing uncertainty: Insights from studying equivocality in the fuzzy front end of product and process innovation projects. *IEEE Transactions on Engineering Management*, *58*(3), 551–563.

Gable, S. L., & Haidt, J. (2005). What (and why) is positive psychology? *Review of General Psychology*, *9*(2), 103–110. http://doi.org/10.1037/1089-2680.9.2.103

Gamble, R. F., & Hale, M. L. (2013). Assessing individual performance in Agile undergraduate software engineering teams. In *IEEE Frontiers in Education Conference* (pp. 1678–1684). Oklahoma City, OK, USA: IEEE. http://doi.org/10.1109/FIE.2013.6685123

Gandomani, T. J., & Nafchi, M. Z. (2015). An empirically-developed framework for Agile transition and adoption: A Grounded Theory approach. *Journal of Systems and Software*, *107*, 204–219. http://doi.org/10.1016/j.jss.2015.06.006

Garrity, E. (2001). Synthesizing user centered and designer centered IS development approaches using general systems theory. *Information Systems Frontiers*, *3*(1), 107–121.

Gasson, S. (1999). The Reality of User-Centered Design. *Journal of End User Computing*, *11*(4), 3–13. http://doi.org/10.4018/joeuc.1999100101

Gasson, S. (2003). Human-Centered vs. User-Centered Approaches to Information System Design. *Journal of Information Technology Theory and Application*, *5*(2), 29–46.

Gaubinger, K., & Rabl, M. (2014). Structuring the Front End of Innovation. In O. Gassmann & F. Schweitzer (Eds.), *Management of the Fuzzy Front End of Innovation* (pp. 15–30). Cham, DE: Springer International Publishing. http://doi.org/10.1007/978-3-319-01056-4_2

Gebert, D., Boerner, S., & Kearney, E. (2006). Cross-functionality and innovation in new product development teams: A dilemmatic structure and its consequences for the management of diversity. *European Journal of Work and Organizational Psychology*, *15*(4), 431–458. http://doi.org/10.1080/13594320600826314

George, G., Haas, M. R., & Pentland, A. (2014). Big Data and Management. *Academy of Management Journal*, *57*(2), 321–326. http://doi.org/10.5465/amj.2014.4002

George, J. M., & Bettenhausen, K. (1990). Understanding prosocial behavior, sales performance, and turnover: A group-level analysis in a service context. *Journal of Applied Psychology*, *75*(6), 698–709. http://doi.org/10.1037/0021-9010.75.6.698

Gilboa, I. (2010). *Theory of Decision under Uncertainty. Econometric Society Monographs* (Vol. 45). Cambridge, UK: Cambridge University Press.

Gill, A. Q., Henderson-Sellers, B., & Niazi, M. (2016). Scaling for agility: A reference model for hybrid traditional-agile software development methodologies. *Information Systems Frontiers*, 1–27. http://doi.org/10.1007/s10796-016-9672-8

Goldenberg, A., Halperin, E., van Zomeren, M., & Gross, J. J. (2016). The Process Model of Group-Based Emotion: Integrating Intergroup Emotion and Emotion Regulation Perspectives. *Personality and Social Psychology Review*, *20*(2), 118–141. http://doi.org/10.1177/1088868315581263

Goldstein, J. (1994). *The unshackled organization: Facing the challenge of unpredictability through spontaneous reorganization*. Portland, OR, USA: Productivity Press, Inc.

Goldstein, J. (1999). Emergence as a Construct: History and Issues. *Emergence*, *1*(1), 49–72. http://doi.org/10.1207/s15327000em0101_4

Goldstein, J. (2000). Emergence: A construct amid a thicket of conceptual snares. *Emergence*, *2*(1), 2–5. http://doi.org/10.1207/S15327000EM0201_02

Gonçalves, J., & Santos, C. (2011). POLVO - Software for Prototyping of Low-Fidelity Interfaces in Agile Development. In J. A. Jacko (Ed.), *Human-Computer Interaction. Design and Development Approaches* (pp. 63–71). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-21602-2_7

Goodman, P. (1986). Impact of task and technology on group performance. In P. Goodman (Ed.), *Designing Effective Work Groups* (pp. 120–167). San Fransisco, CA, USA: Jossey-Bass Inc.

Goodman, P., & Shah, S. (1992). Familiarity and work group outcomes. In S. Worchel, W. Wood, & J. A. Simpson (Eds.), *Group Process and Productivity* (pp. 276–298). Newbury Park, CA, USA: SAGE Publications.

Gooty, J., Connelly, S., Griffith, J., & Gupta, A. (2010). Leadership, affect and emotions: A state of the science review. *The Leadership Quarterly*, *21*(6), 979–1004. http://doi.org/10.1016/j.leaqua.2010.10.005

Göransson, B., Gulliksen, J., & Boivie, I. (2003). The usability design process - Integrating user-centered systems design in the software development process. *Software Process Improvement and Practice*, *8*(2), 111–131. http://doi.org/10.1002/spip.174

Göransson, B., Lif, M., & Gulliksen, J. (2003). Usability Design-Extending Rational Unified Process with a New Discipline. In J. A. Jorge, N. J. Nunes, & J. F. e Cunha (Eds.), *Interactive Systems. Design, Specification, and Verification. DSV-IS 2003. Lecture Notes in Computer Science, vol 2844.* (pp. 316–330). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-540-39929-2_22

Gould, J. J. D., & Lewis, C. (1985). Designing for usability: key principles and what designers think. *Communications of the ACM*, *28*(3), 300–311. http://doi.org/10.1145/3166.3170

Granovetter, M. (1985). Economic Action and Social Structure: The Problem of Embeddedness. *The American Journal of Sociology*, *91*(3), 481–510 ST–Economic Action and Social Structure.

Gregor, S. (2006). The Nature of Theory in Information Systems. *MIS Quarterly*, *30*(3), 611–642.

Gregor, S., & Hevner, A. (2013). Positioning and presenting design science research for maximum impact. *MIS Quarterly*, *37*(2), 337–355.

Gregory, P., Barroca, L., Sharp, H., Deshpande, A., Taylor, K., Uk, A. D. A., & Uk, K. A. (2016). The Challenges That Challenge: Engaging With Agile Practitioners' Concerns. *Information and Software Technology*, *0*, 1–13. http://doi.org/10.1016/j.infsof.2016.03.003

Gren, L., Torkar, R., & Feldt, R. (2015). The prospects of a quantitative measurement of agility: A validation study on an agile maturity model. *Journal of Systems and Software*, *107*, 38–49. http://doi.org/10.1016/j.jss.2015.05.008

Grudin, J. (1991). Interactive systems: Bridging the gaps between developers and users. *Computer*, *24*(4), 59–69. http://doi.org/10.1109/2.76263

Gruman, G., Morrison, A. S., & Retter, T. A. (2007). *Software Pricing Trends * - How Vendors Can Capitalize on the*. Delaware, USA.

Gruner, K., & Homburg, C. (2000). Does customer interaction enhance new product success? *Journal of Business Research*, *2963*(99), 1–14.

Gulliksen, J. (2007). How Do Developers Meet Users? – Attitudes and Processes in Software Development. In G. Doherty & A. Blandford (Eds.), *Interactive Systems. Design, Specification, and Verification* (Vol. 4323, pp. 1–10). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-540-69554-7_1

Gulliksen, J., Göransson, B., Boivie, I., Blomkvist, S., Persson, J., & Cajander, Å. (2003). Key principles for user-centred systems design. *Behaviour & Information Technology*, *22*(6), 397–409. http://doi.org/10.1080/01449290310001624329

Guzman, E., Azócar, D., & Li, Y. (2014). Sentiment analysis of commit comments in GitHub: an empirical study. *Proceedings of the 11th Working Conference on Mining Software Repositories - MSR 2014*, 352–355. http://doi.org/10.1145/2597073.2597118

Hackman, J. R. (2002). *Leading Teams: Setting the Stage for Great Performances*. Boston, MA: Harvard Business Review Press.

Hackman, J. R., & Oldham, G. R. (1980). *Work redesign and motivation.* (E. H. Schein & R. Beckhard, Eds.)*Organization Development Series*. FT Press.

Häger, F., Kowark, T., Krüger, J., Vetterli, C., Übernickel, F., & Uflacker, M. (2015). DT@Scrum: Integrating Design Thinking with Software Development Processes. In H. Plattner, C. Meinel, & L. Leifer (Eds.), *Design Thinking Research* (pp. 263–289). Cham, CH: Springer International Publishing.

http://doi.org/10.1007/978-3-319-06823-7_14

Haikara, J. (2007). Usability in Agile Software Development: Extending the Interaction Design Process with Personas Approach. In G. Concas, E. Damiani, M. Scotto, & G. Succi (Eds.), *Agile Processes in Software Engineering and Extreme Programming* (pp. 153–156). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-540-73101-6_22

Haklay, M., & Weber, P. (2008). OpenStreetMap: User-Generated Street Maps. *IEEE Pervasive Computing*, *7*(4), 12–18. http://doi.org/10.1109/MPRV.2008.80

Hansson, C., Dittrich, Y., & Randall, D. (2006). How to Include Users in the Development of Off-the-Shelf Software: A Case for Complementing Participatory Design with Agile Development. In *Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS'06)* (pp. 1–10). Kauai, HI, USA: IEEE. http://doi.org/10.1109/HICSS.2006.205

Harris, M., & Weistroffer, H. (2009). A New Look at the Relationship between User Involvement in Systems Development and System Success. *Communications of the Association for Information Systems*, *24*(1), 739–756.

Hatfield, E., Cacioppo, J., & Rapson, R. L. (1993). Emotional contagion. *Current Directions in Psychological Science*, *2*(3), 96–99.

He, J., Butler, B., & King, W. (2007). Team Cognition: Development and Evolution in Software Project Teams. *Journal of Management Information Systems*, *24*(2), 261–292. http://doi.org/10.2753/MIS0742-1222240210

Heikkilä, V. T., Paasivaara, M., Rautiainen, K., Lassenius, C., Toivola, T., & Järvinen, J. (2015). Operational release planning in large-scale Scrum with multiple stakeholders - A longitudinal case study at F-Secure Corporation. *Information and Software Technology*, *57*(1), 116–140.

Heimgärtner, R., & Solanki, A. (2014). Using Agile Methods in Intercultural HCI Design Projects. In A. Marcus (Ed.), *Building Sustainable Information Systems* (pp. 123–129). Boston, MA, USA: Springer US. http://doi.org/10.1007/978-3-319-07668-3_13

Hellmann, T. D., Hosseini-Khayat, A., & Maurer, F. (2010a). Agile Interaction Design and Test-Driven Development of User Interfaces – A Literature Review. In T. Dingsøyr, T. Dybå, & N. B. Moe (Eds.), *Agile Software Development* (pp. 185–201). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-12575-1_9

Hellmann, T. D., Hosseini-Khayat, A., & Maurer, F. (2010b). Supporting Test-Driven Development of Graphical User Interfaces Using Agile Interaction Design. In *2010 Third International Conference on Software Testing, Verification, and Validation Workshops* (pp. 444–447). Paris, FR: IEEE. http://doi.org/10.1109/ICSTW.2010.35

Helms, R. W., & Werder, K. (2013). Who Reads Corporate Tweets? Network Analysis of Follower Communities. In *19th Amercias Conference on Information Systems*. Chicago, Illinois, USA.: AIS.

Henderson-Sellers, B., Ralyté, J., Ågerfalk, P. J., & Rossi, M. (2014). Method Engineering as a Social Practice. In *Situational Method Engineering* (pp. 53–68). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-41467-1_3

Hennigs, L. (2012). Making Design Tangible in Software Development Projects. In A. Maedche, A. Botzenhardt, & L. Neer (Eds.), *Software for People* (pp. 151–167). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-31371-4_9

Hevner, A. (2007). A Three Cycle View of Design Science Research. *Scandinavian Journal of Information Systems*, *19*(2), 87–92.

Hildenbrand, T., & Meyer, J. (2012). Intertwining lean and design thinking: software product development from empathy to shipment. In *Software for People* (pp. 217–237). Berlin, Heidelberg, DE: Springer

Berlin Heidelberg.

Hochmüller, E., & Mittermeir, R. T. (2008). Agile process myths. In *Proceedings of the 2008 international workshop on Scrutinizing agile practices or shoot-out at the agile corral - APOS '08* (pp. 5–8). Leipzig, DE: ACM Press. http://doi.org/10.1145/1370143.1370145

Hoda, R., Noble, J., & Marshall, S. (2010). Organizing self-organizing teams. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - ICSE '10* (Vol. 1, pp. 285–294). Cape Town ZA: ACM Press. http://doi.org/10.1145/1806799.1806843

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. http://doi.org/10.1016/j.infsof.2010.10.009

Hodgetts, P. (2005). Experiences integrating sophisticated user experience design practices into agile processes. In *Agile Development Conference (ADC'05)* (pp. 235–242). Denver, CO, USA: IEEE Comput. Soc. http://doi.org/10.1109/ADC.2005.24

Holland, J. (1992). Complex adaptive systems. *Daedalus*, *121*(1), 17–30.

Holland, J. (2006). Studying complex adaptive systems. *Journal of Systems Science and Complexity*, *19*(1), 1–8.

Hollenbeck, J. R., Moon, H., Ellis, A. P. J., West, B. J., Ilgen, D. R., Sheppard, L., … Wagner, J. a. . I. (2002). Structural contingency theory and individual differences: Examination of external and internal person-team fit. *Journal of Applied Psychology*, *87*(3), 599–606. http://doi.org/10.1037//0021-9010.87.3.599

Hollis, B., & Maiden, N. (2013). Extending Agile Processes with Creativity Techniques. *IEEE Software*, *30*(5), 78–84.

Holmstrom, H., Conchúir, E. Ó., Ågerfalk, P. J., Fitzgerald, B., Conchuir, E., Agerfalk, P., & Fitzgerald, B. (2006). Global Software Development Challenges: A Case Study on Temporal, Geographical and Socio-Cultural Distance. In *IEEE International Conference on Global Software Engineering (ICGSE'06)* (pp. 3–11). Washington, DC, USA: IEEE Computer Society. http://doi.org/10.1109/ICGSE.2006.261210

Holzinger, A., Errath, M., Searle, G., Thurnher, B., & Slany, W. (2005). From extreme programming and usability engineering to extreme usability in software engineering education (XP+UE→XU). In *Proceeding COMPSAC-W'05 Proceedings of the 29th annual international conference on Computer software and applications conference* (pp. 169–172). Edinburgh, GB: IEEE Computer Society.

