

Active Story: A Low Fidelity Prototyping and Distributed Usability Testing Tool
for Agile Teams.

By

PATRICK IAN WILSON

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE
DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

CALGARY, ALBERTA

UNIVERSITY OF CALGARY
FACULTY OF GRADUATE STUDIES

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies, a thesis entitled “Active Story: A Low Fidelity Prototyping and Distributed Usability Testing Tool for Agile Teams”, submitted by PATRICK IAN WILSON in partial fulfillment of the requirements of the degree of Master of Science.

Supervisor, Dr. Frank Oliver Maurer, Department of Computer Science

Dr. Eugene Kowch, Faculty of Education

Dr. Michael Richter, Department of Computer Science

ABSTRACT

Supporting low fidelity usability testing in small agile teams is challenging. Agile principles suggest avoiding upfront design if possible, while traditional usability practices tend to favor upfront design of user experiences. Many existing agile interaction designers complain that usability tests require too much overhead to organize the test participants. These difficulties include the large overhead associated with finding participants, and getting them on site and collocated with the usability expert when many short iterations are used. In order to integrate usability testing into an agile process, the amount of upfront work required must be minimized. This thesis presents ActiveStory, a tool for assisting agile user interaction designers in creating and testing low fidelity prototypes with a minimum of effort required to gather and organize participants. The tool allows designers to create low fidelity prototypes and then run usability tests over the internet with distributed participants. The tool will automatically collect usability statistics, such as mouse trails, page visit durations and user comments, while the test is underway. To evaluate ActiveStory, a qualitative pilot study was conducted with several academic teams who were designing and building web applications. The feedback collected was promising and most of the participants were very excited after using the tool.

Acknowledgements

There have been many people who have assisted me in performing research and writing this thesis. I would like to take the opportunity to acknowledge some of those people now.

To Dr. Maurer, thanks for all your advise and assistance. Without your guidance I could never have completed this research.

To the other researchers in the Agile Software Engineering lab, David, Robbie, Yasser, Xin and all the German students. Thanks for being so supportive and allowing me to ask you questions.

Finally, I would like to acknowledge the constant support I received from my family. Thanks for always encouraging and supporting me through my entire education

Dedication

To Michelle, whose constant support got me through.

Publications From This Thesis

Some of the materials and ideas presented in this thesis may have previously appeared in the following peer reviewed publications:

Chengyao Deng, Patrick Wilson and Frank Maurer: Fitclipse, A Fit-based Eclipse Plug-in For Executable Acceptance Test Driven Development, Proceedings of the 8th International Conference on Agile Processes in Software Engineering and eXtreme Programming (XP 2007), Como, Italy 2007 (Springer).

C. Deng, P. Wilson, F. Maurer: Fitclipse: An Eclipse Plug-in For Executable Acceptance Test Driven Development, Proc. on Eclipse Technology eXchange (ETX 2006), Interactive Poster, Oct. 2006

Table of Contents

ABSTRACT.....	i
Acknowledgements.....	ii
Dedication.....	iii
Publications From This Thesis.....	iv
Table of Contents.....	v
List of Tables.....	viii
List of Figures and Illustrations.....	ix
Chapter One: Introduction.....	1
1.1 User Centered Design.....	2
1.2 UI Prototyping and Usability Testing.....	3
1.3 Agile Usability Engineering.....	5
1.3 Motivation.....	7
1.4 Research Problem.....	8
1.5 Goals.....	8
1.6 Thesis Structure.....	9
Chapter Two: Related Work.....	10
2.1 Agile Interaction Design in Practice.....	10
2.1.1 Larry Constantine.....	10
2.1.2 Jeff Patton.....	11
2.1.3 Desiree Sy.....	13
2.1.4 Jennifer Ferreira.....	13
2.2 Existing Tool Support for Low-Fidelity Prototyping.....	16
2.2.1 Pen and Paper.....	17
2.2.2 SILK.....	18
2.2.3 DENIM.....	19
2.2.4 Serena Composer.....	20
2.2.4 Microsoft PowerPoint.....	20
2.2.5 Microsoft Visio.....	21
2.2.6 Intuitect.....	22
2.2.7 Axure RP.....	22
2.3 Existing Tool Support for Usability Testing.....	22
2.3.1 The Original Wizard.....	23
2.3.2 OzLab.....	24
2.3.3 Neimo.....	25
2.3.4 Morae.....	25
2.4 Summary.....	26
Chapter Three: Motivation for Tool Requirements.....	27
3.1 Web Survey.....	27
3.1.1 Objective of the Survey.....	27
3.1.2 Survey Design.....	28
3.1.4 Participant Demographics.....	29
3.1.5 Results.....	31
3.1.6 Summary and Requirements Gathered from Survey.....	32

3.2 Interview Based Survey	33
3.3 Summary of Gathered Tool Requirements	35
Chapter Four: ActiveStory	38
4.1 Tool Overview	38
4.2 Prototype Design Tool Overview	39
4.2.1 Drawing the Prototype	40
4.2.2 Adding Interactivity	42
4.2.3 Exporting the Design	44
4.3 Distributed Wizard of Oz Tool Overview	46
4.3.1 Gathering Participants.....	47
4.3.2 Evaluating the Prototype.....	47
4.4 Reviewing the Usability Data	49
4.4.1 Mouse Trails	50
4.4.2 Page Durations	51
4.4.3 Participant Comments	53
4.5 Implementation Details	54
4.5.1 Design Tool.....	55
4.5.2 Wizard of Oz Usability Test Tool.....	58
4.5.3 Integration via the ActiveStory Exporter	59
4.6 Summary	61
Chapter Five: Tool Analysis	62
5.1 State of the Art	62
5.2 ActiveStory Supported Requirements.....	64
5.3 Limitations of ActiveStory	65
5.4 Summary	66
Chapter Six: Qualitative Evaluation	67
6.1 Objectives	67
6.2 Study Methodology.....	68
6.3 Participants.....	70
6.4 Development of Projects.....	70
6.5 Discussion	71
6.5.1 Ease of Use	72
6.5.2 Pen and Paper Metaphor	76
6.5.3 Efficiency.....	79
6.5.4 Flexibility.....	82
6.5.5 Usefulness of the Data Gathered.....	83
6.5.6 ActiveStory vs Pen and Paper.....	86
6.6 Study Limitations.....	87
6.7 Independent Feedback	88
6.8 Summary	92
Chapter Seven: Conclusion.....	93
7.1 Research Motivations.....	93
7.2 Research Contributions.....	94
7.3 Future Work	95
References.....	97
APPENDIX A: ETHICS CERTIFICATIONS.....	102

