ASEE AMERICAN SOCIETY FOR ENGINEERING EDUCATION

Paper ID #35756

Integrating Usability into the Agile Software Development Life Cycle Using User Experience Practices

Ms. Tori Nichole Gardner, Department of Engineering at St. Mary's University

Miss Gardner currently attends St. Mary's University in pursuit of her MS in software engineering. She graduated with her BS in software engineering in May of 2021 from the same university. Her research interests include human-computer interaction, user experience, and user interface design as well as accessibility.

Dr. Ozgur Aktunc, St. Mary's University

Dr. Ozgur Aktunc is a Professor of Software Engineering and Graduate Program Director at St. Mary's Engineering Department. He joined St. Mary's University in August 2009. Dr. Aktunc received his Ph.D. in Computer Engineering from the University of Alabama at Birmingham (UAB) in 2007. He also received his M.S.E.E. from UAB in 2002. He came to United States from Istanbul, Turkey where he received his B.S. degree in Electrical Engineering from Istanbul Technical University. Dr. Aktunc's main research area is Software Engineering with an emphasis on component-based software development. In particular his research relates to software analysis, testing of web applications, software metrics, and Software Engineering education. He also performs outreach activities to improve computing education in San Antonio through summer programs for middle and high school students.

Session 2022

Integrating Usability into the Agile Software Development Life Cycle Using User Experience Practices

Tori N. Gardner and Ozgur Aktunc
Department of Engineering
St. Mary's University
One Camino Santa Maria, San Antonio, TX, 78228

Abstract

Software products have become so influential that they occupy the majority of our time and aid people in accomplishing an overwhelming percentage of daily tasks. Because of this trend, usability has become more important than ever, and like other non-functional qualities, needs to be integrated at every stage of the software development life cycle (SDLC) to produce the most usable products. Even so, current development practices generally introduce usability in later stages of development, causing a gap in usability integration that largely ignores the earlier phases of development. The emergence of Agile methodologies, while generally successful in improving the quality of products, may have contributed to this integration gap as developers are opting to wait for user testing since it occurs iteratively. In actuality, the Agile philosophy supports the early integration of usability in the form of processes and the artifacts that they can produce. In this paper, we explore human-computer interaction (HCI) processes and artifacts and propose a development framework that champions early usability integration in an Agile environment by analyzing their effect on the Scrum and Kanban development frameworks. To gain insight regarding the success of the framework, senior design project groups at an engineering program were interviewed on their inclusion of the components of the proposed framework during the requirements gathering, analysis, and design phases of development. The components of the framework are assessed on their adoptability, how well they were accepted by the developers, and their perceived contribution to the usability of the final product.

Introduction

The impact of software products on our society is undeniable as we are in an era centered around various devices that guide us throughout our day. For all our efforts to make software products attractive to customers and release products on time, it seems developers are missing the human element of software. Co-founder of the Nielsen Norman Group, Don Norman, comments on the usability of modern software calling it "confusing" pointing out that three essential principles of design— discoverability, feedback, and correction— have been lost[1]. This study is purposed with first making a case for including Human Computer Interaction (HCI) practices in software development activities and identifying methods of integrating User Experience (UX) into early software development with the intent of improving the quality of software products by increasing usability.

For this investigation, it was important to identify a target method of development to evaluate the UX practices and in which a proposed framework for usability integration will be adopted. Agile, the development methodology, is largely the champion of industry today with a reported 84% adoption in April of 2021 according to the 15th Annual State of Agile survey conducted by Digital.ai[2]. However, since Agile is regarded as a set of general principles and can be applied to fields outside of software development, we decided to study the inclusion of usability practices in specific Agile methodologies. We narrowed the methodologies to Scrum and Kanban, which are the two of the most popular agile frameworks in industry at 66% and 6% adoption rates respectively[2].

To build the framework, relevant UX practices and the artifacts they produce were investigated by a literature review to list what experts suggest adding to development flow. A useful UX practice can be defined by evaluating how it fits into the Agile frameworks, Scrum and Kanban, and how it fits into the earlier development phases. Since software has become so diverse and exists in every industry imaginable today, there is no single best fit approach for integrating usability and it will vary with an organization's development approach and the product under development. The proposed framework can be used in a flexible way and tailored to fit the project at hand. Through an extensive literature review, we selected specific activities to include in the framework that fall into three greater categories of approaches: participatory design, ethnographic study, and a more recent approach that modifies the typical Agile iterative approach. These methods were selected to ensure the framework contained a well-rounded and diverse set of options to accommodate a multitude of product types for which some approaches may be more appropriate than others. As such, the more specific activities are examples used to explain the benefit of such an approach and can be swapped for a different activity that seeks the same end goal.