Homan, A. C., van Kleef, G. A., & Sanchez-Burks, J. (2016). Team members' emotional displays as indicators of team functioning. *Cognition and Emotion*, *30*(1), 134–149. http://doi.org/10.1080/02699931.2015.1039494

Honious, J., & Clark, J. (2006). Something to believe in [Reed Elsevier]. In *Agile Conference 2006* (pp. 203–212). Minneapolis, MN, USA: IEEE. http://doi.org/10.1109/AGILE.2006.47

Hosseini-Khayat, A., Hellmann, T. D., & Maurer, F. (2010). Distributed and Automated Usability Testing of Low-Fidelity Prototypes. In *2010 Agile Conference* (pp. 59–66). Orlando, FL, USA: IEEE. http://doi.org/10.1109/AGILE.2010.11

Hudson, W. (2001). Toward Unified Models in User-Centered and Object-Oriented Design. In *Object Modeling and User Interface Design: Designing Interactive Systems* (pp. 313–362). Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

Humayoun, S. R., Dubinsky, Y., & Catarci, T. (2011). A Three-Fold Integration Framework to Incorporate User–Centered Design into Agile Software Development. In *Human Centered Design* (Vol. 5619, pp. 55–64). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-21753-

1_7

Humayoun, S. R., Dubinsky, Y., & Catarci, T. (2014a). *Smart Organizations and Smart Artifacts*. (L. Caporarello, B. Di Martino, & M. Martinez, Eds.) (Vol. 7). Cham, CH: Springer International Publishing. http://doi.org/10.1007/978-3-319-07040-7

Humayoun, S. R., Dubinsky, Y., & Catarci, T. (2014b). User Evaluation Support Through Development Environment for Agile Software Teams. In L. Caporarello, B. di Martino, & M. Martinez (Eds.), *Smart Organizations and Smart Artifacts* (pp. 183–191). Cham, CH: Springer International Publishing Switzerland. http://doi.org/10.1007/978-3-319-07040-7_18

Humayoun, S. R., Dubinsky, Y., Catarci, T., Nazarov, E., & Israel, A. (2012). A model-based approach to ongoing product evaluation. In *Proceedings of the International Working Conference on Advanced Visual Interfaces - AVI '12* (p. 596). Capri Island, IT: ACM Press. http://doi.org/10.1145/2254556.2254666

Humphrey, R. H., Pollack, J. M., & Hawver, T. (2008). Leading with emotional labor. *Journal of Managerial Psychology*, *23*(2), 151–168. http://doi.org/10.1108/02683940810850790

Hussain, Z., Lechner, M., Milchrahm, H., Shahzad, S., Slany, W., Umgeher, M., & Wolkerstorfer, P. (2008). Agile User-Centered Design Applied to a Mobile Multimedia Streaming Application. In *HCI and Usability for Education and Work* (pp. 313–330). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-540-89350-9_22

Hussain, Z., Lechner, M., Milchrahm, H., Shahzad, S., Slany, W., Umgeher, M., … Wolkerstorfer, P. (2008). User Interface Design for a Mobile Multimedia Application: An Iterative Approach. In *First International Conference on Advances in Computer-Human Interaction* (pp. 189–194). Sainte Luce, FR: IEEE Computer Society. http://doi.org/10.1109/ACHI.2008.24

Hussain, Z., Milchrahm, H., Shahzad, S., Slany, W., Tscheligi, M., & Wolkerstorfer, P. (2009). Integration of Extreme Programming and User-Centered Design: Lessons Learned. In P. Abrahamsson, M. Marchesi, & F. Maurer (Eds.), *Agile Processes in Software Engineering and Extreme Programming* (pp. 174–179). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-01853-4_23

Hussain, Z., Slany, W., & Holzinger, A. (2009a). Current State of Agile User-Centered Design: A Survey. In A. Holzinger & K. Miesenberger (Eds.), *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 5889 LNCS, pp. 416–427). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-10308-7_30

Hussain, Z., Slany, W., & Holzinger, A. (2009b). Investigating Agile User-Centered Design in Practice: A Grounded Theory Perspective. In A. Holzinger & K. Miesenberger (Eds.), *HCI and Usability for e-Inclusion* (Vol. 5889, pp. 279–289). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-10308-7_19

Hutcheson, G. D., & Sofroniou, N. (1999). *The Multivariate Social Scientist: Introductory Statistics Using Generalized Linear Models* (1st ed.). London, UK: SAGE Publications Ltd.

Hyung-Jin Park, M., Lim, J. W., & Birnbaum-More, P. H. (2009). The Effect of Multiknowledge Individuals on Performance in Cross-Functional New Product Development Teams. *Journal of Product Innovation Management*, *26*(1), 86–96. http://doi.org/10.1111/j.1540-5885.2009.00336.x

Iivari, J., & Iivari, N. (2006). Varieties of User-Centeredness. In *Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS'06)* (Vol. 8, pp. 1–10). Kauai, HI, USA: IEEE. http://doi.org/10.1109/HICSS.2006.530

Iivari, J., & Iivari, N. (2011). The relationship between organizational culture and the deployment of agile methods. *Information and Software Technology*, *53*(5), 509–520.

http://doi.org/10.1016/j.infsof.2010.10.008

Iivari, N. (2004). Enculturation of user involvement in software development organizations - an interpretive case study in the product development context. In *Proceedings of the third Nordic conference on Human-computer interaction - NordiCHI '04* (pp. 287–296). Tampere, FI: ACM Press. http://doi.org/10.1145/1028014.1028059

Iivari, N., & Abrahamsson, P. (2002). The interaction between organizational subcultures and user-centered design-a case study of an implementation effort. In *Proceedings of the 35th Annual Hawaii International Conference on System Sciences* (pp. 3260–3268). Big Island, HI, USA: IEEE Comput. Soc. http://doi.org/10.1109/HICSS.2002.994362

Ilgen, D. R., Hollenbeck, J. R., Johnson, M., & Jundt, D. (2005). Teams in organizations: from input-process-output models to IMOI models. *Annual Review of Psychology*, *56*, 517–43. http://doi.org/10.1146/annurev.psych.56.091103.070250

Illmensee, T., & Muff, A. (2009). 5 Users Every Friday: A Case Study in Applied Research. In *Agile Conference 2009* (pp. 404–409). Chicago, IL, USA: IEEE. http://doi.org/10.1109/AGILE.2009.45

Inayat, I., Salim, S. S., Marczak, S., Daneva, M., & Shamshirband, S. (2015). A systematic literature review on agile requirements engineering practices and challenges. *Computers in Human Behavior*, *51*(B), 915–929. http://doi.org/10.1016/j.chb.2014.10.046

ISO. (1998). 9241-11. Ergonomic requirements for office work with visual display terminals (VDTs). Geneva, CH: International Organization for Standardization.

ISO. (1999). ISO 13407: Human Centred Design Process for Interactive Systems. Geneva, CH: International Organization for Standardization.

ISO. (2010). ISO 9241-210:2010 - Ergonomics of human-system interaction -- Part 210: Human-centred design for interactive systems. Geneva, CH: International Organization for Standardization.

Isomursu, M., Sirotkin, A., Voltti, P., & Halonen, M. (2012). User Experience Design Goes Agile in Lean Transformation -- A Case Study. In *Agile Conference 2012* (pp. 1–10). Dallas, TX, USA: IEEE. http://doi.org/10.1109/Agile.2012.10

Izard, C. (1977). *Human Emotions*. New York, NY, USA: Plenum Press.

Jain, R., & Suman, U. (2015). A Systematic Literature Review on Global Software Development Life Cycle. *ACM SIGSOFT Software Engineering Notes*, *40*(2), 1–14.

Jarvenpaa, S. L., Shaw, T. R., & Staples, D. S. (2004). Toward Contextualized Theories of Trust: The Role of Trust in Global Virtual Teams. *Information Systems Research*, *15*(3), 250–267. http://doi.org/10.1287/isre.1040.0028

Jennex, M. E. (2015). Literature Reviews and the Review Process : An Editor-in-Chief's Perspective. *Communications of the Association for Information Systems*, *36*(1), paper 8.

Ji, Y. G., & Yun, M. H. (2006). Enhancing the Minority Discipline in the IT Industry: A Survey of Usability and User-Centered Design Practice. *International Journal of Human-Computer Interaction*, *20*(2), 117–134. http://doi.org/10.1207/s15327590ijhc2002_3

Jockers, M. (2016). R Package "syuzhet."

Johns, G. (2006). The essential impact of context on organizational behavior. *Academy of Management Review*, *31*(2), 386–408.

Jokela, T. (2004). Evaluating the user-centredness of development organisations: Conclusions and implications from empirical usability capability maturity assessments. *Interacting with Computers*, *16*(6), 1095–1132. http://doi.org/10.1016/j.intcom.2004.07.006

Jokela, T., Iivari, N., Matero, J., & Karukka, M. (2003). The standard of user-centered design and the standard definition of usability. In *Proceedings of the Latin American conference on Human-computer interaction - CLIHC '03* (Vol. 46, pp. 53–60). Rio de Janeiro, BR: ACM Press. http://doi.org/10.1145/944519.944525

Jurado, F., & Rodriguez, P. (2015). Sentiment Analysis in monitoring software development processes: An exploratory case study on GitHub's project issues. *Journal of Systems and Software*, *104*, 82–89. http://doi.org/10.1016/j.jss.2015.02.055

Jurca, G., Hellmann, T. D., & Maurer, F. (2014). Integrating Agile and User-Centered Design: A Systematic Mapping and Review of Evaluation and Validation Studies of Agile-UX. In *Agile Conference 2014* (pp. 24–32). Orlando, FL, USA: IEEE. http://doi.org/10.1109/AGILE.2014.17

Jurka, T. P. (2012). R Package "sentiment."

Kacmar, K. M., Witt, L. a., Zivnuska, S., & Gully, S. M. (2003). The interactive effect of leader-member exchange and communication frequency on performance ratings. *Journal of Applied Psychology*, *88*(4), 764–772. http://doi.org/10.1037/0021-9010.88.4.764

Kahai, S. S., Carroll, E., & Jestice, R. (2007). Team collaboration in virtual worlds. *ACM SIGMIS Database*, *38*(4), 61. http://doi.org/10.1145/1314234.1314246

Kajko-Mattsson, M., & Nyfjord, J. (2009). A Model of Agile Evolution and Maintenance Process. In *42nd Hawaii International Conference on System Sciences* (pp. 1–10). Big Island, HI: IEEE. http://doi.org/10.1109/HICSS.2009.21

Kakar, A., & Carver, J. (2012). Best Practices for managing the fuzzy front-end of software development (SD): Insights from a systematic review of new product development (NPD) literature. In *Proceedings of International Research Workshop on IT Project Management* (p. Paper 14). Orlando, FL, USA: AISeL.

Kalliamvakou, E., Gousios, G., Blincoe, K., Singer, L., German, D. M., & Damian, D. (2014). The promises and perils of mining GitHub. In *Proceedings of the 11th Working Conference on Mining Software Repositories - MSR 2014* (pp. 92–101). New York, New York, USA: ACM Press. http://doi.org/10.1145/2597073.2597074

Kane, D. (2003). Finding a place for discount usability engineering in agile development: throwing down the gauntlet. In *Proceedings of the Agile Development Conference, 2003. ADC 2003* (pp. 40–46). Salt Lake City, UT, USA: IEEE. http://doi.org/10.1109/ADC.2003.1231451

Kankanhalli, A., Tan, B. C. Y., & Wei, K.-K. (2007). Conflict and Performance in Global Virtual Teams. *Journal of Management Information Systems*, *23*(3), 237–274. http://doi.org/10.2753/MIS0742-1222230309

Kautz, K. (2011). Investigating the design process: participatory design in agile software development. *Information Technology & People*, *24*(3), 217–235. http://doi.org/10.1108/09593841111158356

Kautz, K. (2012). Beyond simple classifications: contemporary information systems development projects as complex adaptive systems. In *Thirty Third International Conference on Information Systems* (pp. 1–20). Orlando, FL, USA: AISeL.

Kearney, E., Gebert, D., & Voelpel, S. C. (2009). When And How Diversity Benefits Teams: The Importance Of Team Members' Need For Cognition. *Academy of Management Journal*, *52*(3), 581–598. http://doi.org/10.5465/AMJ.2009.41331431

Kelly, J. R., & Barsade, S. G. (2001). Mood and Emotions in Small Groups and Work Teams. *Organizational Behavior and Human Decision Processes*, *86*(1), 99–130. http://doi.org/10.1006/obhd.2001.2974

Keltner, D., & Haidt, J. (1999). Social Functions of Emotions at Four Levels of Analysis. *Cognition &*

Emotion, *13*(5), 505–521. http://doi.org/10.1080/026999399379168

Kettunen, P. (2009). Adopting key lessons from agile manufacturing to agile software product development—A comparative study. *Technovation*, *29*(6–7), 408–422. http://doi.org/10.1016/j.technovation.2008.10.003

Khurana, A., & Rosenthal, S. R. (1998). Towards holistic front ends in new product development. *Journal of Product Innovation Management*, *15*(1), 57–74.

Kirkman, B. L., Rosen, B., Tesluk, P. E., & Gibson, C. B. (2004). The Impact of Team Empowerment on Virtual Team Performance: the Moderating Role of Face-To-Face Interaction. *Academy of Management Journal*, *47*(2), 175–192. http://doi.org/10.2307/20159571

Kitchenham, B., & Charters, S. (2007). *Guidelines for performing Systematic Literature Reviews in Software Engineering*. *EBSE Technical Report EBSE-2007-01* (Vol. 2). Durham, UK.

Kitchenham, B., Dybå, T., & Jorgensen, M. (2004). Evidence-Based Software Engineering. In *Proceedings of the 26th International Conference on Software Engineering* (pp. 273–281). Edinburgh, SCO, UK: IEEE.

Kitchenham, B., Pearl Brereton, O., Budgen, D., Turner, M., Bailey, J., & Linkman, S. (2009). Systematic literature reviews in software engineering - A systematic literature review. *Information and Software Technology*, *51*(1), 7–15. http://doi.org/10.1016/j.infsof.2008.09.009

Knight, A. P. (2015). Mood at the Midpoint: Affect and Change in Exploratory Search Over Time in Teams That Face a Deadline. *Organization Science*, *26*(1), 99–118. http://doi.org/10.1287/orsc.2013.0866

Knoll, S. W., & Horton, G. (2011). The Impact of Stimuli Characteristics on the Ideation Process: An Evaluation of the Change of Perspective "Analogy." In *44th Hawaii International Conference on System Sciences* (pp. 1–10). Kauai, HI: IEEE. http://doi.org/10.1109/HICSS.2011.416

Kollmann, J., Sharp, H., & Blandford, A. (2009). The importance of identity and vision to user experience designers on agile projects. In *Agile Conference 2009* (pp. 11–18). Chicago, IL, USA: IEEE. http://doi.org/10.1109/AGILE.2009.58

Kozlenkova, I. V., Samaha, S. a., & Palmatier, R. W. (2014). Resource-based theory in marketing. *Journal of the Academy of Marketing Science*, *42*(1), 1–21. http://doi.org/10.1007/s11747-013-0336-7

Kozlowski, S., & Ilgen, D. (2006). Enhancing the effectiveness of work groups and teams. *Psychological Science in the Public Interest*, *7*(3), 77–124.