APPENDIX B: WEBSURVEY QUESTIONS 104

List of Tables

Table 1: A comparison of the related tools to tool requirements.....	63
Table 2: Evaluation of direct verbal responses to the question "Did you find ActiveStory easy to use?".....	73
Table 3: Did the participants feel ActiveStory metaphored pen and paper.	76
Table 4: the responses to the efficiency research question.	79
Table 5: Participant responses for the flexibility research question.	82
Table 6: Responses to the question regarding the usefulness of the data.	84

List of Figures and Illustrations

Figure 1: The basic stages in traditional user-centered design	3
Figure 2: A diagram depicting the process used by Desiree Sy[5].....	13
Figure 3: A depiction of the findings of Ferriera regarding the possible implementations of Agile-UCD [22].....	14
Figure 4: an example movie storyboard [40].....	18
Figure 5: An example of a storyboard in SILK [39].....	19
Figure 6: Participant Demographics	30
Figure 7: An annotated screenshot of the design tool.....	41
Figure 8: Screenshot of the flip chart controls.....	42
Figure 9: A screenshot of the main window detailing the Activate button.	43
Figure 10: Sequence of events required to create an interaction.	44
Figure 11: A screenshot of the ActiveStory Design Tool with the export button highlighted.	45
Figure 12: The ActiveStory Wizard of Oz tool initial page.....	48
Figure 13: Entering comments in ActiveStory.	49
Figure 14: A mouse trail collected by ActiveStory.	51
Figure 15: Page durations in ActiveStory.....	52
Figure 16: A comment in ActiveStory.....	54
Figure 17: The model of ActiveStory.....	56
Figure 18: Typical file structure showing an imported file link.	57
Figure 19: ActiveStory prototype directory structure.....	60

Chapter One: Introduction

Agile software development has been picking up momentum steadily over the last few years, however, as more companies and developers adopt agile methods, an unease regarding the lack of attention to usability concerns has also surfaced.¹ Agile teams strive to produce high quality software for a minimum cost. They often are good at producing useful software – but sometimes usability takes a backseat.

Traditional approaches to usability engineering tend to suggest three phases of design (all of them being executed before implementation starts): low fidelity, medium fidelity and high fidelity. Low fidelity generally refers to rough sketches on paper or whiteboards, medium fidelity is more polished but still cannot easily be confused with a finished product and finally, high fidelity designs look polished and finished but lacks the underlying implementation of the business logic. This thesis relies heavily on the notion of a low fidelity (low fi) prototype. A low fi prototype is simply a very rough and incomplete representation of a target system. For instance, a common example of a low-fidelity prototype is using a simple pen and paper drawing. A drawing is quick to create, does not look like a completed application and, perhaps most importantly, it can be thrown away at little or not cost penalty.

A usability study is required to evaluate a design at any fidelity level. There are a number of ways to perform a usability study, however, nearly all of

¹ The substantial traffic in the agile-usability newsgroup at yahoo supports this statement.

them require the test participant to be collocated with the designer [3]. This is in part due to the inherent difficulty in collecting usability information in a distributed environment. Many forms of usability data, such as a participant's facial expression, can't be collected in a distributed nature very easily. However, the process of gathering participants and bringing them onsite so that the designer can perform a usability test with them tends to be difficult [41].

1.1 User Centered Design

User-centered design, (UCD) is a design philosophy that places the emphasis of interaction design on the user rather than on the interface. For example, rather than designing beautiful, aesthetically pleasing user interfaces that force the user to change his/her work process to use the system, UCD strives to adapt the user interactions to match the needs and process expectations of actual end users. UCD employs several techniques to help ensure that the final design is usable for the intended users. Some of these techniques include: observing real users perform real tasks (contextual inquiry), creating prototypes that satisfy the information gathered in the contextual inquiry, and finally, testing the prototypes with real user participants to validate the design.[3] Figure 1 shows the phases comprising the iterative user-centered design in a graphical flow chart.