UX Practices and their Roles in Development

A well-defined user profile must be established to support successful user-centric development. All aspects of the user profile, understanding who the user is, what they do, and where they do it, can be achieved through a set of practices known as participatory design. These practices are activities that involve a diverse set of non-designers, including end users, in various co-design activities throughout the analysis and design processes[3]. Because of the diversity of the activities in this group, there are different possible actor interactions, purposes, and contexts for each practice that make some better fit for specific developmental needs. Participatory design can be physically creating a design artifact such as a collage or visual map, verbally discussing and brainstorming ideas, or showing/enacting challenges and tasks to the development team. This entails the user's active engagement and sharing of knowledge and ideas to the technical development team whose role it is to guide them in the design process as they guide the developers with their perspective and domain knowledge. This is not to say that the users are taking the place of the designer, but they are instead working alongside the designers to help communicate the whole problem, challenge assumptions and ideas, and offer insights the designer would be unaware of so that the designer can produce a more educated solution[4]. The following design practices are examples of participatory design and involving the user in these activities can help to create an accurate profile of the user.

Joint Application Development

Joint Application Development (JAD) is a method of requirements elicitation that provides dedicated time for the development team and designers to meet and get to know the user representatives and other stakeholders in a conversational setting. The session should include the discussion of the current system, the challenges of using the system, and the impact it has on the organization that is of concern. Business processes should be defined including the user's current roles and tasks as well as points of pain that they experience. Most transparently, a system's functional requirements including the distinction between needs and wants can be accurately collected from a JAD session. As the users are directly and actively involved with the development team, specific characteristics that are valuable for improved usability can also be attained such as user technology literacy, common goals throughout user groups, and the future product's frequency of use. This allows for the cultivation of more tailored UX design decisions that are not made *for* the user, but rather *with* the user. Stakeholders are also more likely to feel a sense of support and ownership of a system if they play a major role in its design[5]. A less apparent benefit of the JAD session is that it provides the development team with a greater understanding of the hierarchical structure of the organization they are working with. Understanding the organizational structure can reveal candidates for the userbase and the different features that are necessary for and valuable to each. Additionally, JAD sessions can reveal candidates for other design approaches such as design thinking activities and ethnographic studies[4].

Contextual Task Analysis

Investigating what the user does, a vital component of the user profile, can be more accurately achieved through ethnographic studies. Sometimes called the Contextual Inquiry, the contextual task analysis is considered "essential" to UX methodologies[7]. It is the real-time observation and "cognitive decomposition" of a real user's tasks that is intended to collect more accurate data regarding the user experience (UX) of a system to inspire improvements and features for a new system. The result is an in-depth understanding the user's work processes and behaviors that can drive design solutions by means of identifying use cases and preserving natural flow of work without preserving the inefficiencies. A contextual task analysis, such as a cognitive walkthrough, includes both non-developers and developers working closely together in an observational style setting. The developer can understand what the user thinks and feels while they move through their workflow and why they might feel this way[4]. Additionally, ethnographic studies reveal the smallest details of a workflow that may be overlooked by a person who performs their tasks like second nature during a traditional interview or JAD session. This provides essential information to improvements of a system as the "tips and tricks" elicited from the users can become built-in functions in a new system to benefit their work style.

Design Sprints

The Design Sprint, a more recent technique in Agile was developed at Google Ventures[8]. It is a process meant to validate what ideas are accepted and rejected through design prototyping and performing a weekly usability test with end users[9]. The Design Sprint allocates designated time for the development team to reconcile larger business questions and vision ideas directly with the customers before beginning to build the product. The sprint is a five-day process in which each day covers a different goal: understanding, ideation, decision-making, prototyping, and testing. To run a design sprint, a cross-functional team including members from the user-base, marketing, finance, design, and development, as well as team leads should be involved to ensure many perspectives are accounted for. Overall, the design sprint is an effective way to learn fast and fail early which can help reduce inefficiencies, miscommunications, and costly mistakes.

Design Thinking

This set of activities visually represents a user's needs to aid designers in understanding their users and identify critical information. They focus on the end-user rather than the technical aspects of a product[6]. By empowering the user with hands-on creative exercises, design thinking activities produce design artifacts that can be used by developers. These artifacts, such as mind maps, can be used to facilitate user-centric development by visualizing the results of the design thinking activity[10]. Sometimes referred to as its umbrella term, cognitive mapping, this idea organizing strategy is a means of documenting a user's thought process; complex and disconnected as it may be in the beginning stages of design, each thought is valuable data that can be used later. Free from common design document restrictions, mind maps represent a business concept in a tree-like structure that shows the hierarchical relationship between topics. The exercise incorporates the guidance of an end user to achieve a more accurate depiction of the business topic[4]. Categorizing can seem simple on the surface and prove more complicated in practice but is also integral to ease of use and discoverability for the user and therefore cannot be overlooked.

