Software product innovation in agile usability teams: an analytical framework of social capital, network governance, and usability knowledge management

Jeremy T. Barksdale* and
D. Scott McCrickard

Department of Computer Science,
Center for Human Computer Interaction,
Virginia Tech, 2202 Kraft Drive,
Blacksburg VA 24060-0902, USA
E-mail: barksdale@vt.edu
E-mail: mccricks@cs.vt.edu
*Corresponding author

Abstract: As the practice of software engineering matures, project teams are leveraging the expertise of those with a background in other domains such as usability. This paper proposes a model and agenda for understanding and improving social interaction on agile usability software teams. We argue that social interaction on multidisciplinary agile usability teams, as a means to integrating the software development and usability domains, impacts how usability knowledge is managed, and thus, software product innovation. This work contributes:

1 a background, analysis, and discussion of the various agile usability integration strategies to-date
2 a model for investigating the social interaction in agile usability teams
3 a research, practice, and policy agenda for future work toward improving the social interaction in multidisciplinary agile usability software teams.

Keywords: agile usability; agile software development; ASD; usability engineering; social capital; network governance; usability knowledge management.

Reference to this paper should be made as follows: Barksdale, J.T. and McCrickard, D.S. (2012) 'Software product innovation in agile usability teams: an analytical framework of social capital, network governance, and usability knowledge management', *Int. J. Agile and Extreme Software Development*, Vol. 1, No. 1, pp.52–77.

Biographical notes: Jeremy T. Barksdale is a PhD student in the Department of Computer Science at Virginia Tech. He currently works as a Usability Engineer at a software company, and has previous experience working as a Project Manager and Requirements Engineer in the public and private sectors. His primary research interests include the social interaction between developers and usability experts in agile software development teams and, more generally, how technology can support improved team collaboration. He received his MS in Computer Science in 2002, and his BA in Broadcast Production in 2000 from North Carolina A&T State University.

D. Scott McCrickard is an Associate Professor in the Department of Computer Science at Virginia Tech, and a member of the Virginia Tech's Center for Human Computer Interaction. His research interests include human-computer interaction, design methods, user and task modelling, and interface design for mobile computing devices. He received his MS and PhD in Computer Science from Georgia Tech in 1995 and 2000, respectively, and his BS in Mathematical Science from the University of North Carolina, Chapel Hill, in 1992.

1 Introduction

Traditional software teams, comprised of software engineers using waterfall-like methodologies, were limited in their ability to develop quality software on schedule and within budget. Toward resolving these limitations, a team of software experts created the Agile Manifesto that would guide developers to be more effective through a set of principles and values (Fowler and Highsmith, 2001). However, the manifesto did not consider disciplines outside of software engineering.

As the practice of software engineering matured, software teams also began leveraging the knowledge and skills of experts in other areas, such as usability engineering (UE) and interaction design (IxD) to address the limitations of teams without diversity. An advantage of diverse teams is that they can produce better quality software in a shorter time period given the decreased time required by experienced members and the parallel work activity.

However, multidisciplinary software teams face barriers, such as effectively transferring knowledge within the team. Encoding and decoding during knowledge transfer is complicated because members are less likely to have a shared vocabulary and meaning. For example, software engineers and usability engineers have a different understanding of what constitutes a scenario. A scenario for a software engineer will most likely include a formal narrative of the user's activity relative to the system; the usability engineer's scenario will likely take on a less formal narrative form – centred on the user's actions. Although both disciplines use the term 'scenario', the difference in meaning can interfere with effective collaboration.

Interaction barriers can compromise negotiation among members when making system and user experience (UX) decisions. Agile teams constantly prioritise which aspects of the software will get development effort. Given that developers are typically responsible for implementation, if negotiation breaks down between the two groups (e.g., due to miscommunication), it often results in developers bypassing the input of usability experts and potentially compromising the product's usability and innovation.

Previous research has investigated strategies for improving the interaction between software developers and usability experts in agile usability teams. Such strategies include technical process integration (e.g., extreme scenario-based design, XSBD), the sharing of practices (e.g., daily stand-up meetings), and technology integration (e.g., design and development tools). Although there is benefit to these approaches, focus is needed on socio-cognitive interactions.

This research aims to address the interaction-related problems in agile usability teams by exploring how social capital and social network governance contribute to effective management of usability knowledge on agile usability software teams. It also seeks to offer practical guidance on designing cohesive agile usability teams. It is anticipated that the awareness gained from this research will empower teams to modify their dynamics to achieve effective usability knowledge management toward product innovation.

This work contributes:

1 a background from a review of the literature, thematic analysis, and discussion of the various agile usability integration strategies to-date

2 a model for investigating the social interaction in agile usability teams

3 a research, practice, and policy agenda for future work toward improving the social interaction in multidisciplinary agile usability software teams.

2 Agile usability

2.1 Agile software development

A weakness of traditional waterfall approaches was the difficulty to accommodate regular customer feedback, which affected whether the product sufficiently served the client's needs. Requirements were elicited at the beginning of the project during the requirements phase, and incorporating changes based on customer feedback became increasingly infeasible as the development lifecycle progressed. Potential problems arose from the client's changing needs – often because the client was unable to sufficiently communicate their needs without a tangible reference, or because the software team simply misunderstood the client's needs. As stakeholders begin to see and experience the character of the software, they gain a clearer understanding of the needs and wants. This inability to flexibly adjust during development impacted how well the final product met the customer's needs.

The traditional approaches also resulted in software products that ran over budget or were not completed on schedule. For reasons similar to those previously mentioned (e.g., an inability to identify clear needs upfront), the software team could not accurately estimate the required cost and time it would take to develop a product. Essentially, there was risk given the effort was too complex to fully estimate at the beginning of the project. For example, teams had difficulty sufficiently estimating the appropriate team size, the complexity of the algorithms, or the time to resolve development environment configuration management issues that arose. Each of these, and many other points of failure in a development project, erodes the team's ability to control software cost, quality, and schedule (Beck and Andres, 2004).

Agile was one answer to addressing these quality, cost, and schedule challenges. A meeting among 17 software experts in 2001 resulted in a manifesto that outlined a set of values and principles for developing software as an alternative to the traditional development approaches (Fowler and Highsmith, 2001). Software developers and methodologists use the manifesto as a vision for how an agile team should function.

Agile approaches (Highsmith and Cockburn, 2001), such as eXtreme programming (XP) (Beck and Andres, 2004), Scrum (Schwaber, 2004), and others (Ambler, 2008) established a vision for minimising the amount of big upfront planning that was required for a product. For example, XP provides developers a set of rules, values, and practices (Beck and Andres, 2004). In Scrum, emphasis is on the general project management activities versus lower level development activities. Common to both, and other agile processes, is the incremental and iterative progression of product development. The scope of work is allocated across time such that the product is incrementally delivered over a number of iterations and release cycles. In doing so, the team is better able to estimate the development time required, associated costs, and is better able to accommodate change requests. Also, the customer is in a better position to provide feedback since they are able to envision the system incrementally throughout development. More opportunities are available for users to provide feedback since the development team deploys a working product at regular increments.

Teams have taken various courses of action toward becoming agile since the establishment of the manifesto. For example, some teams have adopted the ‘spirit’ of agility by employing a subset of the values and principles, some have rigorously applied agile principles and practices since inception, while others transition into agile more progressively to mitigate organisational and team culture shock. These methods of adoption are acceptable given that different organisations and teams have different objectives and levels of need for agility.

On the surface, the advantages of agile appear to satisfy the major drawbacks of the traditional approaches. However, although the software experts that formulated agile had the best intentions, a key component in the development process was superficially considered – the user. The primary stakeholder that was given attention in agile was the customer, which in many cases is different than the users. Hence, by only understanding the customer’s perspective, the software is still likely to not satisfy those who are purchasing and using the software. Only the users can provide the user’s perspective, and in some cases, the customer is merely guessing what the user wants or what they want without regard for their users.

2.2 Software usability

The user has commonly been considered, in some capacity, in software engineering via design decisions with the adoption of the Unified Modelling Language (UML) (Booch et al., 2005). UML is the integrated work of three separate modelling languages developed for object-oriented analysis and design (OOAD) (Larman, 2002). Modelling the use of the system helps developers better decompose the system, eases development effort, and provides alternative views (Kruchten, 1995) of the system as its complexity increases. In UML, the user is considered via use case diagrams with actors and system components, use cases that detail the interaction between the actors and the system, and usage scenarios that list the tasks users need to accomplish. However, analysis of system use is not equivalent to analysis of system usability, and emphasis was primarily on the use of the system, and indirectly the user. Artefacts were commonly developed by software experts with a system focus, in consultation with the customer, and portrayed the user as a mechanical entity – compromising the ability to gain a richer awareness of the user’s experience and preferences.