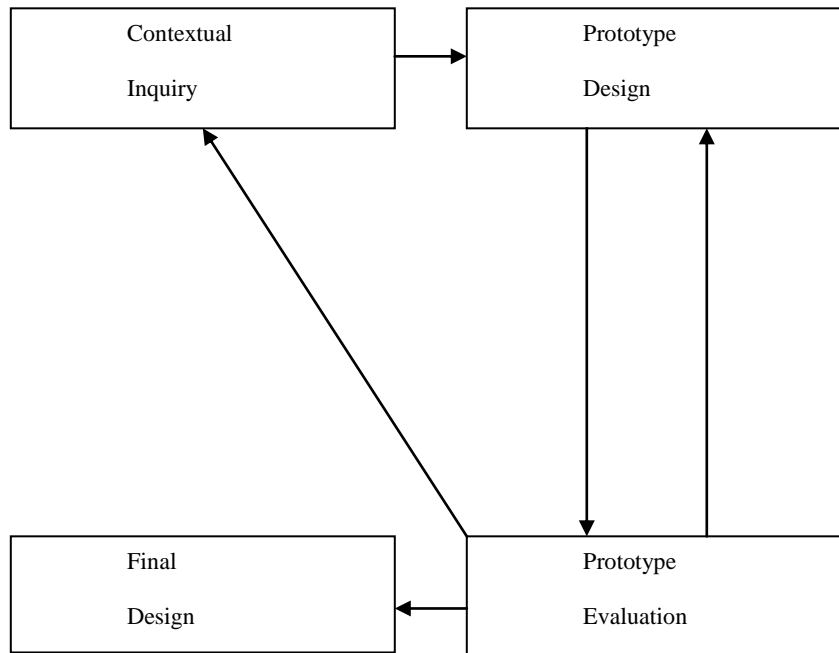


Figure 1: The basic stages in traditional user-centered design

Systems designed using UCD are more likely to be viewed as usable by final end users as the system interactions are tested with actual users to validate the design. Of course this statement is only true if the findings of the usability test are taken into consideration in the next iteration of the prototype design. There are a number of success stories of teams that have used UCD and produced more usable products as a result [20].

1.2 UI Prototyping and Usability Testing

Most interaction designers initially develop some form of low-fidelity user interface prototype. The most widely used method of developing these prototypes is to simply sketch drawings of what the different screens will look like with pen & paper and then get feedback on these from actual end users. This method has

also been applied to other forms of media such as dry erase whiteboards, digital whiteboards and even more sophisticated computer applications that allow designers to draw on the screens of tablet PC's [7, 8]. Some designers, however, prefer to use User Interface Builders such as Microsoft Visual Studio [9], Visio [10] or any other design tool that produces user interfaces which appear polished and (unfortunately) finished. Research has suggested that designers should avoid these types of tools in the early phases of prototype development for a number of reasons. Firstly, they tend to steer the designer and any test subjects, in the direction of superficial details, such as colour choice and alignment, and away from major design decisions like widget choice & placement as well UI (User Interface) behavior. Secondly, these interface tools can require more upfront work to denote the same information as a simple sketch. This statement is not only intuitively true but has also been demonstrated in research [3].

However, a low fidelity user interface sketch must be more than simply a pretty picture, it must help uncover usability issues with the design of the interface. Typically, interaction designers will run a usability test with the low fidelity prototype. These tests can range from a simple Heuristic Evaluation (inspecting the design against a set of usability rules), to a complete usability study with subjects and observers.

It is common knowledge in the usability field (and software engineering in general) that it is cheaper to fix a problem earlier rather than later. Thus, waiting until a working prototype of the system has been developed before testing means that the usability bugs discovered may not be fixed due to a lack of resources. It

follows that testing a low fidelity prototype (if possible) may catch major usability problems before any code has actually been written, and as a result the design can be fixed at a relatively low cost. There are a number of ways to test a low fidelity prototype, the most common approach being a Wizard of Oz usability test.

A Wizard of Oz test is typically used when little or no implementation has been done. A Wizard of Oz test will generally consist of a user, an observer and a “wizard” who controls what the user sees based on the user’s actions [3]. This approach is useful as a working software prototype is not required; however, Wizard of Oz experiments are expensive in terms of preparation, execution and evaluation [11].

1.3 Agile Usability Engineering

Recently, many agile and UCD practitioners started to work hard to unify usability engineering with agile practices. The basic conflict comes from the agile principle to maximize the amount of work not done colliding with the upfront user interface design and testing required by UCD.

During the 1980s, usability testing was “expensive, time consuming and scientific in approach” [3]. As a result, few companies were able to afford usability engineering. Nielsen countered this heavyweight practice by suggesting a methodology he called “Discount User Testing”. The main precept for Discount User Testing is that some usability testing is better than no usability testing and, further, sometimes a scientific style of testing is overkill. To quote Nielsen:

*In discount usability engineering we don't aim at perfection anyway, we just want to find **most** of the usability problems.[41]*

Nielsen suggested three methods of gathering feedback that are inexpensive (scenarios, simplified thinking aloud and heuristic evaluation) [41]. Scenarios (according to Nielsen) are very small prototypes that only define the parts of a system that are required to perform a pre-defined task. For instance, to test the save feature of a word processor, it is not necessary to create a prototype of the entire application but rather just the file menu and save menu item as well as a quick link button. Simplified thinking aloud simply refers to a form of usability testing wherein a test participant is encouraged to think out loud so that the evaluator can jot down some notes during the test. Heuristic evaluation is simply another form of usability testing that simply examines a user interface against a set of best practice guidelines. Nielsen backed up his discount usability approach by demonstrating the optimal cost to value ratio is achieved with approximately 3 to 5 test subjects [3,41,43], whether using Discount User Testing or a more traditional form. Clearly, discount usability engineering stands a better chance of adoption by agile practitioners than traditional usability engineering. Many agile teams have successfully adopted usability engineering, specifically low fidelity prototyping and usability testing in to their development processes [2, 4, 5, 6]. Larry Constantine [2] has described a low fidelity usability test, possible Wizard of Oz as part of his process. Jeff Patton's work [4] is an example implementation of Constantine's process. Patton specifically describes using low

fidelity prototyping and Wizard of Oz testing. Gerard Meszaros describes using Wizard of Oz testing in the same manner as Patton except that Meszaros' team had an additional member who acted as the "help system" [6]. Finally, Desiree Sy hints that her team may also use a form of Wizard of Oz testing to evaluate simple paper prototypes. Clearly, several well-known Agile – UCD practitioners are using both paper prototypes and Wizard of Oz usability tests.