Storyboards and scenarios

Storyboards and scenarios are subtly different tools that support one another, though with nuanced purposes. Storyboards, great for visual learners, offer an illustration of a user's journey when using the system to achieve a specific goal. They represent users' plans, actions, and reactions as they move through the steps of a business process and can even be used to show a high-level representation of what a UI might look like at each stage. As it is guided by the user's intent, developers must not include any unnecessary functions, while preserving and taking count of the necessary details and functions. Scenarios, on the other hand are a textual representation of this same journey or probable example of a future user interacting with the product to achieve a goal. They are considered more detailed than storyboards and add another filter to catch missing links in the process and refine the data from which requirements will be taken. Using scenarios and storyboards is an effective way to bridge the gap between high-level elicitations and lower-level requirements by documenting only "what," and "why," but not "how" a system function[4]. They also help later in development by visualizing a project's status. A developer can view a process from start to finish and can see their progress status across the storyboard or scenario[5].

User stories

The Scrum and Kanban frameworks represent requirements as user-stories, a user-centric requirements documentation technique that preserves the reason for each requirement from the perspective of the user[5]. User stories are real world examples of how one might interact with the system to fulfil a goal. They are developed to better step into the shoes of the user and what they might do when using such a system. Representing more refined requirements, user stories take the form of "As a [user type], I want to [requirement] so that I can [goal]" to not only show the requirements, but keep the developers focused on who the user is and why the task is important to them. Not only does this technique prevent exploring technology-oriented solutions in detail early on, but they also facilitate user-oriented work well into implementation by naturally partitioning large features into tasks that can be loaded into a backlog and distributed among developers[4].

User stories can be written more formally and include more detail in the form of use cases, which offer a detailed step-by-step description of the user's interactions with the system and how the system responds to each event. This back-and-forth conversation between the user and the system creates a flow that is representative of the user's experience and provides what may be the finest filter for missing links and features. It ensures that every system response is accounted for which leads to a more complete experience with consistent feedback and no user confusion. Use cases are more formal than user stories and they document the actor, stakeholders, conditions, triggers, happy path, and alternate scenarios requiring more effort to produce than the other artifacts in this discussion. However, the payoff of writing use cases is that they provide a more complete picture of users' interactions with the system[11].

Formal models and diagrams

Development of more formal models and diagrams provide the opportunity to describe the context and inner workings of a system which will eventually directly guide the implementation. This technique visually represents the ideas in user stories with the benefit of using Computer Aided Software Engineering Tool (CASE)[5]. These models can reveal the inner workings of a user's daily responsibilities and challenges at a level directly valuable to construction. How many, and what type of models are created can be directly related to the project scope as we aim to model what tasks are relevant and from all perspectives necessary. Unified Modeling Language (UML) is one such effective modeling technique with simple semantics and can be used to create a plethora of effective design documents. Like other design documents discussed in this chapter, sequence diagrams, dataflow diagrams, and class diagrams, to name a few, can be developed iteratively and be built into the existing iterations of Scrum and Kanban in flexible ways without slowing down the development process.

How the Framework can be Used in an Agile Project

Contextual design suggests that a foundational design should be established early on based on data that is user-centric. It has been found that no more than 20% of change will occur from one prototype to the next and these changes will rarely ever be the functional structure[4]. This is because rapid prototyping feedback is naturally focused on fine tuning the system that is already presented, so it follows that customer feedback will focus on this as well rather than challenging the system in its entirety. Therefore, it is integral that a basic design is created from customer data from the ground up to ensure developers are building the right product in their prototyping. The UX practices we proposed in this paper can be integrated into development sprints as illustrated in Figure 1.