With the emergence of human computer interaction (HCI) and its research over the past 40 years, usability has grown into a discipline with methods and practices that keep the user in the forefront during software design and development. The increased understanding gained about users in computing (e.g., awareness about their mental models, cognitive load, and tasks) provided deeper insight to software teams about how software can best support their needs. Research also advanced insight about what is visually appealing to users and how software can become more pleasant to their senses. This progress resulted in the establishment of methods such as scenario-based design (SBD) (Rosson and Carroll, 2002), UX design (Law et al., 2008; McClelland, 2005), user interface (UI) design (Nielsen, 1993a), IxD (Löwgren and Stolterman, 2004), UE (Mayhew, 1999; Nielsen, 1993b; Rosson and Carroll, 2002), user-centred design (UCD) (Fox et al., 2008; Göransson et al., 2003; Hussain et al., 2008).

Depending on the team, the associated roles might include a UI designer that is concerned with establishing a visually appealing UI, a UE who ensures the user is able to effectively carry out their intended tasks efficiently, and a UX designer who more broadly aims to make the user's overall experience rewarding. Collectively, those in these roles are categorically referred to as usability experts throughout this research given their emphasis on usability. Common across these approaches and roles is the design of what the UXs – in essence, a greater focus on the user than the system – and the advocating for the users' needs and desires during software development.

Changes in software development have also occurred to accommodate usability. Specifically, development frameworks (such as Oracle's Java Spring Framework, 37Signal's Ruby on Rails, and Microsoft's .NET Framework) that use architectural patterns such as model-view-controller, model-view, and model-view-presenter make it easier for developers to organise their code and separate system and user concerns so they can focus their attention on the important details of system and user design and implementation.

2.3 Agile usability

Usability progressively made its way into agile software teams, yielding what is currently known as agile usability – a term that evolved as the focus of usability in agile development increased.

Software practitioners needed a way to satisfy customers through timely releases, such that changes were feasible without compromising software quality, going over budget, and taking longer than scheduled. Usability experts wanted (and needed) to be included in the process to ensure the usability of the software product. The first need was addressed with the establishment of agile. The second need was satisfied as software engineers realised the need to place more emphasis on the user.

Constantine was among the first, in 2002, to identify that the agile manifesto did not directly consider usability. He argued for the incorporation of usability through his usage-centred design (UC-D) approach – a card-based modelling and decision making process (Constantine and Lockwood, 2002, 2003). From this point onward, research continued to seek ways to integrate the usability agile environments. In some cases, the goal was to increase the software developer's attention on the user by providing to the software team relevant principles and guidelines. In other cases, the argument was made for a separate usability expert role (or team) on the software project.

3 Agile usability integration strategies

The integration of agile and usability is a relatively new research area. Since 2002, various approaches to integration have been discussed among researchers and practitioners. Based on a review of the literature, research on agile usability can be organised within five themes: the adoption of practices by one or both areas, the combination of agile and usability processes, communication between technology used in each domain, incorporation of team members with the necessary skill-set, and integration with greater focus on their interaction.

3.1 Literature review methodology

Over the past decade, agile usability has grown into what this research identifies as five key directions. Table 1 shows the integration strategies of agile usability relative to each year since its emergence in 2002 along with key authors and citations of their publications. The 'no data' category means that an integration category was indeterminable from the publication. The 'none' category means that no integration was presented in the publication, which could result from the article discussing agile usability as a topic but not proposing a method of integration. The primary goal of this literature review was to identify the various turns agile usability has taken since its inception. Hence, the review is not meant to be exhaustive, but a representation of the trends over time as a way to acknowledge past research on the topic and reveal an area of need for future research in agile usability.

Table 1 Summary of integration strategies, related work, and key authors

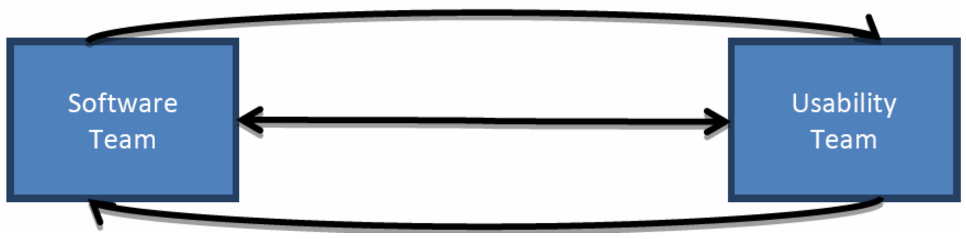
Integration type (%)	*Key authors with related work*
None (14.7%)	Constantine and Lockwood (2002), Hudson (2005), Sharp et al. (2006), Ferreira et al. (2007b), Patton (2007), Lee et al. (2007), Federoff et al. (2008), Bygstad et al. (2008) and Hussain and Slany (2009a, 2009b)
Practices (23.5%)	Ferre et al. (2005a, 2005b), Meszaros and Aston (2006), Chamberlain et al. (2006), Anwar (2006), Parsons et al. (2007), Memmel et al. (2007a, 2007c), Detweiler (2007), Sy (2007), Wolkerstorfer et al. (2008), Hussain et al. (2008, 2009), Evnin and Pries (2008), Sy and Miller (2008), Obendorf and Finck (2008), Fox et al. (2008), Ambler (2008), Adikari et al. (2009), Miller and Sy (2009), Budwig et al. (2009), Benigni et al. (2010) and Sohaib and Khan (2010)
Process (4.4%)	Patton (2002a, 2002b), Constantine (2002), Hansson (2002), Blomkvist (2005), Lee (2006, 2010), Miller (2006), Lee and McCrickard (2007), Memmel et al. (2007b), Haikara (2007), Duchting et al. (2007), Paelke and Nebe (2008), Lee et al. (2009), Carvalho (2010) and Paelke and Sester (2010)
Technology (2.9%)	Memmel et al. (2008) and Nunes (2009)
People (8.8%)	McDonald and Welland (2003), Ghosh (2004), McInerney and Maurer (2005), Lievesley and Yee (2006), Ferreira et al. (2007a) and Singh (2008b)
Social (11.8%)	Tai (2005), Memmel et al. (2007d), Brown et al. (2008), Ungar and White (2008), Ungar (2008), Barksdale et al. (2009), Ferreira et al. (2010) and Barksdale and McCrickard (2010)

The search space included 'agile usability software' in the computer science and business categories for each year, singularly, between 2000 and 2010, inclusive. Google Scholar was used as the search engine to identify related work given its broad reach and coverage of the most relevant publication databases for this topic. The search returned over 5,200 results and resulted in a review of over 65 key publications. Each publication was abstracted by extracting its title, authors, publication venue and year, relevance to the topic, evidence strength, integration category, integration approach, key argument(s)/summary, integration rationale, findings, future work, and any miscellaneous comments. Descriptive statistics were computed and a thematic analysis was conducted to better understand the gaps in the agile usability research area.

3.2 *Practices integration*

Practice integration (Figure 1) was found to be one of the most common approaches to agile usability integration. It is defined here as integration that occurs through the adoption of principles or practices from another field. For example, it occurs when an existing agile process is supplemented with usability practices, but does not entail the complete merging of independent processes.

Figure 1 Practices integration strategy (see online version for colours)



3.2.1 *Agile incorporating usability*

One approach to integrating practices is to incorporate usability practices into agile methods and teams. In this case, practitioners augment their agile methods to include some of the important usability practices. Meszaros and Aston (2006) argue that "Some design up front provides better guidance to the development team and provides earlier opportunities for feedback". They discuss their experience with incorporating usability testing into an XP project by building paper prototypes and conducting wizard-of-oz testing. The project manager and agile coach developed the paper prototypes, the business lead conducted the usability test sessions, and members of the development team acted as the computer and played the role of the help system or observed participants. They found that integration was easy, that usability testing ensured that all work was accounted for and prevented last minute essential scope creep, and that it ultimately resulted in a significant reduction of usability rework.