1.3 Motivation

Currently, Agile – UCD practitioners are predominately using pen and paper based low fidelity prototypes to test interactions as they are designed. Due to the agile influence, design phases are kept as short as possible, where only the necessary features for the upcoming iteration of development are prototyped. In pure UCD, on the other hand, the upfront design phase lasts much longer and every feature in the system requirements are prototyped. As a result, in an Agile context, the evaluation of paper based prototypes must be performed swiftly and efficiently otherwise there is simply not enough time to perform low fidelity usability testing. As mentioned previously, the answer to this problem may be found in part by exploring Nielson's Discount Usability. Simply put, for agile teams some usability testing is better than none at all.

For agile teams, Wizard of Oz usability testing is the common method to evaluate an interaction design in its earliest stages. However, paper-based Wizard of Oz testing requires the test subjects to be collocated with the "wizard". Another weakness with traditional Wizard of Oz usability tests is that the system is not very convincing. It is hard for a test participant to believe that they are actually

evaluating future software when all they can see is several sheets of paper being manipulated by a human. Specifically, the motivations of this thesis are:

1. There is little published work regarding the tools agile interaction designers are using to perform their tasks.
2. There currently do not exist any tools that support agile interaction designers specifically.
3. If such a tool were to be built, there needs to be an evaluation performed on the tool to discover if the tool is more useful than simply prototyping with pen and paper.

1.4 Research Problem

The principle problem this thesis explores is to better understand how Agile and UCD work when paired together in the real world, which tools Agile Usability professionals are currently working with and most importantly, how could a tool be built that facilitates the common processes followed by most Agile Usability designers in such a way as to promote interaction design in teams that currently do not employ such techniques.

1.5 Goals

In order to solve the research problem presented previously, a tool that is designed to assist Agile Usability practitioners in designing low fidelity interactive prototypes, which can be evaluated efficiently enough to be used on small agile teams needed to be developed. The goals of this research and the tool itself are as follows:

1. Conduct a study to determine what tools (agile) interaction designers are using in industry as well as to better understand the requirements for a tool geared specifically towards agile interaction designers.
2. Using the requirements gathered from the initial survey, construct a tool that is specifically oriented for agile interaction designers.
3. Once the tool is built, conduct an initial evaluation to determine if there is a chance that the tool is useful in its purpose.

1.6 Thesis Structure

The rest of this thesis is structured as follows: first there will be a survey of the related literature and tool support, then the initial survey will be discussed and several tool requirements coming out of the survey will be presented. The next Chapter will present ActiveStory and describe some design decisions. Then an empirical evaluation of ActiveStory based on the objective tool requirements will be presented. Following that there will be a discussion of the pilot study that was conducted to find out if ActiveStory fulfills the requirements gathered. Finally, the thesis will conclude with a presentation of some contributions and related work.

Chapter Two: Related Work

The once popular notion that Agile Methods and Interaction Design can not coexist is quickly becoming a thing of the past. Of late, there have been a number of successful attempts to integrate UCD into Agile projects. In terms of published work, three names jump to mind: Larry Constantine, Jeff Patton, and Desiree Sy. It is interesting to note that while there is much in common with each of the processes described in these three publications, the backgrounds of the authors are very different. In two cases (Constantine and Sy) the author was a UCD/usability professional first and incorporated agile methodologies, while in the case of Jeff Patton, the opposite is true. However, at the end of the day, regardless of how the process was reached, all three processes share a lot in common. I will briefly discuss each of these publications to give some background to the defacto method in industry of integrating UCD into Agile methodologies. As well, some noteworthy tools will also be discussed.

2.1 Agile Interaction Design in Practice

This section of the thesis is designed to provide some background information regarding how agile methodologies and interaction design are currently being integrated together to produce more usable end products.

2.1.1 Larry Constantine

Back in 1995, Larry Constantine published the first paper on “Usage Centered Design”[21] (as opposed to User Centered Design). At this point in his

career, Constantine was not an Agile developer, but rather a usability specialist. Several years later, Constantine attempted to unify the lightweight nature of Agile methodologies with his light weight Usage Centered Design. The resulting process has laid the foundation for much other work to follow. Basically, Constantine's process simply explores the user and the user's roles in the system. Then, based on the roles discovered, user tasks are detailed and organized. Finally, a paper prototype is created from the user tasks and "refined" by usability testing. As Constantine admits

The assumption here is that development proceeds through successive release cycles. Successive iterations will pick up additional roles and task cases to guide refinement and expansion of the user interface. On each successive iteration or release cycle, then, the user roles, tasks, and co-operating clusters are reviewed and refined as needed before the next set of paper prototypes is sketched, inspected and refined. [2]

In other words, contrary to the practice at the time Constantine wrote this paper, he is suggesting that only the user interactions for the features to be worked on in the next iteration or release need be designed, as opposed to designing the interfaces for the entire system upfront before any lines of code are written.

2.1.2 Jeff Patton

Jeff Patton's work is very closely related to Larry Constantine's later work. Jeff Patton, however, did not start out his career on the usability side of the fence, but rather as a developer who first discovered agile methods and then usage

centered design. In fact, Patton credits his agile usage centered design process to a lecture by Larry Constantine that he attended [4].

Jeff Patton's process is really a more explicit example of Constantine's. It is important to note that Patton's process does not require large amounts of upfront work. In fact, Patton claims that the UCD process can be cycled in as short as a few hours [42]. Patton starts his process with an identification of participants and what he calls a preconception purge. He gathers all the programmers, interface designers, business people and testers in a single room and they are asked to "let loose". The preconception purge is about brainstorming possible solutions, getting bad ideas out in the open, and coming up with any useful features the new system should have.

Next Patton's team will review the domain in question in an attempt to learn as much about the kinds of task end users will be attempting to complete using the system. From that point on, the process follows very closely that of Constantine. Patton suggests defining user roles and creating a "role model". A role model is simply a collection of roles and the relationships between each of the roles in the system. From the role model, Patton then suggests defining tasks and a task model. Much like a role model, a task model simply defines the relationships between different tasks the users will have to accomplish. After the task model is detailed, a wire-frame pen and paper prototype is created and evaluated [4].