Kozlowski, S. W. J., & Chao, G. T. (2012). The Dynamics of Emergence: Cognition and Cohesion in Work Teams. *Managerial and Decision Economics*, *33*(5–6), 335–354. http://doi.org/10.1002/mde.2552

Kozlowski, S. W. J., Chao, G. T., Chang, C.-H., & Fernandez, R. (2015). Team Dynamics: Using "Big Data" to Advance the Science of Team Effectiveness. In S. Tonidandel, E. King, & J. Cortina (Eds.), *Big Data at Work: The Data Science Revolution and Organizational Psychology* (pp. 272–309). New York, NY, USA: Routledge Academic.

Kristinsdottir, S., Larusdottir, M. K., & Cajander, Å. (2016). Responsibilities and Challenges of Product Owners at Spotify - An Exploratory Case Study. In C. Bogdan, J. Gulliksen, S. Sauer, P. Forbrig, M. Winckler, C. Johnson, … F. Kis (Eds.), *Human-Centered and Error-Resilient Systems Development* (pp. 3–16). Cham, CH: Springer International Publishing. http://doi.org/10.1007/978-3-319-44902-9_1

Kropp, E., & Koischwitz, K. (2014). User-centered-design in agile RE through an On-site User Experience Consultant. In *2014 IEEE 2nd International Workshop on Usability and Accessibility Focused Requirements Engineering (UsARE)* (pp. 9–12). Karlskrona, SE: IEEE. http://doi.org/10.1109/UsARE.2014.6890994

Kuusinen, K. (2014). Improving UX Work in Scrum Development: A Three-Year Follow-Up Study in a Company. In S. Sauer, C. Bogdan, P. Forbrig, R. Bernhaupt, & M. Winckler (Eds.), *5th IFIP WG 13.2 International Conference, Human-Centered Software Engineering 2014* (pp. 259–266). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-662-44811-3_17

Kuusinen, K. (2015). Overcoming Challenges in Agile User Experience Work: Cross-Case Analysis of Two Large Software Organizations. In *2015 41st Euromicro Conference on Software Engineering and Advanced Applications* (pp. 454–458). Funchal, PT: IEEE. http://doi.org/10.1109/SEAA.2015.38

Kuusinen, K. (2016). BoB: A Framework for Organizing Within-Iteration UX Work in Agile Development. In G. Cockton, M. Lárusdóttir, P. Gregory, & Å. Cajander (Eds.), *Integrating User-Centred Design in Agile Development* (pp. 205–224). Cham, CH: Springer International Publishing. http://doi.org/10.1007/978-3-319-32165-3_9

Kuusinen, K., & Mikkonen, T. (2013). Designing User Experience for Mobile Apps: Long-Term Product Owner Perspective. In *2013 20th Asia-Pacific Software Engineering Conference (APSEC)* (pp. 535–540). Bangkok, TH: IEEE. http://doi.org/10.1109/APSEC.2013.77

Kuusinen, K., Mikkonen, T., & Pakarinen, S. (2012). Agile User Experience Development in a Large Software Organization: Good Expertise but Limited Impact. In M. Winckler;, P. Forbrig, & R. Bernhaupt (Eds.), *HCSE 2012: Human-Centered Software Engineering* (pp. 94–111). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-34347-6_6

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. http://doi.org/10.1016/j.infsof.2010.11.010

Lai, W. (2016). *Fitting Power Law Distributions to Data*. Berkley, CA, USA.

Langfred, C. W. (2005). Autonomy and Performance in Teams: The Multilevel Moderating Effect of Task Interdependence. *Journal of Management*, *31*(4), 513–529. http://doi.org/10.1177/0149206304272190

Larman, C. (2003). *Agile and Iterative Development: A Manager's Guide*. Boston, MA, USA: Addison-Wesley Professional.

Larusdottir, M. K. (2011). Usability Evaluation in Software Development Practice. In *Human-Computer Interaction – INTERACT 2011* (pp. 430–433). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-23768-3_50

Larusdottir, M. K., Bjarnadottir, E. R., & Gulliksen, J. (2010). The Focus on Usability in Testing Practices in Industry. In P. Forbrig, F. Paternó, & A. Mark Pejtersen (Eds.), *Human-Computer Interaction. IFIP Advances in Information and Communication Technology* (pp. 98–109). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-15231-3_11

Larusdottir, M. K., Gulliksen, J., & Cajander, Å. (2017). A license to kill – Improving UCSD in Agile development. *Journal of Systems and Software*, *123*, 214–222. http://doi.org/10.1016/j.jss.2016.01.024

Latham, G. P., Winters, D., & Locke, E. A. (1994). Cognitive and motivational effects of participation: a mediator study. *Journal of Organizational Behavior*, *15*(1), 49–63.

Läuter, H. (1985). Cook, R. D., S. Weisberg: Residuals and influence in regression. *Biometrical Journal*, *27*(1), 229. http://doi.org/10.1002/bimj.4710270110

Leavitt, A., Keegan, B. C., & Clark, J. (2016). Ping to Win ? Non-Verbal Communication and Team Performance in Competitive Online Multiplayer Games. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 4337–4350. http://doi.org/10.1145/2858036.2858132

Lee, G., Espinosa, J. A., & DeLone, W. H. (2013). Task Environment Complexity, Global Team Dispersion, Process Capabilities, and Coordination in Software Development. *IEEE Transactions on Software*

Engineering, *39*(12), 1753–1771. http://doi.org/10.1109/TSE.2013.40

Lee, G., & Xia, W. (2005). The ability of information systems development project teams to respond to business and technology changes: a study of flexibility measures. *European Journal of Information Systems*, *14*(1), 75–92. http://doi.org/10.1057/palgrave.ejis.3000523

Lee, G., & Xia, W. (2010). Toward agile: an integrated analysis of quantitative and qualitative field data on software development agility. *Management Information Systems Quarterly*, *34*(1), 87–114.

Lee, J. C. (2006). Embracing agile development of usable software systems. In *CHI '06 extended abstracts on Human factors in computing systems - CHI EA '06* (p. 1767). Montreal, CN: ACM Press. http://doi.org/10.1145/1125451.1125784

Lee, J. C., Judge, T. K., & McCrickard, D. S. (2011). Evaluating eXtreme scenario-based design in a distributed agile team. In *CHI '11 Extended Abstracts on Human Factors in Computing Systems* (pp. 863–877). New York, NY, USA: ACM Press. http://doi.org/10.1145/1979742.1979681

Lee, J. C., & McCrickard, D. S. (2007). Towards Extreme(ly) Usable Software: Exploring Tensions Between Usability and Agile Software Development. In *Agile Conference 2007* (pp. 59–71). Washington, DC, USA: IEEE. http://doi.org/10.1109/AGILE.2007.63

Lee, J. C., McCrickard, D. S., & Stevens, K. T. (2009). Examining the foundations of agile usability with extreme scenario-based design. In *Agile Conference 2009* (pp. 3–10). Chicago, IL, USA.: IEEE. http://doi.org/10.1109/AGILE.2009.30

Lee, J., Lee, H., & Park, J.-G. (2014). Exploring the impact of empowering leadership on knowledge sharing, absorptive capacity and team performance in IT service. *Information Technology & People*, *27*(3), 366–386. http://doi.org/10.1108/ITP-10-2012-0115

Lee, S.-H., Ko, I.-Y., Kang, S., & Lee, D.-H. (2010). A Usability-Pattern-Based Requirements-Analysis Method to Bridge the Gap between User Tasks and Application Features. In *2010 IEEE 34th Annual Computer Software and Applications Conference* (pp. 317–326). Seoul, KR: IEEE. http://doi.org/10.1109/COMPSAC.2010.39

Leszek, A., & Courage, C. (2008). The Doctor is "In" -- Using the Office Hours Concept to Make Limited Resources Most Effective. In *Agile 2008 Conference* (pp. 196–201). Toronto, ON, CA: IEEE. http://doi.org/10.1109/Agile.2008.46

Levi, D. (2014). *Group Dynamics for Teams* (Fourth). Thousands Oaks, CA, USA: SAGE Publications.

Liedtka, J. (2015). Perspective: Linking Design Thinking with Innovation Outcomes through Cognitive Bias Reduction. *Journal of Product Innovation Management*, *32*(6), 925–938. http://doi.org/10.1111/jpim.12163

Lievesley, M. A., & Yee, J. S. R. (2006). The role of the interaction designer in an agile software development process. In *CHI '06 extended abstracts on Human factors in computing systems - CHI EA '06* (pp. 1025–1030). Montreal, CN: ACM Press. http://doi.org/10.1145/1125451.1125647

Liskin, O. (2015). How Artifacts Support and Impede Requirements Communication. In *Working Conference on Requirements Engineering: Foundation for Software Quality* (pp. 132–147). Essen, DE: Springer International Publishing Switzerland. http://doi.org/10.1007/978-3-319-16101-3_9

Liu, M., Hull, C. E., & Hung, Y.-T. C. (2016). Starting open source collaborative innovation: the antecedents of network formation in community source. *Information Systems Journal*. http://doi.org/10.1111/isj.12113

Lizano, F., Sandoval, M. M., & Stage, J. (2014). Integrating Usability Evaluations into Scrum: A Case Study Based on Remote Synchronous User Testing. In M. Kurosu (Ed.), *Human-Computer Interaction. Theories, Methods, and Tools. HCI 2014* (pp. 500–509). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-319-07233-3_46

Loniewski, G., Armesto, A., & Insfran, E. (2011). An architecture-oriented model-driven requirements engineering approach. In *Model-Driven Requirements Engineering Workshop* (pp. 31–38). Trento, IT: IEEE. http://doi.org/10.1109/MoDRE.2011.6045364

Losada, B., Urretavizcaya, M., & de Castro, I. F. (2011). An Integrated Approach to Develop Interactive Software. In P. Campos, N. Graham, J. Jorge, N. Nunes, P. Palanque, & M. Winckler (Eds.), *Human-Computer Interaction – INTERACT 2011* (pp. 470–474). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-23768-3_60

Losada, B., Urretavizcaya, M., & Fernández-Castro, I. (2013). A guide to agile development of interactive software with a "user Objectives"-driven methodology. *Science of Computer Programming*, *78*(11), 2268–2281. http://doi.org/10.1016/j.scico.2012.07.022

Losada, B., Urretavizcaya, M., López-Gil, J.-M., & Fernández-Castro, I. (2012). Combining InterMod agile methodology with usability engineering in a mobile application development. In *Proceedings of the 13th International Conference on Interacción Persona-Ordenador - INTERACCION '12* (pp. 1–8). Elche, ES: ACM Press. http://doi.org/10.1145/2379636.2379674

Lyubomirsky, S., King, L., & Diener, E. (2005). The Benefits of Frequent Positive Affect: Does Happiness Lead to Success? *Psychological Bulletin*, *131*(6), 803–855. http://doi.org/10.1037/0033-2909.131.6.803

Macaulay, C., Sloan, D., Jiang, X. J. X., Forbes, P., Loynton, S., Swedlow, J. R., & Gregor, P. (2009). Usability and User-Centered Design in Scientific Software Development. *IEEE Software*, *26*(1), 96–102. http://doi.org/10.1109/MS.2009.27

Macionis, J. J. (2013). *Sociology* (15th ed.). London, UK: Pearson.

Magee, J. C., & Tiedens, L. Z. (2006). Emotional Ties That Bind: The Roles of Valence and Consistency of Group Emotion in Inferences of Cohesiveness and Common Fate. *Personality and Social Psychology Bulletin*, *32*(12), 1703–1715. http://doi.org/10.1177/0146167206292094

Mahalanobis, P. C. (1936). On the Generalized Distance in Statistics. *National Science Institute of India*, *2*(1), 49–55.

Malone, T. W., & Crowston, K. (1994). The interdisciplinary study of coordination. *ACM Computing Surveys*, *26*(1), 87–119. http://doi.org/10.1145/174666.174668

Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S., & McClosky, D. (2014). The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations* (pp. 55–60). Stroudsburg, PA, USA: Association for Computational Linguistics. http://doi.org/10.3115/v1/P14-5010

Mao, J.-Y., Vredenburg, K., Smith, P. W., & Carey, T. (2005). The state of user-centered design practice. *Communications of the ACM*, *48*(3), 105–109. http://doi.org/10.1145/1047671.1047677

Marks, M., Mathieu, J. E., & Zaccaro, S. (2001). A temporally based framework and taxonomy of team processes. *Academy of Management Review*, *26*(3), 356–376.

Marschak, J., & Radner, R. (1972). *Economic Theory of Teams.* New Heavon, CT, US: Yale University Press.

Marshall, G. N., Wortman, C. B., Kusulas, J. W., Hervig, L. K., & et al. (1992). Distinguishing optimism from pessimism: Relations to fundamental dimensions of mood and personality. *Journal of Personality and Social Psychology*, *62*(6), 1067–1074. http://doi.org/10.1037/0022-3514.62.6.1067

Martin, A., Biddle, R., & Noble, J. (2004). The XP Customer Role in Practice: Three Studies. In *Agile Development Conference* (pp. 42–54). Salt Lake City, UT, USA: IEEE. http://doi.org/10.1109/ADEVC.2004.23

Martin, R. C. (1999). Iterative and incremental development (IID). *C++ Report*, *11*(2), 26–29.