3.2.2 *Usability incorporating agile*

Another approach to practices integration is the incorporation of agile practices into usability methods. In this case, this entailed tailoring usability methods to become more agile-like. Sy (2007) argues that by adjusting how UCD is conducted, the team was able to harness its power to the agile characteristics of speed, responsiveness, and high implementation quality. They tailored their approach to conducting usability tests, interviews, and contextual inquiry to fit within the constraints of the agile framework. Hence, although the usability methods were ultimately integrated into the agile framework, the usability methods were adjusted to incorporate agile practices by decreasing the time required for, and granularity of, usability investigations. This was later implemented by synchronising the agile and UCD activities for efficiency. They found that the new agile UCD method produced better-designed products than the waterfall versions of the same techniques by narrowing the gap between evaluation and incorporation of changes.

3.2.3 Mutual sharing

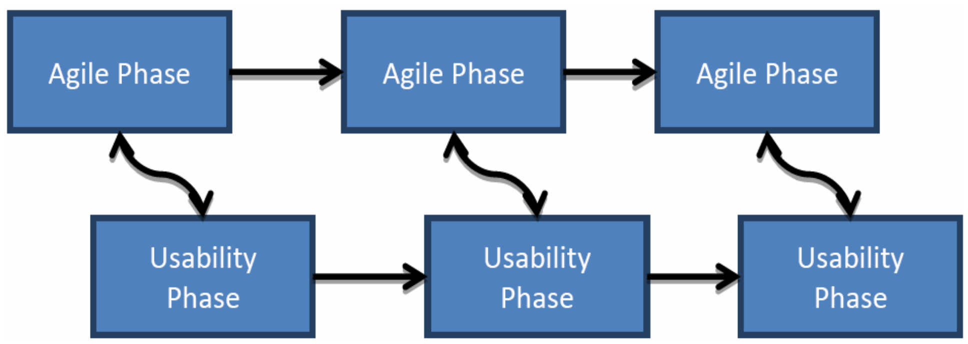
In the most collaborative sense, agile and usability methods incorporate practices from each other to move the two areas closer to one another. From an agile practitioner's perspective, Ambler (2008) argues that UX is important to software development, and that both agile software developers and UX professionals need to adjust for successful integration to work. He recommended that UX professionals go beyond UX in their skillset, become embedded in agile software development (ASD) teams, give agile approaches a chance, and start looking beyond XP when tailoring practices. His suggestions to the agile community were to learn UX skills, to accept that usability is a critical quality factor, and to adopt UI and usage style guidelines.

Parsons et al. (2007) offer strategies for incorporating key practices from each area into agile and usability methods based on their practical experiences. They argue that agile methods have not typically incorporated HCI and usability techniques and tools into their software development processes, and that it is possible to integrate certain practices from UE into an agile approach. Among other strategies for mutual practices integration, they recommend using iterative development throughout the lifecycle, merging user scenarios with stories, and allowing testing of the UI in the user context regularly throughout all phases. They also found that the following ensures high HCI quality: co-locating a UI expert with the development team, letting the product manager-owned backlog control the development process, regular assessment of the application by an external HCI consultant, and making supportive design and technology choices.

3.3 Process integration

The integration of agile and usability processes is another common approach to integrating agile and usability efforts (Figure 2). In this case, independent agile and usability processes are combined and synchronised to provide a single newly designed agile usability process. Although there is seemingly significant overlap between process and practices integration, the difference hinges on whether two completely separate processes were merged into a single process versus picking techniques from a process.

Figure 2 Process integration strategy (see online version for colours)



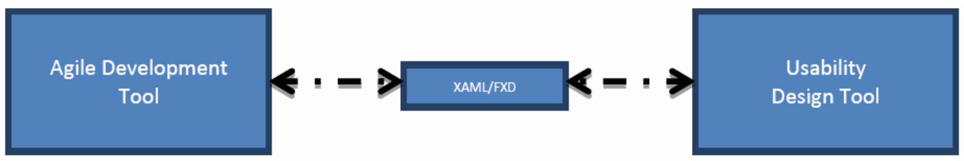
Constantine (Constantine, 2002; Constantine and Lockwood, 2002) advocates for the use of his UC-D process in coordination with agile methods such as XP. He argues that his method is a natural adjunct to an effective agile process (rather than an end-to-end development process) because it is card-based (like story card use in XP), lightweight, and iterative and incremental in nature. Patton (2002a) later instantiates Constantine's UC-D approach in an agile environment, arguing that IxD is a valuable component of any software development process and that it happens whether the intention is there or not. Patton found that although constant collaboration was exhausting, the team's tacit knowledge was 'irreplaceable'. He also noted that the UC-D task cards were useful during testing and other points of reference.

Alternatively, Lee et al. (2009) integrated the SBD process and XP. They argued that there is a need to understand the similarities and differences between XP and SBD as a means to addressing the integration problem. A comparative analysis of the core principles of XP and SBD lead them to the development of XSBD. They found that maintaining and collectively agreeing on a prioritised list of design goals will help to resolve conflicts, claims are effective for capturing design rationale, enforced and opportunistic synchronisation is important, and that a central design record (CDR) – used to support synchronisation of activities – can help with developing a cohesive interface.

3.4 Technology integration

Technology integration (Figure 3) implies that the underlying coordination between the agile development and UE activities occurs through the use of technology. An example of technology integration is when the designer and developer can perform their task using two independent software components or applications that communicate using a common data exchange format to integrate their output. Pyla (2007) developed the Ripple project development environment to foster communication between the software engineering and UE roles. He argued for the need of a connection between SE and UE lifecycles to support communication among roles given they have different levels of iteration and evaluation, different terminology, and requirements representation.

Figure 3 Technology integration strategy (see online version for colours)



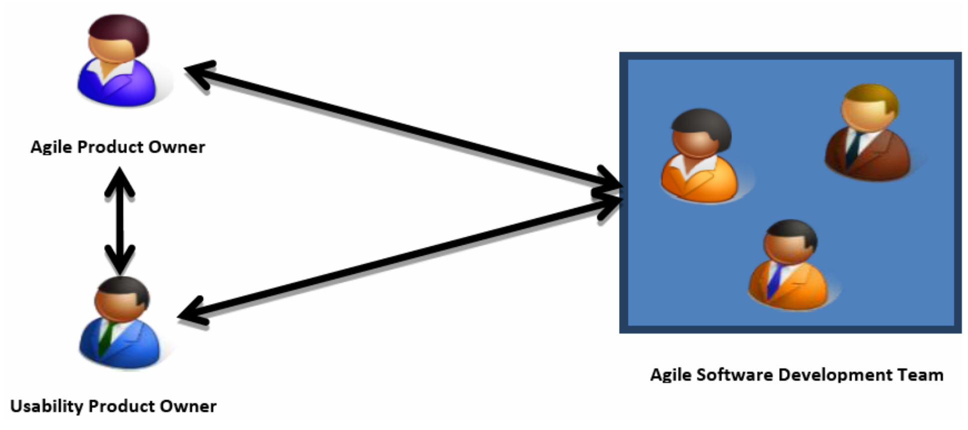
Industry solutions to technology integration entail the use of a declarative mark-up language that facilitates data exchange between the designer's software and the developer's software. For example, when the designer creates an artefact (such as a UI), the code is automatically written in the background. The developer can then incorporate that component directly into the system code they have written. A common example includes Microsoft's eXtensible Markup Language (XML)-based language (eXtensible Application Markup Language, or XAML) implemented by their Windows Presentation Foundation (WPF). Microsoft Expression Blend for creating UI elements and animations (MacVittie, 2006) and Microsoft Visual Studio for system development both utilise XAML as a data interchange format for UI development.

On a smaller scale, Oracle provides similar functionality through the use of a JavaScript Object Notation (JSON)-based declarative markup language – called JavaFX Data (FXD) – as the data interchange format. The JavaFX Production Suite is used, in conjunction with Adobe Illustrator and Photoshop by designers to construct UIs. The Netbeans integrated development environment (IDE) is used by developers to develop system code (Clarke et al., 2009). Design artefacts can be exported and incorporated into the system code via FXD.

3.5 *People integration*

People-focused approaches achieve integration by changing the team's personnel or composition to obtain the required talent. This typically means, for example, adding a designer to the team, but not necessarily specifying an integrated process or set of practices that will be used (Figure 4).