2.1.3 Desiree Sy

Desiree Sy, like Larry Constantine, was a usability expert first and an agile practitioner second. However, unlike Constantine, the change to Agile was a decision of the company and so the transition was an “opportunity to adjust, and consequently improve, our User-Centered Design practices”[5]. Sy describes the approach the company (AutoDesk) she works for employs: there are basically two parallel streams of work happening simultaneously, namely, the user experience design and the development. The development team will implement the designs that the design team created and tested in the previous iteration. In this way, the design team is always one step ahead of the development team, but at the end of the day, working software can always be delivered.

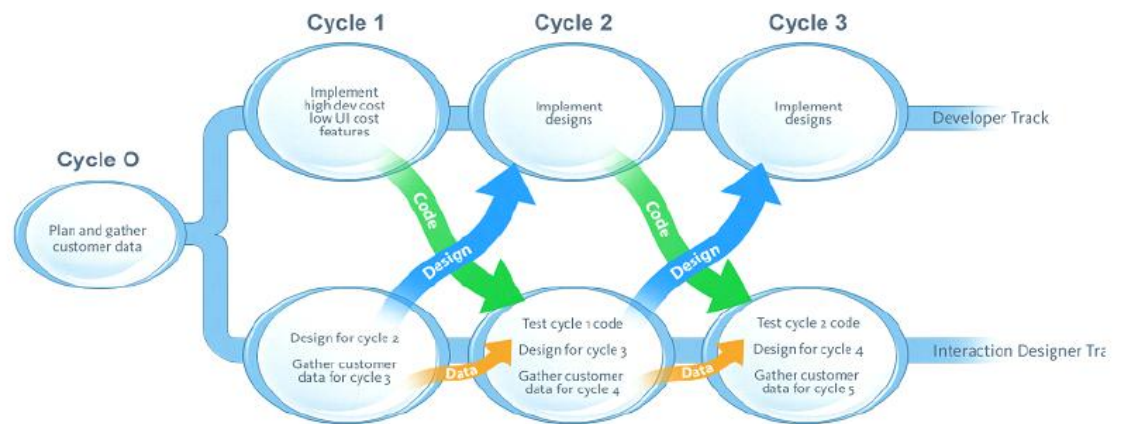


Figure 2: A diagram depicting the process used by Desiree Sy[5]

2.1.4 Jennifer Ferreira

The work of Jennifer Ferreira is noteworthy in the world of Agile Usability, in that she attempts to define “How real world agile teams combine interaction design with their agile development activities” [22]. Simply put,

rather than describing her own agile-usability process (like Patton and Constantine), Ferreira focuses her work on understanding how others incorporate interaction design into agile projects. Her findings are best described in Figure 3. This illustration demonstrates the different possible combinations of design strategy and implementation strategy that are used in actual practice.

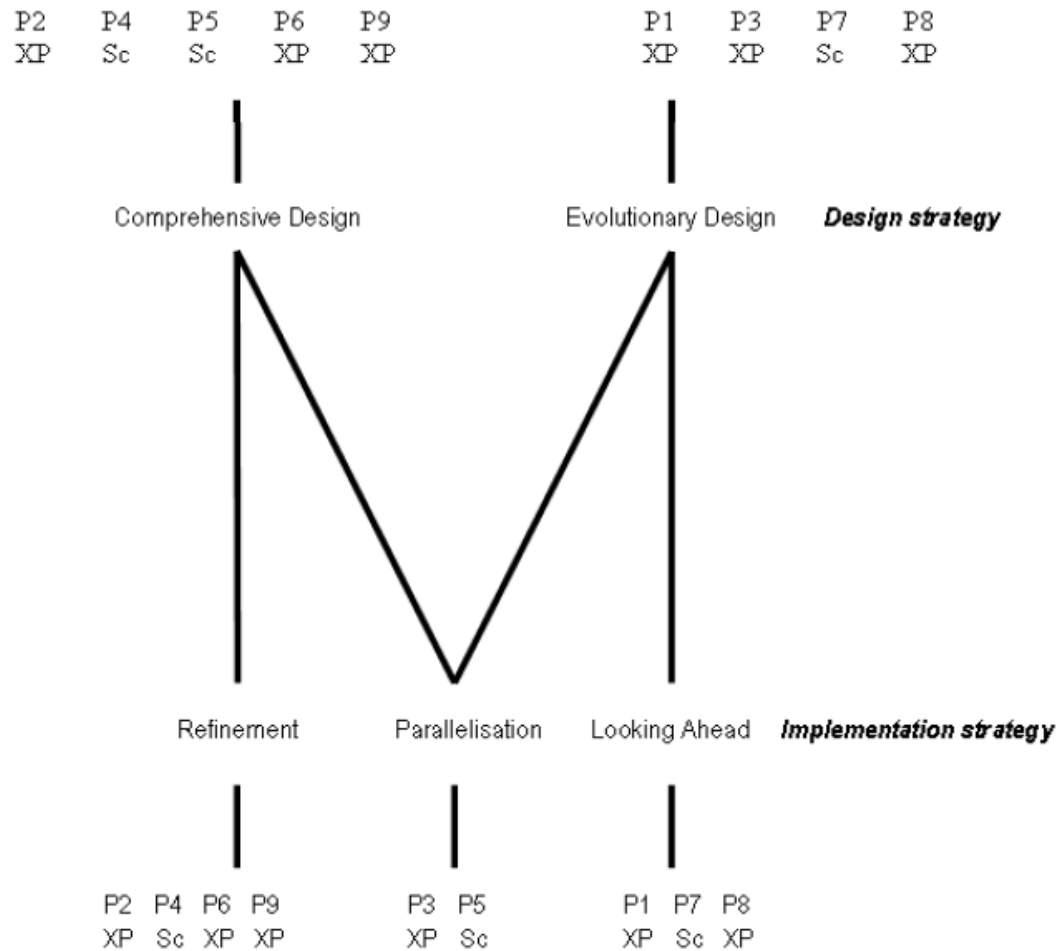


Figure 3: A depiction of the findings of Ferreira regarding the possible implementations of Agile-UCD [22]