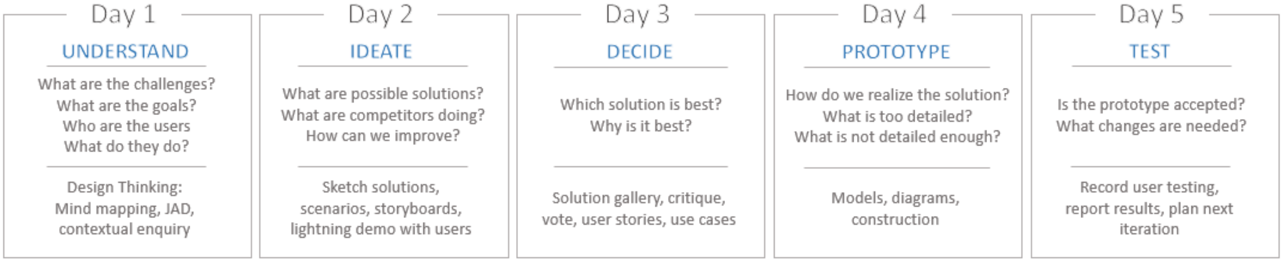

Figure 1. Integration of UX practices into development sprints

As Agile development phases are executed in a concurrent manner, user-testing does not have to be pushed to the end of a cycle. UX has a short lifecycle of its own which can be fit into a project's existing development life cycle to ensure usability is accounted for seamlessly and continuously. It begins with the analysis of the user base, the tasks each user will undertake using the system under development, and the environment in which they will be performing these tasks. All these activities will help to establish the user profile as discussed previously.

The first iterations of UX, conceptual model design, are appropriate for creating the vision with users, JAD sessions, and high-level mockups of the user interface (UI) that showcase options for and establish navigation in a system without contents. The iterations proceed with what should be enough user-data to begin the basic construction of architecture and UI. UX follows in parallel with this construction, adding information that should not interfere with construction, but rather, refine and correct it as the product becomes closer to what the customer requires. The first prototype need not be digital; user testing can commence with paper or card sorting to find the appropriate hierarchy of components in a process by which users are asked to complete tasks with what is available in the design[5].

Once the structure is established, screen design follows in the UI iterations, and should be implemented as a "live prototype". Designers follow page design standards while accounting for the activities each page is responsible for according to user-centric activities discussed in framework that should already have been completed in the conceptual model. At this point, UX fits in with Agile as the result of conceptual design becomes the working prototype developed upon and presented to users for feedback at the end of a sprint. Screen design is also completed iteratively within Agile to "refine and validate" the information display and such that it satisfies both need and usability. While filling out the contents of pages, it is important to have a defined style for the entire program to ensure "quality, coherence, and consistency"[4].

The final set of iterations can be grouped into detailed UX design which develops any previously unassessed functionality and continues to validate the program with established UX goals iteratively until it is fully accepted by users. Release planning directly follows passing user acceptance and can be streamlined with the development and prioritization of user stories. Each user story should represent a cohesive function of the product and therefore has the potential to be released when it surpasses each of these groupings of phases and passes acceptance testing. Since this process is meant to be completed iteratively, it is possible to reach this point for a user story in a matter of weeks, keeping pace with the fast style of Agile.

Validation of the Framework

As a means of validating the effectiveness of the framework and acceptance by developers using Agile frameworks, we developed a questionnaire and interviewed an engineering department's undergraduate senior capstone project teams. Project teams that qualified to be included in this study were customer-oriented in nature, included software components in their solution, and used an Agile methodology such as Scrum or Kanban. Beginning with interviews early in development, each team was briefed on the context of this research and provided with components of the framework to choose to integrate into their project iterations. The teams then participated in a retrospective analysis on the effect of introducing the chosen activities on their workflow and resulting project, the activities' adoptability, and to deliver the resulting design artifacts. The most significant topics considered were the success of the activity in accomplishing set goals, the adoptability into the Agile methodology, and the effort-to-impact score which was rated on a five-point scale which represented "not worth the effort" to "worth the effort."

One such team worked with the Food Bank to improve the company's inventory control system achieved significant improvements with their product management. The employers of the Food Bank had challenges including searching the products in the inventory as there were duplicate products, the errors in the entry of new products due to manual errors and use of paper forms. The senior design team engaged in JAD sessions with stakeholders, produced a mind map, and developed use cases to introduce usability in the analysis and design stages of development. They then developed prototypes of the system using the data they collected from the user experience activities. During the interview, the team expressed the JAD session was successful in revealing unconsidered features and details of relevant tasks that contributed to the team's understanding of the system environment and business processes. The activity reportedly significantly impacted the product by adding features and discovering points of pain of the end users while being considerably adoptable by the development team who suggested it fit naturally into their iterations. It was however less adoptable to the stakeholders who had less dedicated time to spend leading to the JAD sessions lasting only a couple of hours per session. The design thinking activity and production of the mind map rated similarly in impact on the product and was reported as more adoptable as it expected less time commitment. This activity was successful in setting the scope of the product and outlining business processes and possible challenges to explore. The stakeholder was also more willing to engage due to the visual aspect which resulted in a perfect effort-to-impact score. Finally, the development of use cases was used to "shape specific aspects" of the features discovered by the preceding two activities. They were considered highly successful in producing a "framework for construction" and ensuring all micro-tasks were considered. Reportedly fitting easily into the iterations, the use cases were a living document that went through refinements of their own as new information was discovered. Use cases also were attributed the highest possible effort-to-impact score. In general, the team reported that in using these activities, the information gained was "invaluable" and decidedly not "challenging to implement" into their iterations.