Figure 4 People integration strategy (see online version for colours)



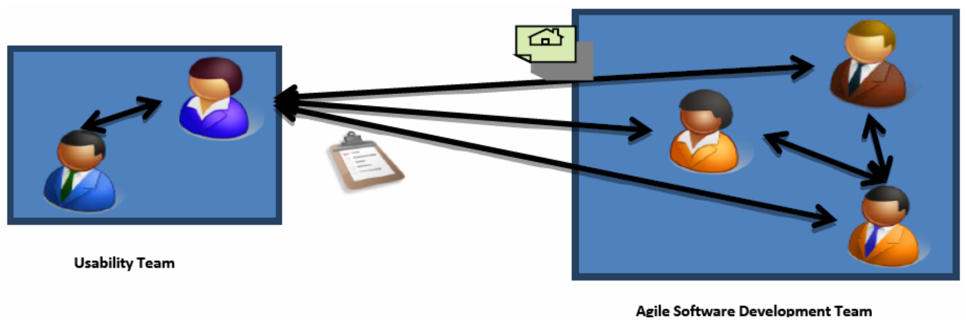
Singh argues that having two product owners in Scrum – called U-Scrum, where one focuses on agile; the other focuses on usability – can improve product usability (Singh, 2008). She found improvements in developer productivity with this structure, noting five factors that are critical to prevent potential obstacles with this approach: the two product owners should be peers, additional coordination may be required, an argument may be needed to justify the additional role, the development team must view personas as an input to the process versus artificial creations, and the UX vision must provide a complete picture for the project.

McDonald and Welland (2003) argue that creating agile multidisciplinary sub-teams of diverse specialists along with coordination teams to maintain communication among specialists will improve the inherent need for diversity in agile web engineering. They identify several stakeholder roles required for large web engineering projects. Namely, these are end-users, clients, domain experts, business experts, software engineers, creative designers, and team leaders. In their approach, sub-teams will include a number of different specialists as well as the infrastructure required for specialists to communicate across teams based on their specialty.

3.6 Social integration

Social integration (Figure 5) involves a remedy where integration occurs via the social construction of knowledge or changing how the team interacts socially. Ungar (2008) and Ungar and White (2008) provide an example of social integration through presentation of a design studio. He argues that holding a design studio (i.e., a workshop) with designers and developers is a viable approach to moving design ahead of development under the compressed time frame of Scrum. The design studio has four components – conduct user research, rapid design generation, evaluate created designs (i.e., the studio component), and the participants from various disciplines. The author found that the design studio facilitates role sharing and knowledge transfer, rapid exploration, early commitment, shared understanding, team cohesion, and the sharing of best practices.

Figure 5 Social integration strategy (see online version for colours)



Similarly, Brown et al. (2008) have investigated the role that stories, sketches, and lists (e.g., a product backlog) play in mediating the interaction between developers and designers. They argue that stories and sketches, as mediating artefacts, have critical roles in the collaboration between interaction (Ix) designers and agile developers. They found that sketches and stories support creation and reflection, facilitate resolution of contradiction, and also work at a level of consciousness that is below the level of self-awareness.

At a higher level of task abstraction, Barksdale and his colleagues (Barksdale and McCrickard, 2010; Barksdale et al., 2009) presented a method for connecting the various domains through the use of concept mapping. They argue that collaborative concept mapping can alleviate politics on agile usability teams and can improve their interaction by facilitating communication while enabling role autonomy. In their approach, usability experts add scenarios to the map, developers link the stories to the scenarios, and both collaborate on that link to provide deeper rationale for the association between the scenarios and stories. They found that – although there is a need for improvement in agile usability and a concept mapping approach is promising for improving team interaction in agile usability environments – role familiarity is important, collaboration and communication do not imply one another, and there is value in providing steps for sharing knowledge versus just the structure to do so.

4 Social interaction integration strategy

This research approach applies social capital, network governance, and usability knowledge management as a lens through which to analyse and understand the role of social interaction in agile usability software teams. This section provides the background necessary to understand the framework and its application in this research.

4.1 Social capital

Social relationships play a key role in how well team members communicate. Healthy relationships can make it easier for team members to ask for help, serve as a support system during difficult times, or generally provide a more fulfilling work experience. A team that dismisses their social dynamics and structure might struggle in accomplishing their tasks, whereas those that leverage these characteristics might function more fluidly through awareness of influential social factors. For example, brainstorming with other team members may help another member resolve some of the issues experienced with a colleague that prevents them from focusing on the project. Talking with other members about the experience may also help establish rapport and build trust within the team.

Social capital provides a means for capturing and understanding these social dynamics. It has been defined in many ways (Adler and Kwon, 2002) across various disciplines, such as political science and public administration, sociology, and business (Table 2). However, there is still a lack of consensus on what it entails and its purpose. For some, social capital is merely a reconstruction of social network theory, and thus unnecessary. For others, it provides a means for providing a more accessible measure of such dynamics and presumes that social networks have value. The most fitting, and selected, definition for this work is that provided by Nahapiet and Goshal (1998).

Social capital is commonly assessed using social network analysis (SNA), which measures and analyses the properties of social networks (Cross et al., 2002). These measures include, for example, centrality (to assess how connected an individual is), cohesion (the degree to which members are directly tied to each other, which can be used to identify cliques), and density (the amount of connectivity among network members). Such data is useful for gaining insight about how valuable the members and the network are, how well social capital is leveraged in networks, and how it can be adjusted for desired outcomes.

Table 2 Definitions of social capital

Authors	*Domain*	*Definition*
Hanifan (1916)	Political science	"… that in life which tends to make these tangible substances count for most in the daily lives of a people, namely, good-will, fellowship, mutual sympathy and social intercourse among a group of individuals and families who make up a social unit, the rural community, whose logical center is the school."
Bourdieu (2008)	Sociology	"The aggregate of the actual or potential resources which are linked to possession of a durable network of more or less institutionalized relationships of mutual acquaintance or recognition."
Schiff (1992)	Business	"… the set of elements of the social structure that affects relations among people and are inputs or arguments of the production and/or utility function."
Burt (1992)	Public administration	"… friends, colleagues, and more general contacts through whom you receive opportunities to use your financial and human capital."
Fukuyama (1997)		"… the existence of a certain set of informal values or norms shared among members of a group that permit cooperation among them."
Nahapiet and Ghoshal (1998)	Business	"… the sum of the actual and potential resources embedded within, available through, and derived from the network of relationships possessed by an individual or social unit."
Putnam (2000)	Public administration	"… connections among individual-social networks and the norms of reciprocity and trustworthiness that arise from them."

4.2 Network governance

Network governance theory emphasises the coordination of informal social systems through establishing or leveraging structures to guide network activities and network-level outcomes (e.g., network efficiency) (Jones and Hesterly, 1997; Provan and Kenis, 2008). This means changing how the network functions to achieve a desired result. It is different from the structural dimension of social capital because it is concerned with the structure of decision-making in the network instead of merely the positioning of members in the network.

The network governance theory used in this study was developed by Provan and Kenis (2008). They define network governance as involving "the use of institutions and structures of authority and collaboration to allocate resources and to coordinate and control joint action across the network as a whole". This theory is used because it not only distinguishes between organisational and network governance (with a focus on the network), but also because it is flexible – able to accommodate application in a variety of domains. Consistent with the intentions of this work, they view the network "as a variable, examining different network governance configurations and the conditions for the effectiveness of each form".

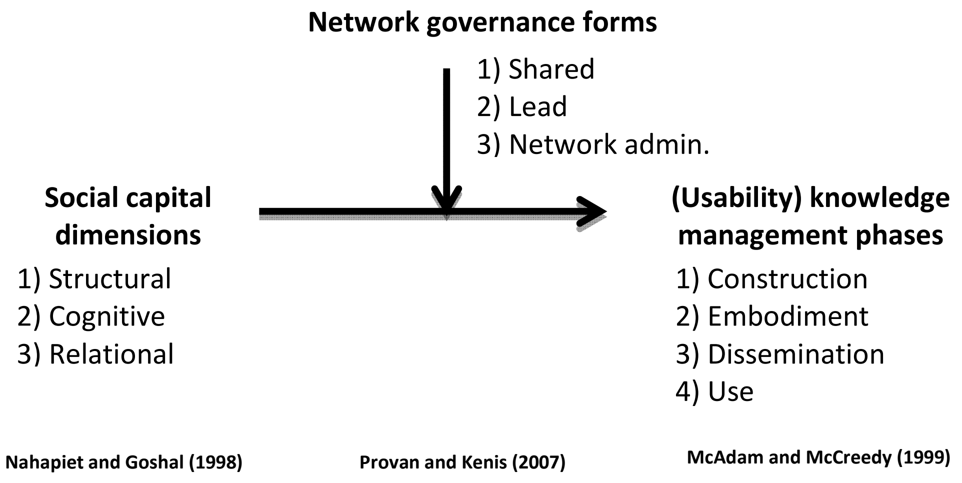
4.3 Knowledge management

The way knowledge is created and shared has implications for and consequences in work team collaboration. What, how, and when knowledge is acquired and exchanged can influence the team's effectiveness and efficiency. If team members do not have a clear understanding of the knowledge they are sharing or attach different meanings to knowledge that is gained, then collaboration can become strained and ineffective. However, if knowledge can be managed in a way that enables a shared understanding and is exchanged at opportune moments, then it could put team collaboration on a more solid footing, possibly increasing the likelihood of the team realising their objectives.