Ferreira's work suggests that while the design strategy and implementation strategy affect each other, they can be chosen independently from one another. In terms of Design strategy, there are two basic options, Comprehensive and Evolutionary. Comprehensive refers to a complete upfront design of the user interaction while evolutionary design tends to allow the design to change during development. These two design strategies can then be paired with one of three possible implementation strategies: Refinement, Parallelisation, or Looking Ahead. Refinement simply means that an existing design is polished and fixed as implementation happens. Parallelism is described as a philosophy where the user interface is designed and implemented completed separately from the rest of the system. In other words, the user interactions can be designed and implemented in an either comprehensive or evolutionary method, while the rest of the system is developed iteratively. The final method for implementing interaction designs is "Looking Ahead". Looking ahead simply refers to the interaction designers working on prototypes and designs for the $n+1$ iteration of development. In other words, the developers are always a step behind the user interaction people, the interactions are designed, then the developers implement the designs in their next iteration.

Ferreira's work does have limitations (it is a little weak in defining the details of a specific interaction design process), but is a rather comprehensive look at how interaction design is being integrated into real world agile teams.

Another researcher at the University of Calgary is also examining a similar topic. David Fox is attempting to compare different processes used by

many agile individuals and companies in an attempt to define any common procedures. At present he has discovered that all the participants he has interviewed fall under one of three approaches to interaction design: a generalist approach, a specialist approach or a mixture of the two. The generalist approach to interaction design is to put many “blue collar” usability people (individuals who are aware of usability principles but by no means experts) on a project. The specialist approach is employed by companies who feel more comfortable with usability being handled by an expert in the usability field [46].

2.2 Existing Tool Support for Low-Fidelity Prototyping

Prototyping tools are an essential part of Interaction Design. Rapid prototyping provides a project three major benefits: Firstly, prototyping lowers the amount of energy required for an Interaction Designer to start his job. Secondly, it decreases the time required to create and evaluate an interaction design which accelerates iterative design cycles and finally, rapid prototyping allows for earlier feedback regarding the design [14]. To the best of my knowledge there are no tools designed specifically to assist agile interaction designers, however, that does not mean that there are not any tools that can be used for this purpose. This section explores the field of low fidelity prototyping tool support for interaction design and lays the groundwork for improvements to be made.

2.2.1 Pen and Paper

Perhaps the most widely used tool in any design field is a simple pen and paper combination. Pen and paper is a fast and inexpensive medium to design with.

In the world of pen and paper design, a common method of depicting interactions or a story is to use a storyboard. Storyboards are commonly used in the movie and game industries to quickly describe a scene. The director wants to get an idea for how the scene will look before expensive Actors and film crews are put on the set. This same principle is used by several interaction design tools (described in more detail later in this chapter).

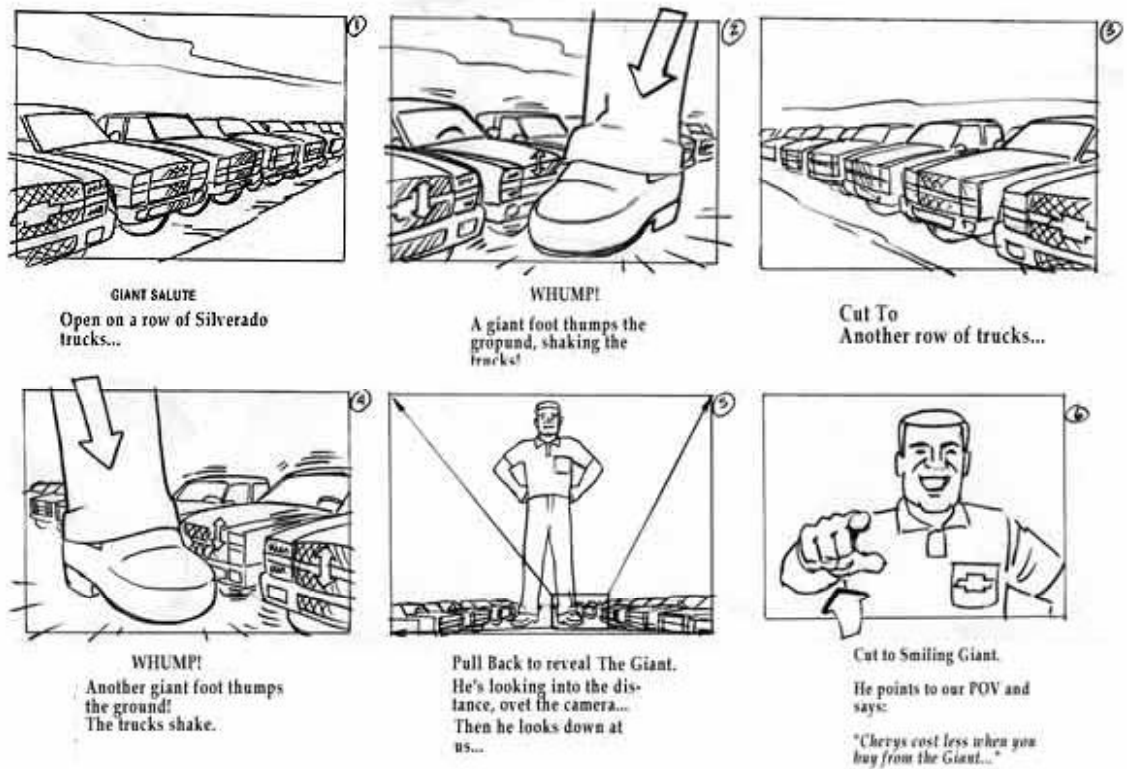


Figure 4: an example movie storyboard [40].