Maruping, L. M., Venkatesh, V., & Agarwal, R. (2009). A Control Theory Perspective on Agile Methodology Use and Changing User Requirements. *Information Systems Research*, *20*(3), 377–399. http://doi.org/10.1287/isre.1090.0238

Maruping, L. M., Zhang, X., & Venkatesh, V. (2009). Role of collective ownership and coding standards in coordinating expertise in software project teams. *European Journal of Information Systems*, *18*(4), 355–371. http://doi.org/10.1057/ejis.2009.24

Mathieu, J. E., Gilson, L. L., & Ruddy, T. M. (2006). Empowerment and team effectiveness: an empirical test of an integrated model. *The Journal of Applied Psychology*, *91*(1), 97–108. http://doi.org/10.1037/0021-9010.91.1.97

Mathieu, J. E., Maynard, M. T., Rapp, T., & Gilson, L. (2008). Team Effectiveness 1997-2007: A Review of Recent Advancements and a Glimpse Into the Future. *Journal of Management*, *34*(3), 410–476. http://doi.org/10.1177/0149206308316061

Maturana, H. R., & Varela, F. J. (1980). *Autopoiesis and Cognition* (Vol. 42). Dordrecht, NL: Springer Netherlands. http://doi.org/10.1007/978-94-009-8947-4

Matz, D. C., & Wood, W. (2005). Cognitive dissonance in groups: the consequences of disagreement. *Journal of Personality and Social Psychology*, *88*(1), 22–37. http://doi.org/10.1037/0022-3514.88.1.22

Maurer, F., & Hellmann, T. D. (2013). People-Centered Software Development: An Overview of Agile Methodologies. In A. de Lucia & F. Filomena (Eds.), *Software Engineering - International Summer Schools, ISSSE 2009-2011, Salerno, Italy. Revised Tutorial Lectures* (pp. 185–215). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-36054-1_7

Mayer, J. D., & Salovey, P. (1995). Emotional intelligence and the construction and regulation of feelings. *Applied and Preventive Psychology*, *4*(3), 197–208. http://doi.org/10.1016/S0962-1849(05)80058-7

Mayhew, D. J., & Tremaine, M. M. (2005). A Basic Framework. In *Cost-Justifying Usability* (pp. 41–101). Boston, MA, USA: Elsevier. http://doi.org/10.1016/B978-012095811-5/50003-1

McCarthy, I. P., Tsinopoulos, C., Allen, P., & Rose-Anderssen, C. (2006). New Product Development as a Complex Adaptive System of Decisions. *Journal of Product Innovation Management*, *23*(5), 437–456. http://doi.org/10.1111/j.1540-5885.2006.00215.x

McGrath, J. E. (1964). *Social psychology: A brief Introduction*. New York, NY, USA: Holt, Rinehart, & Winston.

McGrath, J. E., Arrow, H., & Berdahl, J. L. (2000). The Study of Groups: Past, Present, and Future. *Personality and Social Psychology Review*, *4*(1), 95–105. http://doi.org/10.1207/S15327957PSPR0401_8

McInerney, P., & Maurer, F. (2005). UCD in agile projects. *Interactions*, *12*(6), 19–23. http://doi.org/10.1145/1096554.1096556

McKinsey Digital. (2015). *Industry 4.0 - How to navigate digitization of the manufacturing sector*.

McKinsey Digital. (2016). *McKinsey Digital 2016*.

Medina, N. M., Burella, J., Rossi, G., Grigera, J., & Luna, E. R. (2010). An Incremental Approach for Building Accessible and Usable Web Applications. In L. Chen, P. Triantafillou, & T. Suel (Eds.), *Web Information Systems Engineering – WISE 2010* (pp. 564–577). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-17616-6_49

Melnik, G., & Maurer, F. (2002). Perceptions of agile practices: A student survey. In D. Wells & L. Williams (Eds.), *Extreme Programming and Agile Methods—XP/Agile Universe* (pp. 241–250). Berlin, DE:

Springer Berlin Heidelberg.

Melnik, G., & Maurer, F. (2005). A cross-program investigation of students' perceptions of agile methods. In *27th International Conference on Software Engineering.* (pp. 481–488). St. Louis, MI, USA: IEEE. http://doi.org/10.1109/ICSE.2005.1553593

Memmel, T., Gundelsweiler, F., & Reiterer, H. (2007a). Agile human-centered software engineering. In T. C. Ormerod & C. Sas (Eds.), *Proceeding BCS-HCI '07 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). Lancaster, UK: BCS Learning & Development Ltd.

Memmel, T., Gundelsweiler, F., & Reiterer, H. (2007b). CRUISER: A Cross-Discipline User Interface and Software Enginee ring Lifecycle. In J. A. Jacko (Ed.), *Human-Computer Interaction. Interaction Design and Usability* (pp. 174–183). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-540-73105-4_20

Memmel, T., Reiterer, H., & Holzinger, A. (2007). Agile Methods and Visual Specification in Software Development: A Chance to Ensure Universal Access. In C. Stephanidis (Ed.), *Universal Acess in Human Computer Interaction. Coping with Diversity* (pp. 453–462). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-540-73279-2_51

Menor, L. J., Tatikonda, M. V., & Sampson, S. E. (2002). New service development: areas for exploitation and exploration. *Journal of Operations Management*, *20*(2), 135–157. http://doi.org/10.1016/S0272-6963(01)00091-2

Meso, P., & Jain, R. (2006). Agile software development: adaptive systems principles and best practices. *Information Systems Management*, *23*(3), 19–30.

Meszaros, G., & Aston, J. (2006). Adding Usability Testing to an Agile Project. In *AGILE 2006 (AGILE'06)* (pp. 289–294). Minneapolis, MN, USA: IEEE. http://doi.org/10.1109/AGILE.2006.5

Miller, L. (2005). Case Study of Customer Input For a Successful Product. In *Agile Conference* (pp. 225–234). Denver, CO, USA: IEEE.

Miller, L., & Sy, D. (2009). Agile user experience SIG. *Proceedings of the 27th International Conference*, 2751. http://doi.org/10.1145/1520340.1520398

Mintzberg, H. (1978). Patterns in strategy formation. *Management Science*, *24*(9), 934–948.

Misra, S. C., Kumar, V., & Kumar, U. (2009). Identifying some important success factors in adopting agile software development practices. *Journal of Systems and Software*, *82*(11), 1869–1890. http://doi.org/10.1016/j.jss.2009.05.052

Mittal, S. (2013). Emergence in stigmergic and complex adaptive systems: A formal discrete event systems perspective. *Cognitive Systems Research*, *21*, 22–39. http://doi.org/10.1016/j.cogsys.2012.06.003

Moe, N. B., Dingsøyr, T., & Dybå, T. (2010). A teamwork model for understanding an agile team: A case study of a Scrum project. *Information and Software Technology*, *52*(5), 480–491. http://doi.org/10.1016/j.infsof.2009.11.004

Moe, N. B., Dingsøyr, T., Dybå, T., & Ict, S. (2008). Understanding Self-organizing Teams in Agile Software Development. In *19th Australian Conference on Software Engineering Understanding* (pp. 76–85). Perth, WA, AU: IEEE. http://doi.org/10.1109/ASWEC.2008.28

Mohammad, S. M., & Turney, P. D. (2013). Crowdsourcing a Word-Emotion Association Lexicon. *Computational Intelligence*, *29*(3), 436–465. http://doi.org/10.1111/j.1467-8640.2012.00460.x

Müller-Wienbergen, F., Müller, O., Seidel, S., & Becker, J. (2011). Leaving the Beaten Tracks in Creative Work-A Design Theory for Systems that Support Convergent and Divergent Thinking. *Journal of the Association for Information Systems*, *12*(11), 714–740.

Müller, O., Junglas, I., Brocke, J. vom, & Debortoli, S. (2016). Utilizing big data analytics for information systems research: challenges, promises and guidelines. *European Journal of Information Systems*, *25*(4), 289–302. http://doi.org/10.1057/ejis.2016.2

Murgia, A., Tourani, P., Adams, B., & Ortu, M. (2014). Do developers feel emotions? an exploratory analysis of emotions in software artifacts. In ACM (Ed.), *Proceedings of the 11th Working Conference on Mining Software Repositories - MSR 2014* (pp. 262–271). Hyderabad, IN. http://doi.org/10.1145/2597073.2597086

Najafi, M., & Toyoshiba, L. (2008). Two Case Studies of User Experience Design and Agile Development. In *Agile 2008 Conference* (pp. 531–536). Toronto, ON, CA: IEEE. http://doi.org/10.1109/Agile.2008.67

Nebe, K., & Grötzbach, L. (2006). Aligning user centered design activities with established software development practices. In *Proceedings of the 4th Nordic conference on Human-computer interaction changing roles - NordiCHI '06* (pp. 485–486). Oslo, NO: ACM Press. http://doi.org/10.1145/1182475.1182544

Neff, J. J., Fulk, J., & Yuan, Y. C. (2014). Not in the Mood? Affective State and Transactive Communication. *Journal of Communication*, *64*(5), 785–805. http://doi.org/10.1111/jcom.12109

Nerur, S., & Balijepally, V. (2007). Theoretical reflections on agile development methodologies. *Communications of the ACM*, *50*(3), 79–83. http://doi.org/10.1145/1226736.1226739

Nerur, S., Mahapatra, R., & Mangalaraj, G. (2005). Challenges of migrating to agile methodologies. *Communications of the ACM*, *48*(5), 72–78. http://doi.org/10.1145/1060710.1060712

Nguyen-Duc, A., Cruzes, D. S., & Conradi, R. (2015). The impact of global dispersion on coordination, team performance and software quality-A systematic literature review. *Information and Software Technology*, *57*(1), 277–294. http://doi.org/10.1016/j.infsof.2014.06.002

Niedenthal, P. M., & Brauer, M. (2012). Social Functionality of Human Emotion. *Annual Review of Psychology*, *63*(1), 259–285. http://doi.org/10.1146/annurev.psych.121208.131605

Noll, T., Endrass, J., Scherrer, P., Rossegger, A., Urbaniok, F., & Mokros, A. (2012). A Comparison of Professional Traders and Psychopaths in a Simulated Non-Zero Sum Game Thomas. *Catalyst: A Social Justice Forum*, *2*(2), 1–13.

Nonaka, I., & Takeuchi, H. (1995). *The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation* (1st ed.). Oxford, UK: Oxford University Press.

Nwaka, S., & Hudson, A. (2006). Innovative lead discovery strategies for tropical diseases. *Nature Reviews Drug Discovery*, *5*(11), 941–955. http://doi.org/10.1038/nrd2144

Nyfjord, J., & Kajko-Mattsson, M. (2008). Degree of Agility in Pre-Implementation Process Phases. In *Making Globally Distributed Software Development a Success Story* (pp. 234–245). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-540-79588-9_21

Obendorf, H., & Finck, M. (2008). Scenario-based usability engineering techniques in agile development processes. In *Proceeding of the twenty-sixth annual CHI conference extended abstracts on Human factors in computing systems - CHI '08* (p. 2159). Florence, IT: ACM Press. http://doi.org/10.1145/1358628.1358649

Obendorf, H., Schmolitzky, A., & Finck, M. (2006). XPnUE–defining and teaching a fusion of eXtreme programming and usability engineering. In *HCI Educators Workshop* (pp. 23–24). Limerick, IE.

Oliveira, M. G., & Rozenfeld, H. (2010). Integrating technology roadmapping and portfolio management at the front-end of new product development. *Technological Forecasting and Social Change*, *77*(8), 1339–1354.

Orlikowski, W. J., & Baroudi, J. J. (1991). Studying Information Technology in Organizations : Research Approaches and Assumptions. *Information Systems Research*, *2*(1), 1–28.

Paelke, V., & Nebe, K. (2008). Integrating agile methods for mixed reality design space exploration. In *Proceedings of the 7th ACM conference on Designing interactive systems - DIS '08* (pp. 240–249). Cape Town ZA: ACM Press. http://doi.org/10.1145/1394445.1394471

Paelke, V., & Sester, M. (2010). Augmented paper maps: Exploring the design space of a mixed reality system. *ISPRS Journal of Photogrammetry and Remote Sensing*, *65*(3), 256–265. http://doi.org/10.1016/j.isprsjprs.2009.05.006

Pahuja, S. (2014). Emotion Cards for Agile Teams. Retrieved September 2, 2017, from https://www.infoq.com/news/2014/12/emotion-cards

Pancake, C. M. (1997). Improving quality through user-centered design. In *Quality of Numerical Software - Assessment and enhancement* (pp. 44–60). New York, NY, USA: Springer US. http://doi.org/10.1007/978-1-5041-2940-4_4

Parrott, W. G. (2001). *Emotions in Social Psychology: Essential Readings*. Philadelphia, PA, US: Taylor & Francis.

Patton, J. (2002a). Designing Requirements: Incorporating Usage-Centered Design into an Agile SW Development Process. In D. Wells & L. Williams (Eds.), *Extreme Programming and Agile Methods — XP/Agile Universe 2002* (pp. 1–12). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/3-540-45672-4_1

Patton, J. (2002b). Hitting the target: adding interaction design to agile software development. In *OOPSLA 2002 Practitioners Reports* (p. 7). Salt Lake City, UT, USA.

Pavlou, P. a., & El Sawy, O. a. (2006). From IT Leveraging Competence to Competitive Advantage in Turbulent Environments: The Case of New Product Development. *Information Systems Research*, *17*(3), 198–227. http://doi.org/10.1287/isre.1060.0094

Peffers, K., Tuunanen, T., Rothenberger, M. a., & Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, *24*(3), 45–77. http://doi.org/10.2753/MIS0742-1222240302

Peixoto, C. S. A. (2009). Human-computer interface expert system for agile methods. In *Proceedings of the ITI 2009 31st International Conference on Information Technology Interfaces* (pp. 311–316). Cavtat/Dubrovnik, HR: IEEE. http://doi.org/10.1109/ITI.2009.5196100

Peixoto, C. S. A., & da Silva, A. E. A. (2009). A Conceptual Knowledge Base Representation for Agile Design of Human-Computer Interface. In *2009 Third International Symposium on Intelligent Information Technology Application* (pp. 156–160). Nanchang, PRC: IEEE. http://doi.org/10.1109/IITA.2009.393

Pelled, L. H., Eisenhardt, K. M., & Xin, K. R. (1999). Exploring the black box: An analysis of work group diversity, conflict, and performance. *Administrative Science Quarterly*, *44*(1), 1–28. http://doi.org/10.2307/2667029

Peres, A. L., da Silva, T. S., Silva, F. S., Soares, F. F., Carvalho, C. R. M. De, & Meira, S. R. D. L. (2014). AGILEUX Model: Towards a Reference Model on Integrating UX in Developing Software Using Agile Methodologies. In *Agile Conference 2014* (pp. 61–63). Ieee. http://doi.org/10.1109/AGILE.2014.15

Pescosolido, A. T. (2002). Emergent leaders as managers of group emotion. *The Leadership Quarterly*, *13*(5), 583–599. http://doi.org/10.1016/S1048-9843(02)00145-5

Peters, L. M., & Karren, R. J. (2009). An Examination of the Roles of Trust and Functional Diversity on Virtual Team Performance Ratings. *Group & Organization Management*, *34*(4), 479–504.

http://doi.org/10.1177/1059601107312170

Peters, L. M., & Manz, C. C. (2007). Identifying antecedents of virtual team collaboration. *Team Performance Management: An International Journal*, *13*(3/4), 117–129. http://doi.org/10.1108/13527590710759865

Petersen, K., & Wohlin, C. (2010). The effect of moving from a plan-driven to an incremental software development approach with agile practices. *Empirical Software Engineering*, *15*(6), 654–693. http://doi.org/10.1007/s10664-010-9136-6

Petrovic, K., & Siegmann, M. (2011). Make Space for the Customer: The Shift towards Customer Centricity. In *Design, User Experience, and Usability. Theory, Methods, Tools and Practice* (pp. 485–490). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-21675-6_56

Pikkarainen, M., Haikara, J., Salo, O., Abrahamsson, P., & Still, J. (2008). The impact of agile practices on communication in software development. *Empirical Software Engineering*, *13*(3), 303–337. http://doi.org/10.1007/s10664-008-9065-9

Plonka, L., Sharp, H., Gregory, P., & Taylor, K. (2014). UX Design in Agile: A DSDM Case Study. In G. Cantone & M. Marchesi (Eds.), *Agile Processes in Software Engineering and Extreme Programming* (pp. 1–15). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-319-06862-6_1

Plutchik, R. (1980). A general psychoevolutionary theory of emotion. In H. Kellerman & R. Plutchik (Eds.), *Emotion: Theory, Research and Experience. Theories of Emotion.* (Vol. 1, pp. 3–33). London, UK: Academic Press Inc. http://doi.org/10.1017/S0033291700053769

Posthuma, R. a, Campion, M. C., Masimova, M., & Campion, M. a. (2013). *A High Performance Work Practices Taxonomy: Integrating the Literature and Directing Future Research*. *Journal of Management* (Vol. 39). http://doi.org/10.1177/0149206313478184

Preece, J., Rogers, Y., & Sharp, H. (2002). *Interaction Design: beyond human-computer interaction*. New York, NY, USA: John Wiley & Sons.