5 Agile usability team interaction framework

The goal of the theoretical framework (Figure 6) is to provide researchers and practitioners with an understanding of how social interaction influences the integration of the agile software domain and the usability domain (i.e., analytical purpose). The framework also aids with providing guidance to practitioners on the best practices for achieving effective usability knowledge management in their agile usability teams (i.e., practical purpose). Hence, the understanding gained from the analytical purpose informs the practical purpose.

Figure 6 Theoretical framework



5.1 Social capital and the framework

Social capital is incorporated into the framework as a partial explanation of what influences how usability knowledge is managed. Its inclusion has import for both the analytical and practical purposes. From an analytical perspective, its inclusion in the framework facilitates capturing data about the team's social capital. Its inclusion also helps to answer the question of what and how best to measure social capital. From a practical perspective, it helps to give practitioners insight about what targeted changes they can make to their social capital to create change that is likely to achieve their desired effectiveness for how usability knowledge is managed.

5.2 Network governance and the framework

Network governance serves as another partial explanation of how the relationship between social capital and usability knowledge management is affected by the governance form of the network (or team). In this framework, however, network governance is analysed as having influence on the relationship between social capital and usability knowledge management versus directly on usability knowledge management. It means that some governance forms may influence the relationship between social capital and usability knowledge management and some may not.

5.3 Knowledge management and the framework

Usability knowledge management is the outcome of interest in the framework and of this study. How usability knowledge is managed has implications for the resulting usability of the software product, team management, and customer satisfaction. For example, ineffective or inefficient management of usability knowledge could result in usability decisions not being incorporated into the software because of a breakdown anywhere between the time it was created and its potential use. Although all usability knowledge may not be utilised, there is value in knowing where and why it lost traction and whether it was an intentional or unintentional decision to not use that knowledge.

6 Framework evaluation

To assess the potential effectiveness and applicability of the proposed agile usability team interaction framework to professional practices, we conducted an expert review. An expert review is a structured walkthrough of typical tasks meant to test a process or tool (Shneiderman, 2009). It is a common method in UE and other people-centric sciences, providing rapid and low-cost feedback about a method prior to long-term large-scale deployment (which will be our next steps with our framework).

Specifically, our expert review asked four domain experts to provide feedback on the perceived benefits, limitations, application, and challenges with applying the framework based on their areas of expertise. They were provided with materials related to our framework, and they were asked to respond to a series of questions about it. The evaluation covered four areas of inquiry: the perceived benefits, application, limitations, and application challenges of the framework. Follow-up questions were used to resolve discrepancies.

6.1 Method

Data were collected via an online survey from four professionals with expertise in software engineering, ASD, usability, behavioural analysis, social capital and network analysis, and UX. The average work experience across experts was 13 years. Each participant was provided with an overview and description of the framework for reference.

6.2 Results

Participants reported such benefits as the framework's potential to better integrate team members of different backgrounds and improve the productivity and performance of the team.

In terms of application of the framework, the experts stated that using scenarios as guidance would be useful in applying the framework, and that the framework could be applied in teams where there is uncertainty in the team or innovative leadership willing to take risks. It was also communicated that techniques such as workshops, tutorials, consulting, and discussions were effective methods for applying the framework.

Participants reported the following limitations of the framework: the possibility of the team experiencing gridlock if they were highly polarised, that the framework should be more accessible to practitioners given its currently theoretical nature, that it should also appeal to agile developers, that it should provide actionable guidance for the practitioner, that there might exist conflict between the goals of the individual team members and the team, and that the framework should be sensitive enough to recognise influences that are not explicitly part of the model.

The limitations noted are consistent with the vision for the framework. The current implementation of the framework is intended to help us understand existing interactions in a multidisciplinary agile usability team. The insight gained from studying teams with the existing model will help us identify additional factors that might influence how usability knowledge is managed and inform the crafting of practical guidance on how team members can improve their interaction and minimise polarisation. Additionally, usability knowledge management is the outcome for the current model, but it is our goal to gradually expand the area of concern to other roles toward improving knowledge management throughout the team generally.

Key challenges to applying the framework mentioned by several reviewers were obtaining buy-in by management and team members, and motivating practitioners to utilise the framework over existing approaches. Also mentioned were challenges with ensuring effective communication and coordination in the team, and balancing short-term and long-term strategic interests.

Adoption is a common concern when introducing tools or approaches, and we acknowledge these potential challenges. Our aim is to increase the likelihood of adoption by building the framework on an empirical foundation. By studying teams using the existing model, we anticipate the model will ultimately better reflect the actual concerns of multidisciplinary agile usability teams. Informing the framework with empirical data from agile usability teams will also help establish buy-in from practitioners. We also anticipate that keeping agile principles in mind during the development of the framework will help toward mitigating application challenges.

7 Discussion

Reflecting on the existing agile usability integration strategies, through the lens of the proposed framework, helps clarify the importance of social interaction in agile usability teams. This section discusses the role social interaction plays in existing agile usability integration strategies – thus comparing each strategy with social capital, network governance, and usability knowledge management. The purpose of this critical analysis is to highlight the relevance of social interaction in existing integration strategies.

7.1 Practices integration via framework lens

Practices integration was defined earlier as the sharing of practices employed by each domain: ASD and usability. The team's social interaction can influence how effective the team is at sharing practices. For example, the team structure, the level of shared meaning and learning within the team, relationships among members across domains, and the level of inclusion when it comes to deciding how the team will function are likely to impact to what extent practices sharing occurs.

Team structure attributes, such as the density of communication links within the team, can impact whether sufficient communication occurs among members across domains. If communication is primarily between a couple of agile software developers and usability, for example, they could miss opportunities to share best practices based on their unique experiences.

Cognitive attributes, such as team learning and shared meaning, can affect how easy it is for members to share best practices. Members of each domain have a language that is unique to their domain, and overlap between domains may or may not exist. In instances where there is not an overlap, communicating a best practice requires translation on both parts to clearly describe, understand, and apply the practice.

Relational attributes (e.g., trust) can influence whether members of a domain trust sharing practices with each other. For instance, if the usability experts believe that sharing practices with agile software developers will be used against them in some way, they may be less inclined to share that knowledge.

Network governance, concerned with the decisions of resource allocation in and coordination of network members, can affect whether shared practices are implemented. In a shared governance form, members have greater control over whether practices are employed within the team, whereas a team lead might make that decision in a lead governance form.

The effects of practices integration on usability knowledge management without consideration of the team's social interaction could yield differing results. Examining practices integration through each framework component shows that sharing does not occur irrespective of the team's style of social interaction.

7.2 Process integration via framework lens

Process integration – the integration of disparate ASD and usability processes – more strongly considers social interaction than practices integration, but still has its limitations. Its primary limitation is the explicit and routinised approach to social interaction. While this may be ideal for process-driven tasks (e.g., writing code) and still fundamental to teams, a more personable compliment is needed when interaction with others is necessary.

The team's structure can facilitate or undermine the integrated process used by the agile software developers and usability experts. If the process prescribes that the developers and usability experts communicate with each other at regular intervals, but they are isolated from each other in their structure, this could undermine that objective of the integrated process.

Although an integrated process aids in providing a shared language within the team via the process language, it cannot take the place of shared meaning across domains. Team members of both domains may understand and follow the integrated process, but the possibility for misunderstandings about the product remains. In essence, shared process language does not imply shared domain language.

Relational attributes are commonly overlooked in an integrated process. Interaction prescribed by the process is commonly task-focused and less concerned with whether following it builds such things as trust. Also, a low level of trust in the team could adversely impact the effectiveness of the process. For example, members may follow the process but withhold useful information, or they may simply selectively use the process.

A benefit of the integrated process strategy is that it facilitates coordination among members, which complements the network governance forms. What network governance contributes (besides the process) is knowledge of how the social interaction impacts the various structures for making team resource allocation and coordination decisions.