2.2.2 SILK

SILK [7] was one of the first tools in the field to take advantage of a pen and paper metaphor for computer assisted design. SILK allows designers to sketch designs and create interactive prototypes. Interactions in SILK are defined by drawing “storyboards” [23]. A storyboard, in SILK, is a graphical representation of a desired system state change to be enacted when the user triggers the change. An example of a storyboard sketch in SILK can be found in Figure 5.

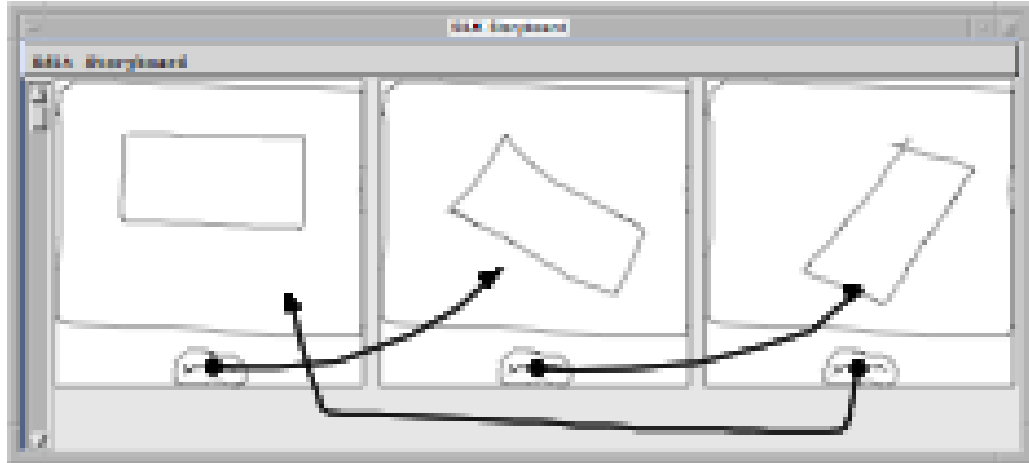


Figure 5: An example of a storyboard in SILK [39].

SILK starts to stray from the pen and paper metaphor with its widget recognition feature. SILK attempts to recognize widgets based on line drawings in the low fidelity prototype. There are a number of benefits from widget recognition, the most notable being that silk can allow interactions with a widget if it knows what type of widgets exist. For example, suppose a designer draws a scrollbar on an interface, if SILK knows that it is a scrollbar, it can allow test users to interact with the drawing as if it were a scrollbar. Widget recognition also allows SILK prototypes to be automatically increased in fidelity without requiring a complete rework. SILK is a prototyping tool, the prototypes created by it do not attempt to collect any usability data.

2.2.3 DENIM

DENIM [14] extends SILK by tailoring the design process to website designers. It provides designers with a site map that can be edited. DENIM also supports a zooming feature that allows the sitemap to “zoomed” into and thus provide the designer with a way of adding detail to different pages on the site.

Navigation from one page to another page is defined in essentially the same manner as in SILK, via a storyboard type interface. Both SILK and DENIM are especially relevant to this thesis because they are one of the few tools that allow designers to sketch interactive prototypes. DENIM designs can be executed in run mode, however, they can't be executed in a distributed nature. These prototypes, like SILK Prototypes, do not automatically collect usability data.

2.2.4 Serena Composer

Serena Composer is an example of a process modeling tool which also supports user interaction creation. It is however, very model driven and as a result probably requires more upfront design than an Agile team would be comfortable with. The tool also has a lack of basic drawing support and it can't export a design into a stand-alone prototype [28]. On the positive side, Serena Composer is one of the few tools that allows for conditional logic to be incorporated into the user interaction prototype [28]. Serena Composer also does not support usability data collection.

2.2.4 Microsoft PowerPoint

Microsoft PowerPoint [24] is another commonly used tool for creating low-fidelity interactive prototypes. PowerPoint prototypes are not typically pen and paper style but rather a more refined flavor of low fidelity prototypes. Designers can quickly create the screen elements using shapes, images and text and then link the different slides together using selectable regions. As most people have access to PowerPoint, these designs theoretically can be (and have been) tested in a

distributed environment. Two issues with this process can easily be uncovered. First, as research has shown, most agile design teams prefer to work with pen and paper or whiteboards for creating low fidelity prototypes [4,5,6,22] (although many do like the prototype interaction provided with PowerPoint). The second downside is that no method exists within PowerPoint to gather data about how the prototype was actually used, thus, no usability data collection is possible in a distributed environment.

2.2.5 Microsoft Visio

Another commonly used prototyping tool is Microsoft Visio [10]. Visio is a product for vector-based diagramming. Usability engineers have used Visio to create low fidelity prototypes in a similar fashion to using PowerPoint. Interactions can be added to Visio diagrams by adding “hyperlinks” between pages in a Visio project. The Visio design can also be exported to HTML, where it can be tested in a web browser. Visio exports a design to html by simply rendering all the features and controls in the design into a single image and then overlaying the hyperlinked regions, this provides the illusion that a fully functional (read only) prototype has been created [25]. It should perhaps be noted that prototypes created in this manner are read only, meaning that the user is not able to enter data or manipulate widgets, the only interactions used in practice are hyperlinks. Visio prototypes also do not support usability data collection.

2.2.6 Intuitect

Intuitect [26] is an html design plugin for Microsoft Visio. Intuitect allows designs created in Visio to be exported into a more richly functional prototype. For example, a basic design created and exported in Visio (without Intuitect) will be essentially an image in a web browser, while the same design exported with Intuitect will have interactive form widgets that the user can enter text into. Intuitect, like Visio, does not collect usability data.