Qumer, A., & Henderson-Sellers, B. (2007). Construction of an Agile Software Product-Enhancement Process by Using an Agile Software Solution Framework (ASSF) and Situational Method Engineering. In *31st Annual International Computer Software and Applications Conference - Vol. 1* (Vol. 1, pp. 539–542). Beijing, PRC: IEEE. http://doi.org/10.1109/COMPSAC.2007.98

Radjenović, D., Heričko, M., Torkar, R., & Živkovič, A. (2013). Software fault prediction metrics: A systematic literature review. *Information and Software Technology*, *55*(8), 1397–1418. http://doi.org/10.1016/j.infsof.2013.02.009

Radley, N. (2014). Survey: How Team Mood Can Impact Project Management. Retrieved September 2, 2017, from http://blog.softwareadvice.com/articles/project-management/survey-team-mood-project-management-0814

Rafla, T., Robillard, P. N., & Desmarais, M. (2007). A method to elicit architecturally sensitive usability requirements: its integration into a software development process. *Software Quality Journal*, *15*(2), 117–133. http://doi.org/10.1007/s11219-006-9009-9

Raison, C., & Schmidt, S. (2013). Keeping user centred design (UCD) alive and well in your organisation: Taking an agile approach. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *8012 LNCS*, 573–582. http://doi.org/10.1007/978-3-642-39229-0_61

Rajah, R., Song, Z., & Arvey, R. D. (2011). Emotionality and leadership: Taking stock of the past decade of research. *The Leadership Quarterly*, *22*(6), 1107–1119. http://doi.org/10.1016/j.leaqua.2011.09.006

Ramasubbu, N., Bharadwaj, A., & Tayi, G. K. (2015). Software Process Diversity: Conceptualization,

Measurement, and Analysis of Impact on Project Performance. *Mis Quarterly*, *39*(4), 787–807.

Rasmussen, R., & Christensen, A. (2011). Selecting users for participation in IT projects: Trading a representative sample for advocates and champions? *Interacting with Computers*, *23*(2), 176–187. http://doi.org/10.1016/j.intcom.2011.02.006

Rejeb, H. Ben, Boly, V., & Morel-Guimaraes, L. (2008). A New Methodology Based on Kano Model for the Evaluation of a New Product Acceptability during the Front-End Phases. In *32nd Annual IEEE International Computer Software and Applications Conference* (pp. 619–624). Turku, FI: IEEE. http://doi.org/10.1109/COMPSAC.2008.94

Ren, Y., & Argote, L. (2011). Transactive memory systems 1985–2010: An integrative framework of key dimensions, antecedents, and consequences. *The Academy of Management Annals*, *5*(1), 189–229.

Rexfelt, O., & Rosenblad, E. (2006). The progress of user requirements through a software development project. *International Journal of Industrial Ergonomics*, *36*(1), 73–81. http://doi.org/10.1016/j.ergon.2005.08.002

Rising, L., & Janoff, N. S. (2000). The Scrum software development process for small teams. *IEEE Software*, *17*(4), 26–32. http://doi.org/10.1109/52.854065

Rittenbruch, M., McEwan, G., Ward, N., Mansfield, T., & Bartenstein, D. (2002). Extreme Participation - Moving Extreme Programming Towards Participatory Design. In T. Binder, J. Gregory, & I. Wagner (Eds.), *7th biennial Participatory Design Conference, PDC 2002* (pp. 29–41). Malmoe, SE: CPSR.

Ronggang Zhou, Shengshan Huang, Xiangang Qin, & Huang, J. (2008). A survey of user-centered design practice in China. In *2008 IEEE International Conference on Systems, Man and Cybernetics* (pp. 1885–1889). Singapore, SGP: IEEE. http://doi.org/10.1109/ICSMC.2008.4811564

Rosenbaum, S., Rohn, J. A., & Humburg, J. (2000). A toolkit for strategic usability: results from workshops, panels, and surveys. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, *2*(1), 337–344.

Rosenberg, J. (1997). Some misconceptions about lines of code. In *Proceedings Fourth International Software Metrics Symposium* (pp. 137–142). IEEE Comput. Soc. http://doi.org/10.1109/METRIC.1997.637174

Royce, W. W. (1970). Managing the development of large software systems: concepts and techniques. In *IEEE Technical Papers of Western Electronic Show and Convention (WesCon)* (pp. 328–338). Los Angeles, CA, USA: IEEE.

Russell, J. A. (1980). A circumplex model of affect. *Journal of Personality and Social Psychology*, *39*(6), 1161–1178. http://doi.org/10.1037/h0077714

Russell, J. A. (2003). Core affect and the psychological construction of emotion. *Psychological Review*, *110*(1), 145–172. http://doi.org/10.1037/0033-295X.110.1.145

Salah, D., Paige, R., & Cairns, P. (2014a). A Practitioner Perspective on Integrating Agile and User Centred Design. In *28th International BCS Human Computer Interaction Conference* (pp. 100–109). Southport, UK. http://doi.org/10.14236/ewic/hci2014.11

Salah, D., Paige, R., & Cairns, P. (2014b). A systematic literature review for agile development processes and user centred design integration. In *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering - EASE '14* (p. Article 5). London, UK: ACM Press. http://doi.org/10.1145/2601248.2601276

Salah, D., Paige, R., & Cairns, P. (2014c). Integrating Agile Development Processes and User Centred Design- A Place for Usability Maturity Models? In S. Sauer, C. Bogdan, P. Forbrig, R. Bernhaupt, & M. Winckler (Eds.), *Human-Centered Software Engineering. HCSE 2014. Lecture Notes in Computer Science* (pp. 108–125). Berlin, Heidelberg, DE: Springer Berlin Heidelberg.

http://doi.org/10.1007/978-3-662-44811-3_7

Salas, E. (2005). Is there a "Big Five" in Teamwork? *Small Group Research*, *36*(5), 555–599. http://doi.org/10.1177/1046496405277134

Salas, E., Rozell, D., Mullen, B., & Driskell, J. E. (1999). The Effect of Team Building on Performance: An Integration. *Small Group Research*, *30*(3), 309–329. http://doi.org/10.1177/104649649903000303

Salvador, C., Nakasone, A., & Pow-Sang, J. A. (2014). A systematic review of usability techniques in agile methodologies. In *Proceedings of the 7th Euro American Conference on Telematics and Information Systems - EATIS '14* (pp. 1–6). Valparaiso, CL: ACM Press. http://doi.org/10.1145/2590651.2590668

Sarker, S., & Sarker, S. (2009). Exploring Agility in Distributed Information Systems Development Teams: An Interpretive Study in an Offshoring Context. *Information Systems Research*, *20*(3), 440–461. http://doi.org/10.1287/isre.1090.0241

Sarpong, D., & Maclean, M. (2012). Mobilising differential visions for new product innovation. *Technovation*, *32*(12), 694–702.

Sawyer, S. (2004). Software development teams. *Communications of the ACM*, *47*(12), 95–99.

Sawyer, S., Guinan, P. J., & Cooprider, J. (2010). Social interactions of information systems development teams: a performance perspective. *Information Systems Journal*, *20*(1), 81–107. http://doi.org/10.1111/j.1365-2575.2008.00311.x

Scheiber, F., Wruk, D., Oberg, A., Britsch, J., Woywode, M., Maedche, A., … Plach, M. (2012). Software Usability in Small and Medium Sized Enterprises in Germany: An Empirical Study. In *Software for People* (pp. 39–52). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-31371-4_3

Schön, E., Thomaschewski, J., & Escalona, M. J. (2017). Agile Requirements Engineering: A systematic literature review. *Computer Standards & Interfaces*, *49*(1), 79–91. http://doi.org/10.1016/j.csi.2016.08.011

Schultze, U., & Avital, M. (2011). Designing interviews to generate rich data for information systems research. *Information and Organization*, *21*(1), 1–16. http://doi.org/10.1016/j.infoandorg.2010.11.001

Schwaber, K., & Beedle, M. (2001). *Agile Software Development with Scrum*. *Series in Agile Software Development*. Upper Saddle River, NJ, USA: Pearson.

Schwaber, K., & Sutherland, J. (2013). The Scrum Guide. *The Definitive Guide to Scrum: The Rules of the Game*. Cambridge, MA, USA: Scrum.Org and ScrumInc.

Seaman, C. B. (1999). Qualitative methods in empirical studies of software engineering. *IEEE Transactions on Software Engineering*, *25*(4), 557–572. http://doi.org/10.1109/32.799955

Seaton, P., & Stewart, T. (1992). Evolving task oriented systems. In *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '92* (pp. 463–469). Monterey, CA, USA: ACM Press. http://doi.org/10.1145/142750.142900

Seffah, A., Desmarais, M. C., & Metzker, E. (2005). HCI, Usability and Software Engineering Integration: Present and Future. In *Human-Centered Software Engineering — Integrating Usability in the Software Development Lifecycle* (pp. 37–57). Dordrecht, NL: Springer Netherlands. http://doi.org/10.1007/1-4020-4113-6_3

Seffah, A., & Metzker, E. (2004). The obstacles and myths of usability and software engineering. *Communications of the ACM*, *47*(12), 71–76. http://doi.org/10.1145/1035134.1035136

Sein, M. K., Henfridsson, O., Rossi, M., & Lindgren, R. (2011). Action Design Research. *MIS Quarterly*, *35*(1), 37–56. http://doi.org/10.1080/0268396022000017725

Serrador, P., & Pinto, J. K. (2015). Does Agile work? — A quantitative analysis of agile project success. *International Journal of Project Management*, *33*(5), 1–12. http://doi.org/10.1016/j.ijproman.2015.01.006

Shah, D., Rust, R. T., Parasuraman, a., Staelin, R., & Day, G. S. (2006). The Path to Customer Centricity. *Journal of Service Research*, *9*(2), 113–124. http://doi.org/10.1177/1094670506294666

Sharifi, H., & Zhang, Z. (1999). A methodology for achieving agility in manufacturing organisations: An introduction. *International Journal of Production Economics*, *62*(1–2), 7–22. http://doi.org/10.1016/S0925-5273(98)00217-5

Sharot, T. (2012). *The Optimism Bias: A Tour of the Irrationally Positive Brain*. New York, NY, USA: Vintage.

Sibghatullah, M., Hussain, S., & Hussain, S. (2006). An Approach to Effective Product Development Life Cycle. In *International Conference on Emerging Technologies* (pp. 719–726). Peshawar, PK: IEEE. http://doi.org/10.1109/ICET.2006.335975

Sidky, A. (2007). *A Structured Approach to Adopting Agile Practices : The Agile Adoption Framework*. Virginia Polytechnic Institute and State University.

Sillitti, A., Ceschi, M., Russo, B., & Succi, G. (2005). Managing Uncertainty in Requirements: A Survey in Documentation-Driven and Agile Companies. In *11th IEEE International Software Metrics Symposium (METRICS'05)* (p. 17). Como, IT: IEEE. http://doi.org/10.1109/METRICS.2005.29

Silva, F. S., Soares, F. S. F., Peres, A. L., Azevedo, I. M. De, Vasconcelos, A. P. L. F., Kamei, F. K., & Meira, S. R. D. L. (2015). Using CMMI together with agile software development: A systematic review. *Information and Software Technology*, *58*, 20–43. http://doi.org/10.1016/j.infsof.2014.09.012

Singh, M. (2008). U-SCRUM: An Agile Methodology for Promoting Usability. In *Agile 2008 Conference* (pp. 555–560). Toronto, ON, CA: IEEE. http://doi.org/10.1109/Agile.2008.33

Singh, P. V., & Phelps, C. (2013). Networks, Social Influence, and the Choice Among Competing Innovations: Insights from Open Source Software Licenses. *Information Systems Research*, *24*(3), 539–560. http://doi.org/10.1287/isre.1120.0449

Smith, M. F. (1991). *Software Prototyping: Adoption, Practice, and Management*. New York, NY, USA: Mcgraw Hill Book Co Ltd.

Sohaib, O., & Khan, K. (2010). Integrating usability engineering and agile software development: A literature review. In *International Conference On Computer Design and Applications* (Vol. 2, pp. V2-32-V2-38). Qinhuangdao, PRC: IEEE. http://doi.org/10.1109/ICCDA.2010.5540916

Sperry, R., & Jetter, A. (2009). Theoretical framework for managing the front end of innovation under uncertainty. In *Portland International Conference on Management of Engineering & Technology* (pp. 2021–2028). Portland, OR: IEEE. http://doi.org/10.1109/PICMET.2009.5261940

Spohrer, K., Gholami, B., & Heinzl, A. (2012). Team Learning in Information Systems Development-A Literature Review. In *European Conference on Information Systems* (p. Paper 223). Barcelona, ES: AISeL.