Although it is likely that usability knowledge management is less of a concern with the process integration strategy, awareness of the social interaction can ease its management. Knowing the strengths and weaknesses of the team's social interaction style enables the team to adjust their interaction to get the most out of the process, and thereby minimise barriers to effective management of usability knowledge.

7.3 Technology integration via framework lens

Technology integration leverages the communication between the tools used by each domain. An emphasis on social interaction is important in the context of technology integration because it is easier for members to work through their tools as their primary method of interacting. The limitation is more obvious in this strategy than the practices and process integration strategies. Namely, that limitation is the risk of social interaction being practically non-existent.

Without consideration of the structural dimension in the technology integration strategy, some critical communication may not happen. Instead, members might believe that what is put into the tools is all that matters. For example, the usability expert might update the UI design in the tool, which updates in the developer's tool. However, there are other topics (e.g., design goals) that have to be discussed.

Establishing shared meaning and understanding is challenging when person to person interaction exists. When technology is the primary medium, some of the personal context can be lost if social interaction is not built into the way tools are used.

Building trust and establishing relationships within teams can be challenging. To some degree, technology can facilitate building relationships given the ease with which members can communicate with each other. However, such computer mediated communication can result in a loss of information that can adversely affect the relationship between team members. By focusing on social interaction, teams can be more strategic in what technology they use and how they use it.

Usability knowledge management can benefit from the use of technology when the appropriate technology is used to institutionalise the knowledge. However, the quality of the knowledge entered into and used from knowledge management systems can be affected by the structural, cognitive, relational, and governance nature of the team.

7.4 People integration via framework lens

People integration is the addition of team members (or the skillset through training of existing members) to fulfil the requirement for a usability member. The primary limitation with this strategy is that little, if any, attention is given to how the change in the team will influence the social interaction among team members.

The impact of adding a person (or skillset) to the team without considering the team's structure and communication flow could create political tension among the team. For example, the team might have a very different dynamic if the usability expert is responsible to a developer versus the manager over developers. The former might result in a greater emphasis on developer activity than usability activities.

When finding shared meaning and understanding, it is easy to dismiss the need to emphasise establishing common ground in the people integration strategy. For example, a usability expert might be added to the team, resulting in an assumption that the addition is all that is necessary. To the contrary, effort should also go into ensuring the team and usability expert are willing to learning from each other for meaningful output to occur.

On the relational dimension, if the usability expert does not make a conscious effort to gain the trust of the other members of the team or learn the norms, it can make their collaboration more difficult. Conversely, developers must gain the trust of the usability expert as a means to ensuring that usability knowledge is incorporated into the product.

It is possible that usability knowledge management could be more difficult at the team level if the social interaction between the usability expert and the rest of the team is not sufficient. For example, usability knowledge creation might be difficult without the necessary access to the users.

8 Implications and recommendations

8.1 Practical implications and recommendations

The field of software development has been perceived as being comprised of those who prefer working at a computer to socialising with others. For practitioners, the proposed relationship between social interaction and usability knowledge management could mean that team social dynamics have a greater impact on the software product than realised, which could mean a need for a more social culture among development teams.

As agile usability software teams are formed, the proposed relationships suggest that social interaction should be factored into team-related decisions. For example, it might prove useful for teams to more strategically consider the structural, cognitive, relational characteristics of their team, as well as the governance form used in their team. This might especially prove advantageous in teams where usability experts and software developers are not on separate teams and are disproportionate in size. For example, if there are only one or two usability experts on a larger agile development team, the team interaction framework can help balance the influence via the governance form used, or by identifying and strengthening a particular dimension of social capital (e.g., the cognitive dimension toward greater shared meaning) among the team members.

As usability continues to gain traction in ASD teams, interaction among team members will become ever more important. This will especially hold in cases such as mobile development, where UI development can play a key role in whether users keep or discard mobile applications. This team interaction framework is well suited to help teams mature their interactions and adjust to the platform for which they're developing.

Another practical implication is that teams may experience difficulty with encouraging some team members to move out of their existing comfort zone. Team leads and other team level managers can possibly mitigate such concerns by minimising the tension that can develop between what is communicated and what is rewarded. By bringing communication and rewards into alignment for team members, they may be more likely to accept and respond to the need for more social interaction. For example, communication can occur by consistently encouraging developers and usability experts to work directly with one another instead of making assumptions about insight that falls in each other's areas of expertise (e.g., assuming what the user might prefer or technical limitations). In this example, one possible reward is to publicly acknowledge, during team meetings, the benefits that resulted from the interaction between the usability expert and the developer.

8.2 Research implications and recommendations

As shown from the review of related literature, additional study of the social interaction in agile usability software teams is needed. Agile usability research serves as an opportunity for research and practice to contribute to one another toward the production of grounded and empirical knowledge. This knowledge then has greater relevance to practice and is likely to be more consumable by practitioners. Conversely, practice can better inform research as practitioners are likely to provide more relevant feedback from their involvement in the research.

As research on agile usability continues, there will be a need for identifying lines of inquiry that provide understanding about whether, and in what cases, social interaction is most critical in agile usability software teams. This also includes determining which kinds and what levels of social interaction are most beneficial to product usability and innovation at varying phases of the project. For example, is social interaction more important in the beginning, middle, or end of an iteration or a release? Is less social interaction less beneficial if all team members have the understanding needed to complete a task? Is the cognitive dimension more important during the beginning of an iteration and the relational dimension more important at the end of an iteration? Consideration of these and many other related questions can help agile usability teams better understand the role social dynamics play in their team and can better arm them with the understanding required to make adjustments.

8.3 Policy implications and recommendations

In line with encouraging a change in team interaction at the team level, management may need to modify how they measure performance to ensure that they are encouraging effective social interaction in and across teams from the corporate level. Corporate changes could include modifying their core values or how they are communicated, building recognition of cross-team interaction into their corporate incentive program, or consistently communicate and reinforce its importance when during interaction with the team members and during organisational meetings.

Understanding the impact of social interaction on organisational policies is possible via a strategic partnership between researchers and practitioners. Access to organisations is imperative when conducting research that benefits the practice of developing software products. Currently, it can be challenging to gain access to experts and the proprietary data of organisations.

To empirically understand and improve the social interaction of agile usability software teams and its effects through organisational policies, a precursor of policies that guide the collaboration between researchers and practitioners is needed. Such policies might formalise, for example, the extent to which employees can participate in studies, how participant anonymity and data confidentiality will be maintained, and the benefits of the study to the organisation and the community generally.

9 Conclusions

This research endeavoured to contribute a history of the development of agile usability integration, propose a framework for evaluating the social interaction and usability knowledge management in agile usability software teams, and to discuss the implications of and recommendations for improving the social interaction and management of usability knowledge in agile usability software teams.

We identified five categories of agile usability integration – practice, process, technology, people, and social – that have been researched or used. We focused on the social integration approach and critiqued the role of social interaction in the other four integration approaches. More specifically, we discussed the potential impact of not duly considering the role of social interaction – in agile usability and in other integration strategies – on team dynamics and product usability and innovation. Finally, we discussed the implications and recommendations for practice, research, and policy relative to the framework.

Acknowledgements

The authors thank the expert reviewers for their time in providing feedback about the framework. The authors also thank the journal reviewers and editor for their thoughtful comments.

References

Adikari, S., McDonald, C. and Campbell, J. (2009) 'Little design up-front: a design science approach to integrating usability into agile requirements engineering', *Human-Computer Interaction. New Trends*, Vol. 5610, pp.549–558, Springer.

Adler, P. and Kwon, S. (2002) 'Social capital: prospects for a new concept', *Academy of Management Review*, Vol. 27, No. 1, pp.17–40, JSTOR.

Ambler, S.W. (2008) 'Tailoring usability into agile software development projects', *Maturing Usability*, pp.75–95, Springer, London.

Anwar, M.F. (2006) 'Engineering the requirements in user-centered design and agile development methodologies', *CUSEC 2006*, Citeseer, p.56.

Barksdale, J.T. and McCrickard, D.S. (2010) 'Concept mapping in agile usability: a case study', *Proceedings of the 28th of the International Conference Extended Abstracts on Human Factors in Computing Systems*, ACM, pp.4691–4694.

Barksdale, J.T., Ragan, E.D. and McCrickard, D.S. (2009) 'Easing team politics in agile usability: a concept mapping approach', *Agile Conference, 2009, AGILE'09*, IEEE, pp.19–25.

Beck, K. and Andres, C. (2004) *Extreme Programming Explained: Embrace Change*, 2nd ed., Addison-Wesley Professional, USA.