The selection of only a few methods to be used by a project team each year was due to the relatively short duration and limited scope of the senior capstone projects. A more complete investigation will be necessary to truly measure the effectiveness of such a framework. Future work may include introducing the complete framework to the designers before development begins, that is, pre-planning phase. For a true comparison between the results of using the framework and opting for a construction-driven approach, it may be valuable to have an additional development team work in parallel on the same project but omit UX activities. This would provide a baseline, or a control group, so that the effects of using the framework can be more apparent.

Summary and Conclusions

Since information systems have come into non-specialist use, the importance of application usability has only grown with the invasion of software-driven lifestyles. User Experience cannot be left as an afterthought to be accounted for in user testing. User-centric analysis and design activities can often be overlooked in favor of producing code in the modern, rapid development style of software engineering. Even so, there exist many processes supported by HCI that fit in the Agile approach taken today. User involvement in fact-finding, observation, and cognitive techniques produce valuable information for a foundation on which quality products can be built. The proposed framework is a guide to prevent usability from being an afterthought and provide a more appealing approach to user-centric and Agile development. In applying these processes that aid user-centric design, developers can produce usable and satisfying products without sacrificing the speed of development that Agile offers.

References

1. Norman, Don. 2015. “Why Are Apple’s Products so Confusing? They Ignore Design Principles | LinkedIn.” https://www.linkedin.com/pulse/why-apples-products-so-confusing-ignore-design-don-norman/?trk=eml-b2_content_ecosystem_digest-network_publishes-365-null&redirectFromSplash=true (November 14, 2021).
2. “15th Annual State Of Agile Report | Digital.Ai.” https://digital.ai/resource-center/analyst-reports/state-of-agile-report (January 20, 2022).
3. Choudhary, Shubhangi. 2019. “Participatory Design —Tools and Techniques : Purpose and Context.” *Medium*. https://medium.com/@i.shubhangich/participatory-design-tools-and-techniques-purpose-and-context-cc877790d4a6 (September 23, 2021)
4. Jacko, Julie A. 2012. *Human Computer Interaction Handbook: Fundamentals, Evolving Technologies, and Emerging Applications, Third Edition*. CRC Press.
5. Tilley, Scott R. 2020. *Systems Analysis and Design*. 12th edition. Australia ; Boston, MA: Centage.
6. Wallwood, Marty. 2020. “3 Design Thinking Exercises to Make Problem-Solving More Exciting.” *Medium*. https://uxdesign.cc/3-design-thinking-exercises-to-make-problem-solving-more-exciting-98bc3bb67350 (September 23, 2021).
7. Salazar, Kim. 2020. “Contextual Inquiry: Inspire Design by Observing and Interviewing Users in Their Context.” Nielsen Norman Group. https://www.nngroup.com/articles/contextual-inquiry/ (January 17, 2022).
8. “The Design Sprint — GV.” http://www.gv.com/sprint (January 18, 2022).
9. Lo, Gloria. 2021. “What’s a Design Sprint and Why Is It Important?” *Medium*. https://uxplanet.org/whats-a-design-sprint-and-why-is-it-important-f7b826651e09 (September 23, 2021).
10. Gibbons, Sarah. 2019. “Cognitive Maps, Mind Maps, and Concept Maps: Definitions.” *Nielsen Norman Group*. https://www.nngroup.com/articles/cognitive-mind-concept/ (January 18, 2022).
11. “Use Cases.” 2013. https://www.usability.gov/how-to-and-tools/methods/use-cases.html (January 19, 2022).

TORI N. GARDNER
Miss Gardner currently attends St. Mary’s University in pursuit of her MS in software engineering. She graduated with her BS in software engineering in May of 2021 from the same university. Her research interests include human-computer interaction, user experience, and user interface design as well as accessibility.

OZGUR AKTUNC
Ozgur Aktunc, Ph.D., is a Professor of Software Engineering and Graduate Program Director at St. Mary’s University Engineering Department. Dr. Aktunc’s main research area is software engineering with an emphasis on component-based software development. His research relates to software analysis, software metrics, agile development, and usability of web applications.