Spraragen, S. L. (2005). The challenges in creating tools for improving the software development lifecycle. *ACM SIGSOFT Software Engineering Notes*, *30*(4), 1. http://doi.org/10.1145/1082983.1083118

Sproull, L., & Kiesler, S. (1986). Reducing Social Context Cues: Electronic Mail in Organizational Communication. *Management Science*, *32*(11), 1492–1512. http://doi.org/10.1287/mnsc.32.11.1492

Srivastava, A., Bartol, K. M., & Locke, E. A. (2006). Empowering Leadership in Management Teams: Effects on Knowledge Sharing, Efficacy, and Performance. *Academy of Management Journal*, *49*(6), 1239–1251. http://doi.org/10.5465/AMJ.2006.23478718

Staples, D. S., & Webster, J. (2008). Exploring the effects of trust, task interdependence and virtualness on knowledge sharing in teams. *Information Systems Journal*, *18*(6), 617–640. http://doi.org/10.1111/j.1365-2575.2007.00244.x

Stary, C. (1995). User interface design: the WHO, the WHAT, and the HOW revisited. In *Proceedings Nineteenth Annual International Computer Software and Applications Conference (COMPSAC'95)* (pp. 178–183). Dallas, TX, USA: IEEE. http://doi.org/10.1109/CMPSAC.1995.524777

Stevens, E. (2014). Fuzzy front-end learning strategies: Exploration of a high-tech company. *Technovation*, *34*(8), 431–440.

Stewart, K., & Gosain, S. (2006). The impact of ideology on effectiveness in open source software development teams. *Mis Quarterly*, *30*(2), 291–314.

Strode, D. E. (2015). A dependency taxonomy for agile software development projects. *Information Systems Frontiers*. http://doi.org/10.1007/s10796-015-9574-1

Suchman, L. (1995). Making work visible. *Communications of the ACM*, *38*(9), 56–64. http://doi.org/10.1145/223248.223263

Sundstrom, E., de Meuse, K. P., & Futrell, D. (1990). Work teams: Applications and effectiveness. *American Psychologist*, *45*(2), 120–133. http://doi.org/10.1037//0003-066X.45.2.120

Sundstrom, E., & McIntyre, M. (2000). Work groups: From the Hawthorne studies to work teams of the 1990s and beyond. *Group Dynamics: Theory, Research, and Practice*, *4*(1), 44–67. http://doi.org/10.1037//1089-2699

Sutcliffe, A., Thew, S., & Jarvis, P. (2011). Experience with user-centred requirements engineering. *Requirements Engineering*, *16*(4), 267–280. http://doi.org/10.1007/s00766-011-0118-z

Svensson, H., & Höst, M. (2005). Views from an Organization on How Agile Development Affects Its Collaboration with a Software Development Team. In F. Bomarius & S. Komi-Sirviö (Eds.), *Product Focused Software Process Improvement* (Vol. 3547, pp. 487–501). Berlin, Heidelberg, DE: Springer Berlin Heidelberg. http://doi.org/10.1007/b137178

Switzky, A. (2012). Incorporating UCD into the software development lifecycle. In *Proceedings of the 2012 ACM annual conference extended abstracts on Human Factors in Computing Systems Extended Abstracts - CHI EA '12* (p. 469). Austin, TX, USA: ACM Press. http://doi.org/10.1145/2212776.2212823

Sy, D. (2007). Adapting Usability Investigations for Agile User-centered Design. *Journal of Usability Studies*, *2*(3), 112–132.

Sy, T., Côté, S., & Saavedra, R. (2005). The Contagious Leader: Impact of the Leader's Mood on the Mood of Group Members, Group Affective Tone, and Group Processes. *Journal of Applied Psychology*, *90*(2), 295–305. http://doi.org/10.1037/0021-9010.90.2.295

Tessarolo, P. (2007). Is integration enough for fast product development? an empirical investigation of the contextual effects of product vision. *Journal of Product Innovation Management*, *24*(1), 69–82.

Torres, T. (2016). The Rise of Modern Product Discovery. Retrieved September 2, 2017, from https://www.producttalk.org/2016/03/rise-modern-product-discovery

Totterdell, P., Wall, T., Holman, D., Diamond, H., & Epitropaki, O. (2004). Affect Networks: A Structural Analysis of the Relationship Between Work Ties and Job-Related Affect. *Journal of Applied Psychology*, *89*(5), 854–867. http://doi.org/10.1037/0021-9010.89.5.854

Tourani, P., Jiang, Y., & Adams, B. (2014). Monitoring sentiment in open source mailing lists: exploratory study on the apache ecosystem. In *CASCON '14 Proceedings of 24th Annual International Conference on Computer Science and Software Engineering* (pp. 34–44). Markham, ON, CA: ACM.

Travers, J., & Milgram, S. (1969). An Experimental Study of the Small World Problem. *Sociometry*, *32*(4), 425–443.

Tripp, J. F., Riemenschneider, C. K., & Thatcher, J. B. (2016). Job satisfaction in agile development teams: Agile development as work redesign. *Journal of the Association of Information Systems*, *17*(4), 267–307.

Troy, L. C., Hirunyawipada, T., & Paswan, A. K. (2008). Cross-Functional Integration and New Product Success: An Empirical Investigation of the Findings. *Journal of Marketing*, *72*(6), 132–146. http://doi.org/10.1509/jmkg.72.6.132

Ungar, J. (2008). The Design Studio: Interface Design for Agile Teams. In *Agile 2008 Conference* (pp. 519–524). Toronto, ON, CA: IEEE. http://doi.org/10.1109/Agile.2008.51

Ungar, J., & White, J. (2008). Agile user centered design. In *Proceeding of the twenty-sixth annual CHI conference extended abstracts on Human factors in computing systems - CHI '08* (pp. 2167–2178). Florence, IT: ACM Press. http://doi.org/10.1145/1358628.1358650

van Hoek, R. I., Harrison, A., & Christopher, M. (2001). Measuring agile capabilities in the supply chain. *International Journal of Operations & Production Management*, *21*(1/2), 126–148. http://doi.org/10.1108/01443570110358495

van Kleef, G. A. (2009). How Emotions Regulate Social Life: The Emotions as Social Information (EASI) Model. *Current Directions in Psychological Science*, *18*(3), 184–188. http://doi.org/10.1111/j.1467-8721.2009.01633.x

van Kleef, G. A., Homan, A. C., Beersma, B., van Knippenberg, D., van Knippenberg, B., & Damen, F. (2009). Searing Sentiment Or Cold Calculation? The Effects Of Leader Emotional Displays On Team Performance Depend On Follower Epistemic Motivation. *Academy of Management Journal*, *52*(3), 562–580. http://doi.org/10.5465/AMJ.2009.41331253

Vanhanen, J., Itkonen, J., & Sulonen, P. (2003). Improving the interface between business and product development using agile practices and the cycles of control framework. In *Proceedings of the Agile Development Conference* (pp. 71–80). Salt Lake City, UT, USA: IEEE. http://doi.org/10.1109/ADC.2003.1231455

Vanhanen, J., & Lassenius, C. (2005). Effects of pair programming at the development team level : An experiment. In *Proceedings of the 4th International Symposium on Empirical Software Engineering* (pp. 336–345). Noosa Heads, QU, AU: IEEE.

Vasilescu, B., Posnett, D., Ray, B., Brand, M. G. J. van den, Serebrenik, A., Devanbu, P., & Filkov, V. (2015). Gender and Tenure Diversity in GitHub Teams. *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI)*, 3789–3798. http://doi.org/10.1145/2702123.2702549

Vasilescu, B., Yu, Y., Wang, H., Devanbu, P., & Filkov, V. (2015). Quality and productivity outcomes relating to continuous integration in GitHub. In *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering - ESEC/FSE 2015* (pp. 805–816). New York, New York, USA: ACM Press. http://doi.org/10.1145/2786805.2786850

Verschuren, P., Doorewaard, H., & Mellion, M. J. (2010). *Designing a research project* (2nd ed.). The Hague, NL: Eleven International Publishing House.

Vidgen, R., & Wang, X. (2009). Coevolving Systems and the Organization of Agile Software Development. *Information Systems Research*, *20*(3), 355–376. http://doi.org/10.1287/isre.1090.0237

Vlaanderen, K., Jansen, S., Brinkkemper, S., & Jaspers, E. (2011). The agile requirements refinery: Applying SCRUM principles to software product management. *Information and Software Technology*, *53*(1), 58–70. http://doi.org/10.1016/j.infsof.2010.08.004

Volberda, H. W. (1996). Toward the Flexible Form: How to Remain Vital in Hypercompetitive Environments. *Organization Science*, *7*(4), 359–374. http://doi.org/10.1287/orsc.7.4.359

vom Brocke, J., Simons, A., Niehaves, B., Riemer, K., Plattfaut, R., Cleven, A., … Reimer, K. (2009). Reconstructing the Giant: On the Importance of Rigour in Documenting the Literature Search Process. In *17th European Conference on Information Systems* (pp. 2206–2217). Verona, IT: AISeL.

Vredenburg, K., Mao, J.-Y., Smith, P. W., & Carey, T. (2002). A survey of user-centered design practice. In *Proceedings of the SIGCHI conference on Human factors in computing systems Changing our world, changing ourselves - CHI '02* (pp. 471–478). Minneapolis, MN, USA: ACM Press. http://doi.org/10.1145/503457.503460

Wageman, R., Hackman, J. R., & Lehman, E. (2005). Team Diagnostic Survey: Development of an Instrument. *The Journal of Applied Behavioral Science*, *41*(4), 373–398. http://doi.org/10.1177/0021886305281984

Wale-Kolade, A., Nielsen, P. A., & Päivärinta, T. (2013). Usability Work in Agile Systems Development Practice: A Systematic Review. In H. Linger, J. Fisher, A. Barnden, C. Barry, M. Lang, & C. Schneider (Eds.), *Building Sustainable Information Systems* (pp. 569–582). Boston, MA: Springer US. http://doi.org/10.1007/978-1-4614-7540-8_44

Walter, F., & Bruch, H. (2008). The positive group affect spiral: a dynamic model of the emergence of positive affective similarity in work groups. *Journal of Organizational Behavior*, *29*(2), 239–261. http://doi.org/10.1002/job.505

Walter, F., & Scheibe, S. (2013). A literature review and emotion-based model of age and leadership: New directions for the trait approach. *The Leadership Quarterly*, *24*(6), 882–901. http://doi.org/10.1016/j.leaqua.2013.10.003

Wan Ahmad, W. F., Butt, S. M., & Rahim, L. (2013). Usability Evaluation of the Agile Software Process. In *IVIC 2013 Third International Visual Informatics Conference on Advances in Visual Informatics* (pp. 640–651). Selangor, MY: Springer New York. http://doi.org/10.1007/978-3-319-02958-0_58

Wang, G., & Seibert, S. E. (2015). The impact of leader emotion display frequency on follower performance: Leader surface acting and mean emotion display as boundary conditions. *The Leadership Quarterly*, *26*(4), 577–593. http://doi.org/10.1016/j.leaqua.2015.05.007

Wang, S., Zheng, S., Xu, L., Li, D., & Meng, H. (2008). A literature review of electronic marketplace research: Themes, theories and an integrative framework. *Information Systems Frontiers*, *10*(5), 555–571. http://doi.org/10.1007/s10796-008-9115-2

Watts, D. J., & Strogatz, S. H. (1998). Collective dynamics of "small-world" networks. *Nature*, *393*(6684), 440–2.

Weathington, J. (2016). How to build a data-driven culture with emotion. Retrieved September 2, 2017, from http://www.techrepublic.com/article/how-to-build-a-data-driven-culture-with-emotion/

Webster, J., & Watson, R. T. (2002). Analyzing the Past to Prepare for the Future: Writing a Literature Review. *MIS Quarterly*, *26*(2), xiii–xxiii. http://doi.org/10.1.1.104.6570

Werder, K., Helms, R. W., & Jansen, S. (2014). Social Media for Success: A Strategic Framework. In *Pacific Asia Conference on Information Systems 2014* (p. Paper 92). Chengdu, PRC: AISeL.

Werder, K., & Maedche, A. (2017a). *Emotional Leadership of Open Source Development Teams: Examining Collaboration Structure and Collaboration Function*. Mannheim, DE.

Werder, K., & Maedche, A. (2017b). *Explaining the Emergence of Team Agility : A Complex Adaptive Systems Perspective*. Mannheim, DE.

Werder, K., Zobel, B., & Maedche, A. (2016). PDISC – Towards a Method for Software Product

DISCovery. In A. Maglyas & A.-L. Lamprecht (Eds.), *International Conference on Software Business* (1st ed., Vol. 114, pp. 47–62). Cham, CH: Springer International Publishing. http://doi.org/10.1007/978-3-319-40515-5_4

Werder, K., Zobel, B., & Maedche, A. (2017). *PDISC – A Method for Software Product DISCovery*. Mannheim, DE.

West, D., & Grant, T. (2010). *Agile Development: Mainstream Adoption Has Changed Agility*. *For Application Development & Program Management Professionals January*. Cambridge, MA, USA.

Williams, H., & Ferguson, A. (2007). The UCD Perspective: Before and After Agile. In *Agile Conference 2007* (pp. 285–290). Washington, DC, USA: IEEE. http://doi.org/10.1109/AGILE.2007.61

Winter, R. (2017). Model enablement in organizational design and enterprise engineering: the why, the how and the what. *Organizational Design and Enterprise Engineering*, *1*(1), 37–42. http://doi.org/10.1007/s41251-016-0007-7

Wolkerstorfer, P., Tscheligi, M., Sefelin, R., Milchrahm, H., Hussain, Z., Lechner, M., & Shahzad, S. (2008). Probing an agile usability process. In *Proceeding of the twenty-sixth annual CHI conference extended abstracts on Human factors in computing systems - CHI '08* (p. 2151). Florence, IT: ACM Press. http://doi.org/10.1145/1358628.1358648

Wood, R. E. (1986). Task complexity: Definition of the construct. *Organizational Behavior and Human Decision Processes*, *37*(1), 60–82. http://doi.org/10.1016/0749-5978(86)90044-0

Xiang, C., Yang, Z., & Zhang, L. (2016). Improving IS development teams' performance during requirement analysis in project—The perspectives from shared mental model and emotional intelligence. *International Journal of Project Management*, *34*(7), 1266–1279. http://doi.org/10.1016/j.ijproman.2016.06.009

Xiong, Y., & Wang, A. (2010a). A new combined method for UCD and software development and case study. In *The 2nd International Conference on Information Science and Engineering* (pp. 1–4). Hangzhou, PRC: IEEE. http://doi.org/10.1109/ICISE.2010.5690032

Xiong, Y., & Wang, A. (2010b). A new combined method for UCD and software development and case study. *The 2nd International Conference on Information Science and Engineering*, 1–4. http://doi.org/10.1109/ICISE.2010.5690032

Xu, H., Creighton, O., Boulila, N., & Demmel, R. (2013). User model and system model: the yin and yang in user-centered software development. *Proceedings of the 2013 ACM International Symposium on New Ideas, New Paradigms, and Reflections on Programming & Software - Onward! '13*, 91–100. http://doi.org/10.1145/2509578.2514737

Yang, H.-L., & Tang, J.-H. (2004). Team structure and team performance in IS development: a social network perspective. *Information & Management*, *41*(3), 335–349. http://doi.org/10.1016/S0378-7206(03)00078-8

Yin, R. K. (2008). *Case Study Research: Design and Methods* (4th ed.). Thousands Oaks, CA, USA: SAGE Publications.