Benigni, G., Gervasi, O., Passeri, F. and Kim, T.H. (2010) 'USABAGILE_Web: a web agile usability approach for web site design', *Computational Science and Its Applications – ICCSA 2010*, Springer, pp.422–431.

Blomkvist, S. (2005) 'Towards a model for bridging agile development and user-centered design', *Human-Centered Software Engineering – Integrating Usability in the Software Development Lifecycle*, Springer, pp.219–244.

Booch, G., Rumbaugh, J. and Jacobson, I. (2005) *Unified Modeling Language User Guide*, The (Addison-Wesley Object Technology Series), Addison-Wesley Professional, USA.

Bourdieu, P. (2008) 'The forms of capital', in N.W. Biggart (Ed.): *Readings in Economic Sociology*, Blackwell Publishers Ltd., Oxford.

Brown, J., Lindgaard, G. and Biddle, R. (2008) 'Stories, sketches, and lists: developers and interaction designers interacting through artefacts', *Agile 2008 Conference*, IEEE, pp.39–50.

Budwig, M., Jeong, S. and Kelkar, K. (2009) 'When user experience met agile: a case study', *Proceedings of the 27th International Conference Extended Abstracts on Human Factors in Computing Systems*, pp.3075–3084, ACM.

Burt, R.S. (1995) *Structural Holes: The Social Structure of Competition*, First Harvard Univ. Press, USA.

Bygstad, B., Ghinea, G. and Brevik, E. (2008) 'Software development methods and usability: perspectives from a survey in the software industry in Norway', *Interacting with Computers*, Vol. 20, No. 3, pp.375–385, Elsevier.

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

Chamberlain, S., Sharp, H. and Maiden, N. (2006) 'Towards a framework for integrating agile development and user-centred design', *Extreme Programming and Agile Processes in Software Engineering*, Vol. 4044, pp.143–153, Springer.

Clarke, J., Connors, J. and Bruno, E.J. (2009) *JavaFX: Developing Rich Internet Applications*, Prentice Hall PTR, USA.

Constantine, L. (2002) 'Process agility and software usability: toward lightweight usage-centered design', *Information Age*, Vol. 8, No. 8, pp.1–10.

Constantine, L.L. and Lockwood, L.A.D. (2002) 'Usage-centered engineering for web applications', *IEEE Software*, Vol. 19, No. 2, pp.42–50, IEEE Computer Society.

Constantine, L.L. and Lockwood, L.A.D. (2003) 'Usage-centered software engineering: an agile approach to integrating users, user interfaces, and usability into software engineering practice', *Proceedings of the 25th International Conference on Software Engineering*, IEEE Computer Society, pp.746–747.

Cross, R., Borgatti, S. and Parker, A. (2002) 'Making invisible work visible: using social network analysis to support strategic collaboration', *California Management Review*, Vol. 44, No. 2, pp.25–47, University of California.

Detweiler, M. (2007) 'Managing UCD within agile projects', *Interactions*, Vol. 14, No. 3, pp.40–42, ACM.

Duchting, M., Zimmermann, D. and Nebe, K. (2007) 'Incorporating user centered requirement engineering into agile software development', *Proc. HCII*, pp.58–67, Springer, Berlin, Heidelberg.

Evnin, J. and Pries, M. (2008) 'Are you sure? Really? A contextual approach to agile user research', *Agile 2008 Conference*, IEEE, pp.537–542.

Federoff, M., Villamor, C., Miller, L., Patton, J., Rosenstein, A., Baxter, K. and Kelkar, K. (2008) 'Extreme usability: adapting research approaches for agile development', *Proc. CHI Extended Abstracts*, pp.2269–2272, ACM.

Ferre, X., Juristo, N. and Moreno, A. (2005a) 'Framework for integrating usability practices into the software process', *Product Focused Software Process Improvement*, Vol. 3547, pp.202–215, Springer.

Ferre, X., Juristo, N. and Moreno, A. (2005b) 'Which, when and how usability techniques and activities should be integrated', *Human-Centered Software Engineering – Integrating Usability in the Software Development Lifecycle*, pp.173–200, Springer, Netherlands.

Ferreira, J., Noble, J. and Biddle, R. (2007a) 'Up-front interaction design in agile development', *Agile Processes in Software Engineering and Extreme Programming*, Vol. 4536, pp.9–16, Springer.

Ferreira, J., Noble, J. and Biddle, R. (2007b) 'Agile development iterations and UI design', *AGILE 2007*, IEEE, pp.50–58.

Ferreira, J., Sharp, H. and Robinson, H. (2010) 'Values and assumptions shaping agile development and user experience design in practice', *Agile Processes in Software Engineering and Extreme Programming*, Springer, pp.178–183.

Fowler, M. and Highsmith, J. (2001) 'The agile manifesto', *Software Development*, Vol. 9, No. 8, pp.28–35, Miller Freeman, Inc., 1993, San Francisco, CA.

Fox, D., Sillito, J. and Maurer, F. (2008) 'Agile methods and user-centered design: how these two methodologies are being successfully integrated in industry', *Agile, 2008, AGILE'08*, IEEE, pp.63–72.

Fukuyama, F. (1997) 'Social capital and the modern capitalist economy: creating a high trust workplace', *Stern Business Magazine*, Vol. 4, No. 1, pp.1–16.

Ghosh, G. (2004) 'Agile, multidisciplinary teamwork', *Methods & Tools*, available at http://www.methodsandtools.com/archive/archive.php?id=17 (accessed on 25 July 2011).

Göransson, B., Gulliksen, J. and Boivie, I. (2003) 'The usability design process – integrating user-centered systems design in the software development process', *Software Process: Improvement and Practice*, Vol. 8, No. 2, pp.111–131.

Haikara, J. (2007) 'Usability in agile software development: extending the interaction design process with personas approach', *Agile Processes in Software Engineering and Extreme Programming*, Vol. 4536, pp.153–156, Springer.

Hanifan, L.J. (1916) 'The rural school community center', *Annals of the American Academy of Political and Social Science*, Vol. 67, No. 1, pp.130–138, JSTOR.

Hansson, C. (2002) *User Driven Software Development in a Small Company*, Blekinge Institute of Technology.

Highsmith, J. and Cockburn, A. (2001) 'Agile software development: the business of innovation', *Computer*, Vol. 34, No. 9, pp.120–127, IEEE.

Hudson, W. (2005) 'A tale of two tutorials: a cognitive approach to interactive system design and interaction design meets agility', *Interactions*, Vol. 12, No. 1, pp.49–51, ACM.

Hussain, Z. and Slany, W. (2009a) 'Current state of agile user-centered design: a survey', *HCI and Usability for e-Inclusion*, Vol. 5889, pp.416–427.

Hussain, Z and Slany, W. (2009b) 'Investigating agile user-centered design in practice: a grounded theory perspective', *HCI and Usability for e-Inclusion*, Vol. 5889, pp.279–289, Springer.

Hussain, Z., Lechner, M. and Milchrahm, H. (2008) 'Agile user-centered design applied to a mobile multimedia streaming application', *HCI and Usability for Education and Work*, Vol. 5298, pp.313–330.

Hussain, Z., Milchrahm, H., Shahzad, S. and Slany, W. (2009) 'Integration of extreme programming and user-centered design: lessons learned', *Extreme Programming*, Vol. 31, No. 3, pp.174–179.

Jones, C. and Hesterly, W. (1997) 'A general theory of network governance: exchange conditions and social mechanisms', *Academy of Management Review*, Vol. 22, No. 4, pp.911–945.

Kruchten, P. (1995) 'The 4+1 View Model of Architecture', *Software*, Vol. 12, No. 6, pp.42–50, IEEE.

Larman, C. (2002) *Applying UML and Patterns : An Introduction to Object-Oriented Analysis and Design and the Unified Process*, 2nd ed., Prentice Hall PTR, Upper Saddle River, NJ.

Law, E., Roto, V., Vermeeren, A.P.O.S., Kort, J. and Hassenzahl, M. (2008) 'Towards a shared definition of user experience', *Proc. CHI Extended Abstracts*, pp.2395–2398, ACM, Florence, Italy.

Lee, J. and McCrickard, D. (2007) 'Towards extreme(ly) usable software: exploring tensions between usability and agile software development', *Agile 2007*, pp.59–71, IEEE Computer Society, Washington, DC.

Lee, J.C. (2006) 'Embracing agile development of usable software systems', *Proc. CHI Extended Abstracts*, pp.1767–1770, ACM.