Yoo, Y., & Alavi, M. (2004). Emergent leadership in virtual teams: What do emergent leaders do? *Information and Organization*, *14*(1), 27–58. http://doi.org/10.1016/j.infoandorg.2003.11.001

Zellner, A. (1962). An Efficient Method of Estimating Seemingly Unrelated Regressions and Tests for Aggregation Bias. *Journal of the American Statistical Association*, *57*(298), 348–368. http://doi.org/10.1080/01621459.1962.10480664

Zhang, P., Carey, J., Te'eni, D., & Tremaine, M. (2005). Integrating Human-Computer Interaction Development into the Systems Development Life Cycle: A Methodology. *Communications of the Association for Information Systems*, *15*(1), Article 29.

Zhang, Z.-X., Hempel, P. S., Han, Y.-L., & Tjosvold, D. (2007). Transactive memory system links work team characteristics and performance. *The Journal of Applied Psychology*, *92*(6), 1722–1730. http://doi.org/10.1037/0021-9010.92.6.1722

Appendix A: Study 1 - Additional Codes

Table 38 – List of Additional Codes that were omitted in Figure 8 and their Respective Hierarchy Level.

Level 1	Level 2	Level 3	Level 4
Process	Design Up Front	Big Design Up Front	-
Process	Design Up Front	Little Design Up Front	-
Practices	Evaluation	Customer / User Involvement	Participatory Design
Practices	Evaluation	Customer / User Involvement	Direct Involvement
Practices	Evaluation	Customer / User Involvement	User Representative
Practices	Evaluation	Usability Evaluation (Expert)	Cognitive Walkthrough
Practices	Evaluation	Usability Evaluation (Expert)	Heuristic Evaluation
Practices	Evaluation	Usability Testing	RITE
Practices	Evaluation	Usability Testing	Thinking Aloud
Practices	Evaluation	Usability Testing	Wizard-of-Oz

Appendix B: Study 1 - Practices for UCASD

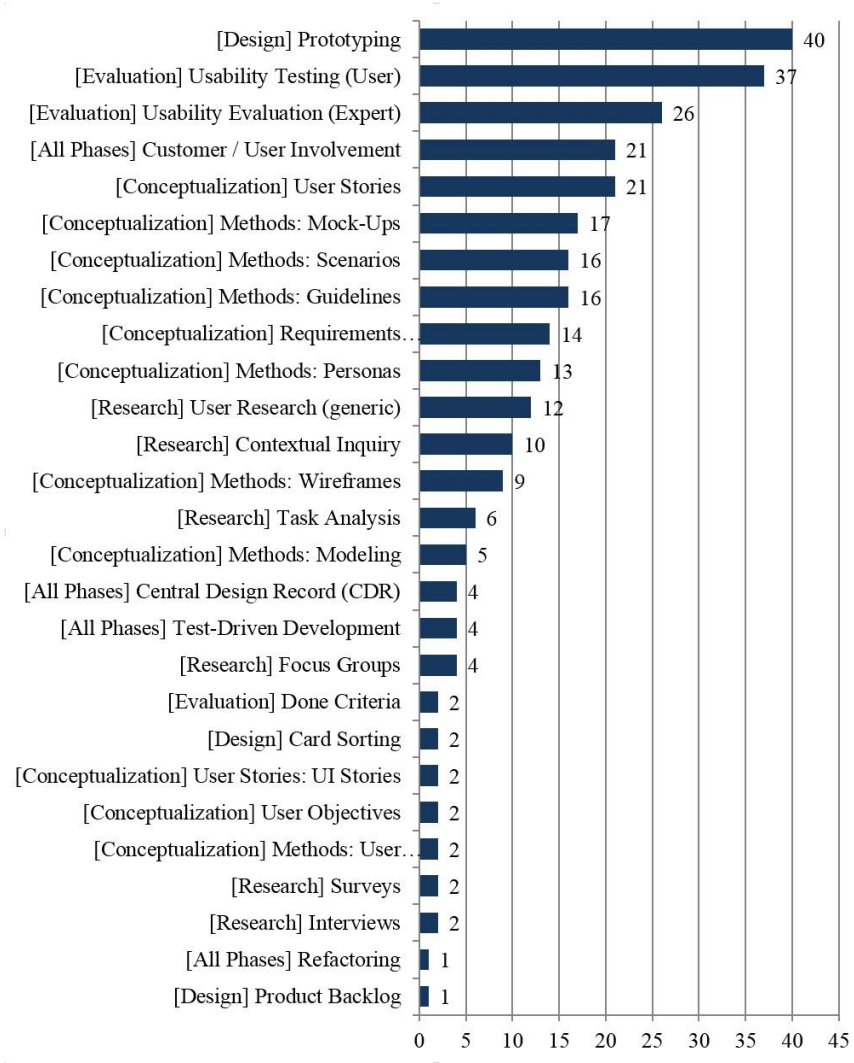

Figure 18 – Codes and Number of Articles Related to Dimension "Practices".

Appendix C: Study 1 - Search String Operationalization

Database Name	ACM Digital Library
Options	- / -
Remarks	- The ACM Digital Library does not offer advanced options for database searches. Thus, the search string had to be created manually by specifying in which metadata fields a search should be performed. - It is not possible to export the result list from the ACM Digital Library. Thus, an import of the search results into *Citavi* was omitted and the analysis for inclusion in stage II was performed directly based on the result list.
Search String	(Title: Ergonomics or Title: „Human-Computer Interaction" or Title:" Interaction Design" or Title: Usability or Title: „User Experience" or Title: „User-Centered Design" or Abstract: Ergonomics or Abstract:" Human-Computer Interaction" or Abstract: „Interaction Design" or Abstract: Usability or Abstract: „User Experience" or Abstract: „User- Centered Design" or Keywords: Ergonomics OR Keywords: „Human- Computer Interaction" OR Keywords: „Interaction Design" OR Keywords: Usability OR Keywords: „User Experience" OR Keywords: „User- Centered Design") and (Title: Agile or Title: Scrum or Title: „Extreme Programming" or Title: Lean or Title: „Crystal Clear" or Title: „Feature Driven Development" or Title: „Dynamic Software Development" or Abstract: Agile or Abstract: Scrum or Abstract: „Extreme Programming" or Abstract: Lean or Abstract: „Crystal Clear" or Abstract: „Feature Driven Development" or Abstract: „Dynamic Software Development" or Keywords: Agile OR Keywords: Scrum OR Keywords: „Extreme Programming" OR Keywords: Lean OR Keywords: „Crystal Clear" OR Keywords:" Feature Driven Development" OR Keywords: „Dynamic Software Development")
Search Results	78 Hits

Database Name	EBSCO Host
Options	- Search in *Business Source Premier* and *Academic Search Premier* - *Advanced Search* was employed - *Full Text* option was activated
Remarks	- / -
Search String	TX (Ergonomics OR "Human-Computer Interaction" OR "Interaction Design" OR Usability OR "User Experience" OR "User-Centered Design") AND TX (Agile OR Scrum OR "Extreme Programming" OR Lean OR "Crystal Clear" OR "Feature Driven Development" OR "Dynamic Software Development") AND TX ("Software Development" OR "Systems Development")
Search Results	385 Hits (335 after automated removal of duplicates)

Database Name	IEEE Xplore
Options	- Menu path: *Advanced Search → Command Search* - Search restricted to metadata
Remarks	- / -
Search String	(Ergonomics OR "Human-Computer Interaction" OR "Interaction Design" OR Usability OR "User Experience" OR "User-Centered Design") AND (Agile OR Scrum OR "Extreme Programming" OR "Lean" OR „Crystal Clear" OR "Feature Driven Development" OR "Dynamic Software Development")
Search Results	125 Hits

Database Name	ProQuest
Options	- Menu path: *Advanced Search → Command Line Search* - No date restriction - *Full Text* option activated to restrict to subscribed journals - *Peer Reviewed* option activated
Remarks	- ALL (<Search String>) to search in all metadata fields was employed
Search String	ALL ((Ergonomics OR "Human-Computer Interaction" OR "Interaction Design" OR Usability OR "User Experience" OR "User-Centered Design") AND (Agile OR Scrum OR "Extreme Programming" OR "Lean" OR "Crystal Clear" OR "Feature Driven Development" OR "Dynamic Software Development") AND ("Software Development" OR "Systems Development"))
Search Results	362 Hits

Database Name	ScienceDirect
Options	- Menu path: *Journals → Expert Search* - Sources limited to *Subscribed Sources* - *All Years* option activated - Limit by document type: *Article* - Selected Subjects: *Business, Management and Accounting*; *Computer Science*; *Decision Science*
Remarks	- FULL-TEXT (<Search String>) to search in full text was employed
Search String	FULL-TEXT ((Ergonomics OR "Human-Computer Interaction" OR "Interaction Design" OR Usability OR "User Experience" OR "User- Centered Design") AND (Agile OR Scrum OR "Extreme Programming" OR Lean OR "Crystal Clear" OR "Feature Driven Development" OR "Dynamic Software Development") AND ("Software Development" OR "Systems Development")) - Journals
Search Results	194 Hits

Database Name	SpringerLink
Options	- / -
Remarks	- The SpringerLink database has a limited search string size which did not fit the needs of the performed database search. Thus, the search string had to be split and three separate queries were submitted. Besides, the search had to be limited to title and abstract due to technical limitations. - It is not possible to export the result list from SpringerLink. Thus, an import of the search results into *Citavi* was omitted and the analysis for inclusion in stage II was performed directly based on the result list.
Search String	Search String 1 (Agile OR Scrum OR "Extreme Programming"): ab :((ergonomics or "human-computer interaction" or "interaction design" or usability or "user experience" or "user-centered design") and (Agile or Scrum or "Extreme Programming")) Search String 2 (Lean OR "Crystal Clear"): ab :((ergonomics or "human-computer interaction" or "interaction design" or usability or "user experience" or "user-centered design") and (Lean OR "Crystal Clear")) Search String 3 ("Feature Driven Development" OR "Dynamic Software Development"): ab :((ergonomics or "human-computer interaction" or "interaction design" or usability or "user experience" or "user-centered design") and ("Feature Driven Development" OR "Dynamic Software Development"))
Search Results	Search String 1: 52 Hits ; Search String 2: 06 Hits; Search String 3: 00 Hits Sum: 58 Hits

Appendix D: Study 2 – Overview of Final Articles

Table 39 – List of Primary Articles, their Quality Rating, Publications Type, and their Frequency Used within Different Challenges (Cha.), Recommendations (Rec.), and Requirements (Req.).

Reference	QR	Type	Number of		
			Cha.	Rec.	Req.
Adikari et al., 2009	5	Conference	2	1	2
Brhel et al., 2015	5	Journal	2	2	2
Fox et al., 2008	5	Conference	1	2	2
Hildenbrand and Meyer, 2012	5	Book	1	3	4
Hollis and Maiden, 2013	5	Journal	1	1	3
Inayat et al., 2015	5	Journal	1	3	1
Kajko-Mattsson and Nyfjord, 2009	5	Conference	1	1	3
Kakar and Carver, 2012	5	Workshop	1	1	1
Khurana and Rosenthal, 1998	5	Journal		1	2
Kuusinen, 2014	5	Conference	1	1	1
Nyfjord and Kajko-Mattsson, 2008	5	Book	1	2	2
Patton, 2002	5	Conference	1	1	1
Salah et al., 2014a	5	Conference	2	4	6
Salah et al., 2014b	5	Conference	2	4	4
Silva da Silva et al., 2011	5	Conference	1	2	1
Ebert, 2006	4	Journal	1	2	1
Ferre and Medinilla, 2007	4	Conference	1		2
Ferreira et al., 2007	4	Conference		2	0
Frishammar et al., 2011	4	Journal			1
Honious and Clark, 2006	4	Conference		2	1
Jain and Suman, 2015	4	Notes		1	2
Knoll and Horton, 2011	4	Conference			1
Loniewski et al., 2011	4	Workshop			1
Menor et al., 2002	4	Journal			1
Miller, 2005	4	Conference		2	1
Oliveira and Rozenfeld, 2010	4	Journal			1
Qumer and Henderson-Sellers, 2007	4	Conference	1	1	1
Rejeb et al., 2008	4	Conference	1	1	1
Sarpong and Maclean, 2012	4	Journal			1
Sibghatullah et al., 2006	4	Conference	1	3	2
Sohaib and Khan, 2010	4	Conference	2	2	2
Sperry and Jetter, 2009	4	Conference			1
Stevens, 2014	4	Journal			1
D. Sy, 2007	4	Journal		2	4
Tessarolo, 2007	4	Journal	1	1	2
Vanhanen et al., 2003	4	Conference	2	3	2
Barksdale and McCrickard, 2012	3	Journal		1	1
Cloyd, 2001	3	Journal	1	1	3
Gamble and Hale, 2013	3	Journal	1		1
Gaubinger and Rabl, 2014	3	Book			1
Heikkilä et al., 2015	3	Journal	1		1
Liskin, 2015	3	Book	1		1
Salvador et al., 2014	3	Conference	1	1	1
Williams and Ferguson, 2007	3	Conference	1	1	1

Appendix E: Study 3 - Interview Guidelines

UNIVERSITY OF MANNHEIM
BUSINESS SCHOOL

PROF. DR. A. MÄDCHE
CHAIR OF INFORMATION SYSTEMS IV –
INSTITUTE FOR ENTERPRISE SYSTEMS

Interview Protocol: User-Centered and Agile SD Teams

1. Introduction: (2min)
Karl Werder (Research Assistant) from University of Mannheim, Institute for Enterprise Systems

We are interested in design and development process of Product Teams leading to a usable and useful product and possible contingency factors.

The information gathered will be used for research purposes. Therefore the data will be anonymized. No original records will be shared with your employer.

The Interview is structured in two parts. First I would like to better understand your current situation. Following I have some specific questions certain aspects of your team, the organization, technology used and the product itself.

2. Recording Permission: (1min)
With your permission I would like to record the Interview. This would allow me to completely focus on the Interview now, only taking limited notes.