Lee, J.C. (2010) 'Integrating scenario-based usability engineering and agile software development', PhD dissertation, Virginia Tech, Blacksburg, VA.

Lee, J.C., McCrickard, D.S. and Stevens, K.T. (2009) 'Examining the foundations of agile usability with eXtreme scenario-based design', *2009 Agile Conference*, IEEE, pp.3–10.

Lee, J.C., Wahid, S., McCrickard, D.S., Chewar, C. and Congleton, B. (2007) 'Understanding usability: investigating an integrated design environment and management system', *Interactive Technology and Smart Education*, Vol. 4, No. 3, pp.161–175, Emerald Group Publishing Limited.

Lievesley, M.A. and Yee, J.S.R. (2006) 'The role of the interaction designer in an agile software development process', *Proc. CHI Extended Abstracts*, pp.1025–1030, ACM.

Löwgren, J. and Stolterman, E. (2004) *Thoughtful Interaction Design: A Design Perspective on Information Technology*, The MIT Press, USA.

MacVittie, L.A. (2006) *XAML in a Nutshell*, O'Reilly Media, Inc., Sebastopol, CA.

Mayhew, D. (1999) *The Usability Engineering Lifecycle: A Practitioner's Handbook for User Interface Design*, Morgan Kaufmann, San Francisco, CA.

McAdam, R. and McCreedy, S. (1999) 'A critical review of knowledge management models', *The Learning Organization*, Vol. 6, No. 3, pp.91–101.

McClelland, I. (2005) ''User experience' design a new form of design practice takes shape', *Proc. CHI Extended Abstracts*, pp.1096–1097, ACM, Portland, OR, USA.

McDonald, A. and Welland, R. (2003) 'Agile web engineering (AWE) process: multidisciplinary stakeholders and team communication', *Web Engineering*, Springer, pp.253–312.

McInerney, P. and Maurer, F. (2005) 'UCD in agile projects: dream team or odd couple?', *Interactions*, Vol. 12, No. 6, pp.19–23, ACM.

Memmel, T., Bock, C. and Reiterer, H. (2008) 'Model-driven prototyping for corporate software specification', *Engineering Interactive Systems*, Springer, pp.158–174.

Memmel, T., Gundelsweiler, F. and Reiterer, H. (2007a) 'Agile human-centered software engineering', *Proceedings of the 21st British HCI Group Annual Conference on People and Computers: HCI... But Not as We Know It*, British Computer Society, Vol. 1, pp.167–175.

Memmel, T., Gundelsweiler, F. and Reiterer, H. (2007b) 'CRUISER: a cross-discipline user interface and software engineering lifecycle', *Human-Computer Interaction: Interaction Design and Usability*, Springer, pp.174–183.

Memmel, T., Gundelsweiler, F. and Reiterer, H. (2007c) 'Prototyping corporate user interfaces: towards a visual specification of interactive systems', *Proceedings of the Second IASTED International Conference on Human Computer Interaction*, ACTA Press, pp.177–182.

Memmel, T., Reiterer, H. and Holzinger, A. (2007d) 'Agile methods and visual specification in software development: a chance to ensure universal access', *Universal Access in Human Computer Interaction: Coping with Diversity*, Vol. 4554, pp.453–462, Springer.

Meszaros, G. and Aston, J. (2006) 'Adding usability testing to an agile project', *Agile Conference*, IEEE, p.6.

Miller, L. (2006) 'Case study of customer input for a successful product', *Agile Conference, 2005, Proceedings*, IEEE, pp.225–234.

Miller, L. and Sy, D. (2009) 'Agile user experience SIG', *Proceedings of the 27th international Conference Extended Abstracts on Human Factors in Computing Systems*, ACM, pp.2751–2754.

Nahapiet, J. and Ghoshal, S. (1998) 'Social capital, intellectual capital, and the organizational advantage', *Academy of Management Review*, Vol. 23, No. 2, pp.242–266, JSTOR.

Nielsen, J. (1993a) 'Iterative user-interface design', *Computer*, Vol. 26, No. 11, pp.32–41, IEEE.

Nielsen, J. (1993b) *Usability Engineering*, Morgan Kaufmann, San Francisco, CA.

Nunes, N.J. (2009) 'What drives software development: bridging the gap between software and usability engineering', *Human-Centered Software Engineering*, Springer, pp.9–25.

Obendorf, H. and Finck, M. (2008) 'Scenario-based usability engineering techniques in agile development processes', *Proc. CHI Extended Abstracts*, pp.2159–2166, ACM.

Paelke, V. and Nebe, K. (2008) 'Integrating agile methods for mixed reality design space exploration', *Proceedings of the 7th ACM Conference on Designing Interactive Systems*, ACM, pp.240–249.

Paelke, V. and Sester, M. (2010) 'Augmented paper maps: exploring the design space of a mixed reality system', *ISPRS Journal of Photogrammetry and Remote Sensing*, Vol. 65, No. 3, pp.256–265, Elsevier.

Parsons, D., Lal, R., Ryu, H. and Lange, M. (2007) 'Software development methodologies, agile development and usability engineering', *ACIS 2007 Proceedings*, Citeseer, p.21.

Patton, J. (2002a) 'Hitting the target: adding interaction design to agile software development', *OOPSLA 2002 Practitioners Reports*, ACM, pp.1–7.

Patton, J. (2002b) 'Designing requirements: incorporating usage-centered design into an agile SW development process', *Extreme Programming and Agile Methods – XP/Agile Universe 2002*, pp.95–102, Springer.

Patton, J. (2007) 'Understanding user centricity', *IEEE Software*, November, pp.9–11.

Provan, K.G. and Kenis, P. (2008) 'Modes of network governance: structure, management, and effectiveness', *Journal of Public Administration Research and Theory*, Vol. 18, No. 2, pp.229–252.

Putnam, R. (2000) *Bowling Alone: The Collapse and Revival of American Community*, Simon & Schuster, New York, NY.

Pyla, P. (2007) 'Connecting the usability and software engineering life cycles through a communication-fostering software development framework and cross-pollinated computer science courses', PhD dissertation, Virginia Tech, Blacksburg, VA.

Rosson, M. and Carroll, J. (2002) *Usability Engineering: Scenario-Based Development of Human-Computer Interaction*, Morgan Kaufmann, San Francisco, CA.

Schiff, M. (1992) 'Social capital, labor mobility, and welfare', *Rationality and Society*, Vol. 4, No. 2, p.157, Sage Publications.

Schwaber, K. (2004) *Agile Project Management with Scrum (Microsoft Professional)*, 1st ed., Microsoft Press, Redmond, WA.

Sharp, H., Biddle, R., Gray, P., Miller, L. and Patton, J. (2006) 'Agile development: opportunity or fad?', *Proc. CHI Extended Abstracts*, pp.32–35, ACM.

Shneiderman, B. (2009) *Designing the User Interface: Strategies for Effective Human-computer Interaction*, 5th ed., Harlow: Pearson Education, Upper Saddle River, NJ.

Singh, M. (2008) 'U-SCRUM: an agile methodology for promoting usability', *Agile, 2008, AGILE'08, Conference*, IEEE, pp.555–560.

Sohaib, O. and Khan, K. (2010) 'Integrating usability engineering and agile software development: a literature review', *2010 International Conference on Computer Design and Applications (ICCDA)*, IEEE, Vol. 2, pp.32–38.

Sy, D. (2007) 'Adapting usability investigations for agile user-centered design', *Journal of Usability Studies*, Vol. 2, No. 3, pp.112–132.

Sy, D. and Miller, L. (2008) 'Optimizing agile user-centred design', *Proc. CHI Extended Abstracts*, pp.3897–3900, ACM Press, New York, New York, USA, doi:10.1145/1358628.1358951.

Tai, G. (2005) 'A communication architecture from rapid prototyping', *Proc. Workshop on Human and Social Factors of Software Engineering*, Vol. 30, No. 4, pp.1–3.

Ungar, J. (2008) 'The design studio: interface design for agile teams', *Agile 2008*, pp.519–524, IEEE.

Ungar, J.M. and White, J.A. (2008) 'Agile user centered design: enter the design studio – a case study', *Proc. CHI Extended Abstracts*, pp.2167–2178, ACM.

Wolkerstorfer, P., Tscheligi, M., Sefelin, R., Milchrahm, H., Hussain, Z., Lechner, M. and Shahzad, S. (2008) 'Probing an agile usability process', *Proc. CHI Extended Abstracts*, pp.2151–2158, ACM.