UNIVERSITY OF MANNHEIM
BUSINESS SCHOOL

PROF. DR. A. MÄDCHE
CHAIR OF INFORMATION SYSTEMS IV – INSTITUTE FOR ENTERPRISE SYSTEMS

3. Questions

3.1. Principles (25min)

Separate Product Discover and Product Creation
What information (e.g. clear product scope and product vision) have been available before the design and development work of Product v.X started?
Probe: Would initial (high-level) user research have any effect and why? (e.g. contextual inquiry, task analysis, interviews)?

Incremental and iterative development
Can you describe the development process of Product v.X?
Probe: Would the implementation of agile principles have any effect and why? (i.e. incremental and iterative work)?

Parallel Interwoven tracks
How did you manage the interplay (i.e. communicate, exchange information, collaborate, etc.) between design efforts and development work?
Probe: If the design was ready before development (of that part) starts – would it have any effect and why?
Conducting design and development efforts in parallel starts – would it have any effect and why?

Continuous Stakeholder integration
When did you involve stakeholders (i.e. Interessensvertreter or Akteure) in the design and development of Product v.X?
Probe: Receiving feedback from stakeholders throughout the entire process – would it have any effect and why?

Artifact mediated communication
How do you exchange possible design solutions within the team and with stakeholders?
What kind of deliverables are created within the development process?
Probe: What effect would the use of artifacts during the design and development have on the team's communication (internally + externally) ?

UNIVERSITY OF MANNHEIM
BUSINESS SCHOOL

PROF. DR. A. MÄDCHE
CHAIR OF INFORMATION SYSTEMS IV – INSTITUTE FOR ENTERPRISE SYSTEMS

3.2. Contingency Factors (25min)

Team:
Stability / familiarity - Have you been familiar with all other team members before the start of design and development work for Product v.X?
Cross-functionality - Was the team equipped with a set of cross-functional skills and expertise?
User knowledge - What did you know about the user (base)?
User access - In case of questions, was it easy for you to get direct access to representative users? (as opposed to many "man-in-the-middle")
Goal interdependence - Was your success tied to the success of the team?

Product:
Complexity - Would you characterize the product as complex (front-end vs back-end)?
Interdependency - How often do you depend on the information/output of other people to complete your tasks?
(Dev. tools supporting Product Design?)

Technology:
Access - Do you have easy access to necessary tools and technologies to carry out your work?
Flexibility - Are you free to choose what development and communication tools to use for your work?
Channel/ knowledge base - Can you easily find and access Product v.X related information during design and development (e.g. requirement, sketches, models, etc)?

Organizational
Management support - Did management (and/or other team members) recognize the value of usability?
Budget - Was there sufficient budget allocated to the design of the product?

UNIVERSITY OF MANNHEIM
BUSINESS SCHOOL

PROF. DR. A. MÄDCHE
CHAIR OF INFORMATION SYSTEMS IV – INSTITUTE FOR ENTERPRISE SYSTEMS

Notes:
Separate Product Discover and Product Creation:
_____
_____
_____
Incremental and iterative development:
_____
_____
_____
Parallel Interwoven tracks:
_____
_____
_____
Continuous Stakeholder integration:
_____
_____
_____
Artifact mediated communication:
_____
_____
_____

Team:
_____
_____
_____
Product:
_____
_____
_____
Technology:
_____
_____
_____
Organizational:
_____
_____
_____

UNIVERSITY OF MANNHEIM
BUSINESS SCHOOL

PROF. DR. A. MÄDCHE
CHAIR OF INFORMATION SYSTEMS IV – INSTITUTE FOR ENTERPRISE SYSTEMS

Background Information (5min)

Product and Release:

Budget (in thousand € /year):
_____

Start of product design and development:_____

End of product design and development:_____

Group

What was the group size of the design and development team and their roles?
_____

Number of team members located at each site:
_____

Person

How much experience do you have in (years) ....

- ....Usability Engineering / Interaction Design?
  _____
- ...Software Development?
  _____
- ...Agile software development?
  _____

How many years are you in your current role?
_____

Appendix F: Study 4 – Extracting Emotions

For our data collection, we rely on secondary data. Contrary to survey data, this data is not generated for the purpose of scientific analysis. Therefore, we do not capture any perceived data points. However, we extract and transform existing textual data using natural language processing (NLP) in order to generate additional measurements. For our data processing, we rely on R, an open source programming language and statistical computing software, that is easily accessible and benefits from a larger community. Latter continuously extends the software with additional statistical analysis packages. For the extraction of sentiment in general and discrete emotions in particular based on textual data, we identify two prominent packages within the R community, the sentiment package (Jurka, 2012) and the Syuzhet package (Jockers, 2016). We present a brief comparison in Table 40.

Table 40 – Comparison of Selected Sentiment and Emotion Extraction Packages in R.

Algorithm	Sentiment R Package	Syuzhet R Package
Authors	Jurka, 2012	Jockers, 2016
Lexicon	WordNet-Affect sentiment lexicon	NRC Lexicon
# of Words in Lexicon	1512	14182
Example of lexicon use	Jurado and Rodriguez, 2015	Mohammad and Turney, 2013
Number of discrete Emotions	six classes of emotions	eight classes of emotions
Emotions	Anger, disgust, joy, surprise, fear, and sadness	anger, fear, anticipation, trust, surprise, sadness, joy, and disgust
Source of Emotions	Ekman and Davidson, 1994	Plutchik and Kellerman, 1980
Further detailed steps		-Filter for words (excl. non-letter characters) -Remove duplicate words -Extract emotion for each word, non-existent words score zero -Sum up emotions for each word to form final emotion

Following some preliminary analysis and prior studies, we identified two main challenges that when addressed enhance the accuracy of the emotional extraction (Jurado & Rodriguez, 2015). First, we want to enhance the results to consider negations. Second, we exclude error related emotions as a context specific adjustment. An analysis of 60 randomly selected comments suggests that 29 comments include negation and 10 comments include error messages.

In order to identify negations, we conducted additional steps before summing up the emotions. We conducted additional sentence splitting, tokenization and part-of-speech tagging through the use of the Stanford CoreNLP toolkit, as suggested by prior studies (Manning et al., 2014). Following the indication of the part-of-speech analysis, we detect negations for adjectives, adverbs, and verbs. Using a four-word negation window, we identify negations that invert the extracted emotion, suggesting the use of the

dimensional counterpart. Once the emotions are summed up per comment, we divide the emotion number by the number of sentences to account for text length.

In order to identify errors, we collected 1163 compiler errors and exception messages as well as 129 common language exception messages as a basis to develop a list of keywords detecting the same. Based on the analysis of these messages, we derived the following list of domain specific keywords associated with error messages:

- attach, attachment, bash, build, broke, merge, error, case, catch, compact, console, content, inconvenient, render, bug, static, generic, variable, main, enhanced, default, production, remove, supported, unsupported, execution, gateway, system, error, argument and action.

While the Syuzhet package replaces punctuations with a space as its default, we replaced punctuations with an empty string following prior studies (Jurado & Rodriguez, 2015). Therefore, we maintain the structural function of punctuations. A comparison of both algorithm suggests that our adjustment lead to less emotional extractions, which is consistent with prior studies (Jurado & Rodriguez, 2015). Using Microsoft TypeScript (44413 total comments) as an example project, we find that the percentage of neutral emotions extracted increase from 30% to 50%, as a result of our adjustment. Following, we present two examples indicating the benefits of our adjustments:

Example 1:

- FYI: Please don't use 'mdl-' prefixes in custom additions. Prefix the project or company name, or something else. That increases the likelihood of collisions in future releases if we decide to use the same component name. While it is on the developer for using a namespace they **have no real** need using. It still can **create** a hard upgrade later on. Best **avoid** that since it is much easier to do it from the start.

Table 41 – Method Comparison of Emotional Tagging – Example 1.

Method	Joy	Sad	Anger	Fear	Anticipation	Surprise	Trust	Disgust	Positive	Negative
Syuzhet	1	0	0	1	1	0	1	0	2	1
Syuzhet + Adjustments	1	0	0	1	0	0	0	1	1	2

As it is noted in the example, the bold words express emotions and were extracted by the new tailored approach. The Syuzhet approach did not consider negation; therefore, the word "real" was classified into trust and positive. While, the tailored approach has classified it into disgust and negative because it contains

negation. Further, Syuzhet classified the last word in the comment "start" into anticipation and the implemented approach did not pick it. The tailored approach did not select it, because it is a noun. Ignoring the word "start" was beneficial for improving the outcomes, since there was no real emotions associated with it; based on the sentence's context. Table 41 presents an overview of the results for example 1.

Example 2:

- Well, we don't really have the resources to maintain a powerpc option. But I encourage you do make your branch *the* PPC branch of Homebrew. We'll even link you from the **relevant** places. Sound **good**?

Table 42 – Method Comparison of Emotional Tagging – Example 2.

Method	Joy	Sad	Anger	Fear	Anticipation	Surprise	Trust	Disgust	Positive	Negative
Syuzhet	3	0	0	0	1	1	4	0	5	0
Syuzhet + Adjustments	2	0	0	0	1	1	3	0	3	0

The bold words (”good”, ”relevant” and ”encourage”) are the one picked by the tailored approach and they do reflect true emotions from the text. The Syuzhet approach added the underlined words shown in the example. These extra words misled the overall emotions, as they do not reflect true sentiments in the sentences. It is worth mentioning that the underlined words are nouns. Hence, the decision of removing nouns during sentiments extraction is very beneficial in this example. Table 42 provides an overview of the results for example 2.

Curriculum Vitae (short) – Karl Werder

Ph.D. Student in Information Systems, University of Mannheim, Mannheim (DE)	**Since 2014**
M.Sc. in Business Informatics, Utrecht University, Utrecht (NL)	**2011 – 2013**
B. Sc. in Computer Science, Conservatoire National des Arts et Métiers, Paris (FR) and University of Applied Science, Darmstadt (DE)	**2008 – 2011**
Chamber of Commerce Diploma IT-Specialist, Ferdinand-Braun-Schule, Fulda (DE)	**2004 – 2007**
University-Entrance Diploma (vocational), Konrad-Zuse-Schule, Fulda (DE)	**2002 – 2004**

Publication List – Karl Werder

Publications related to this thesis:

Liu, X., Werder, K., Maedche, A. (2016). A Taxonomy of Digital Service Design Techniques. In *Proceedings of the 37th International Conference on Information Systems*, Dublin, IE: AISeL.

Werder, K., & Wang, H. Y. (2016). Towards a Software Product Industry Classification. New Trends in Software Methodologies, Tools and Techniques: *Proceedings of the 15th International Conference on Intelligent Software Methodologies, Tools and Techniques* (pp. 27 – 37), Larnaca, CY: IOS Press.

Werder, K. (2016). Team Agility and Team Performance – The Moderating Effect of User Involvement. In *Proceedings of the 30th European Conference on Information Systems* (Paper 3), Istanbul, TR: AISeL.

Werder, K., Zobel, B., & Maedche, A. (2016). PDISC–Towards a Method for Software Product DISCovery. In *Proceedings of the 7th International Conference of Software Business* (pp. 47-62). Ljubljana, SI:Springer.

Brhel, M., Meth, H., Maedche, A., & Werder, K. (2015). Exploring principles of user-centered agile software development: A literature review. *Information and Software Technology*, 61, 163-181.

Werder, K., Haake, P., & Maedche, A. (2014, September). Usability Readiness of German Software SMEs-Three Segments and their Characteristics. In *Mensch & Computer* (pp. 185-194). Munich, DE:Springer.

Stevens, G., Burmester, M., Brandenburg, S., Döbelt, S., Kugelmeier, D., Schlömer, I., Schmidt, R., Thüring M., Werder, K. & Daniel, Z. (2014). Usability für die betriebliche Praxis. *Mensch & Computer – Workshopband* (p. 143). Munich, DE:Springer

Further Publications:

Morschheuser, B., Werder, K., Hamari, J., & Abe, J. (2017). How to gamify? Development of a method for gamification. To appear in *Proceedings of the 50th Annual Hawaii International Conference on System Sciences*. Big Island, HI: IEEE.

Feil, S., Kretzer, M., Werder, K., & Maedche, A. (2016). Using gamification to tackle the cold-start problem in recommender systems. In *Proceedings of the 19th ACM Conference on Computer Supported Cooperative Work and Social Computing Companion* (pp. 253-256). Portland, US:ACM.

Werder, K., Helms, R. W., & Jansen, S. (2014). Social Media for Success: a Strategic Framework. In *Proceedings of 18th Pacific Asia Conference on Information Systems* (Paper 92). Chengdu, PRC: AISeL.

Helms, R. W., & Werder, K. (2013). Who Reads Corporate Tweets? Network Analysis of Follower Communities. In *Proceedings of the 19th Americas Conference on Information Systems* (pp. 3998-4009). Chicago, IL: AISeL.

Working Papers:

Werder, K., Zobel, B. & Maedche, A. PDISC - A Method for Software Product Discovery

Werder, K. & Maedche, A. Explaining the Emergence of Team Agility: A Complex Adaptive Systems Perspective

Werder, K. & Maedche, A. Emotional Leadership of Open Source Development Teams: Examining Collaboration Structure and Collaboration Functions.

Werder, K. & Helms, R. W. Social Media Strategy: Conceptualization, Typology, and Implications.

Li, Y., Maedche, A., Balasubramaniam, B. & Werder, K. Software Development Process Ambidexterity and Project Performance: A Coordination Cost and Coordination View

Heckmann, C., Werder, K. & Maedche, A. Ambidexterity in Information Systems Research – Overview of Conceptualizations, Antecedents and Outcomes

Eidesstattliche Versicherung

gemäß § 6 Abs. 1 Ziff. 4 der Promotionsordnung des Karlsruher
Instituts für Technologie für die Fakultät für Wirtschaftswissenschaften

1. Bei der eingereichten Dissertation zu dem Thema *Teams in Agile Software Development: Design Principles and Examination of Human Factors* handelt es sich um meine eigenständig erbrachte Leistung.

2. Ich habe nur die angegebenen Quellen und Hilfsmittel benutzt und mich keiner unzulässigen Hilfe Dritter bedient. Insbesondere habe ich wörtlich oder sinngemäß aus anderen Werken übernommene Inhalte als solche kenntlich gemacht.

3. Die Arbeit oder Teile davon habe ich *bislang nicht* an einer Hochschule des In- oder Auslands als Bestandteil einer Prüfungs- oder Qualifikationsleistung vorgelegt.

4. Die Richtigkeit der vorstehenden Erklärungen bestätige ich.

5. Die Bedeutung der eidesstattlichen Versicherung und die strafrechtlichen Folgen einer unrichtigen oder unvollständigen eidesstattlichen Versicherung sind mir bekannt. Ich versichere an Eides statt, dass ich nach bestem Wissen die reine Wahrheit erklärt und nichts verschwiegen habe.

Karlsruhe, den *18.04.2017*

Karl Werder