2.2.7 Axure RP

AxureRP [17] overcomes many of the weaknesses encountered when prototyping with PowerPoint, Visio or any other tool not specifically designed for creating prototypes. Axure allows designers to create high fidelity prototypes that can be exported to html format. Unlike PowerPoint and Visio, Axure supports different types of input events, specifically: click, mouse enter and mouse leave [18]. However, it does not support usability data collection.

2.3 Existing Tool Support for Usability Testing

In the previous sections, I have discussed how agile practitioners in the real world have been incorporating interaction design into their methodologies. I have also described several tools to assist user interaction designers (both agile and not agile) in creating prototypes. Next I will cover a number of tools designed to assist interaction designers in evaluating user experiences.

2.3.1 The Original Wizard

Wizard of Oz testing was first developed (formally) for use with natural language computer applications [29,30]. Kelly describes in his papers, a project to create a natural language computer application for interactive calendaring called CAL: Calendar Access Language. The methodology he used to evaluate the interface (spoken interaction) resembles the famous scene at the end of the 1939 film “The Wizard of Oz” where the character of the Wizard is revealed to be nothing more than an old man controlling an illusion of a wizard, which is probably why Kelley termed the process the “Oz Paradigm” [30]. The Wizard of Oz technique has also been used to evaluate natural language applications in the automotive setting [31]. This process lends itself very well to natural language applications because the computer is relatively disguised from the end user, however, the same basic procedure has been used for visual interaction testing as well [3,4,2,5,6]. Since the initial creation of the idea in 1985, Wizard of Oz testing has been adapted to meet the needs of several specific domains. One such example is the tool Topiary [14]. Topiary is a Wizard of Oz testing tool designed specifically for Location Enhanced Applications. The field of Location Enhanced Applications have many difficult problems to solve. Thus a prototyping tool which allows for early feedback on a design which will not be realized in an actual working system for an extended period of time is a real asset. For the purpose of this thesis, however, I am treating Topiary as a proof of concept that visual Wizard of Oz tools are possible and are an asset. Topiary is too specific in domain to be of general help to Agile teams, but if a generic Wizard of Oz tool

could be created, it would greatly assist designers for the same reasons Wizard of Oz testing has assisted designers in the past.

2.3.2 OzLab

OzLab [32] is a tool designed to assist user interaction designers in evaluating possible design decisions that are either too costly or complicated to create a working prototype for. The OzLab system provides a method for creating a mockup prototype that can be manually controlled by the “wizard”, another person who is controlling the system in a manner that is convincing to the user testing the system [33]. It should be noted that OzLab has an interesting ethical issue because the wizard is deliberately deceiving the test participant into believing that the system is making decisions that it is not in fact capable of making [33]. This is a small matter, but it should be pointed out as most of the other tools discussed do not have this issue. It should also be mentioned that while OzLab appears to the test participant to be functioning on its own, it actually still requires a real human to pull the strings in the background, which could be an issue for small agile development teams. However, on the plus side, because a human is controlling the system, the prototype is able to emulate very complicated interactivity that might otherwise be too much overhead to try to incorporate into a working prototype. It is not clear from the publications on this subject if OzLab is able to automatically collect and record usability data.

2.3.3 Neimo

Neimo is a computer application that attempts to support a standard usability testing process [34]. Most usability labs are not digitally supported but rather resemble a combination of a music and film studio. Test participants are filmed from several angles simultaneously, their facial expressions and the contents of the monitor are filmed, and generally even the sounds the user makes are recorded [3]. Neimo, attempts to take this complicated process and attempt to automatically synchronize all the data so that it can be used more effectively. Neimo, like OzLab, also supports Wizard of Oz testing in its more purist form. Test subjects use an incomplete system that a Wizard is manipulating behind a curtain. However, unlike OzLab, Neimo supports the idea of multiple wizards. For instance, it is not difficult to imagine an interaction that might be difficult for a single person to handle on her/his own. It could also be the case that several wizards are responsible for only one interaction, thus multiple interaction can be handled simultaneously. Neimo also supports multimodal forms of input. In other words, Neimo attempts to assist wizards in dealing with multiple yet simultaneous forms of input, a feature not included in OzLab.

2.3.4 Morae

Morae [36] is a novel tool released by the Michigan based company TechSmith. Morae, like Neimo, is a comprehensive suite of tools designed to act as a usability lab. Where Morae become an interesting solution for agile teams is in its use of cheap, available equipment. Morae does not require a full usability lab to record with. Morae was designed with inexpensive forms of recording

tools in mind. Morae uses web cameras, built in microphones and screen capture technology to provide similar data to Neimo, but for a fraction of the cost. However, the administration of Wizard of Oz tests is left much to the test administrator.

2.4 Summary

In this section, I have discussed several agile interaction design methodologies used in practice. I have also provided a literature and tool survey regarding tools that an agile team could potentially use for creating and evaluating prototypes. It is interesting to note however, that all of the prototype evaluation solution previously discuss require at least one person per test participant to administer the usability test as none of the tools distribute well. This fact also requires test participants to be collocated with the administrator of the usability evaluation. The final point of interest relates to the usability data that the Wizard of Oz tools collect. All of the testing tools discussed collect raw usability data (video, sound and screen captured video). Nielsen has suggested that recording, watching and analyzing these types of data are expensive and take away time which could be better spent by testing with more participants [41]. These are weaknesses that my thesis work aims to overcome.

Chapter Three: Motivation for Tool Requirements

In the previous chapters, I have discussed the most prevalent agile interaction design techniques and procedures. I have also presented some related tools that exist that may be utilized to support these practices. In this chapter, I will present the results of a web survey and formal interviews as both motivations for the thesis work as well as a platform to gather requirements for the tool itself.

3.1 Web Survey

In May-June 2007, a web survey was conducted to gain an understanding of how both agile and traditional teams create and use low fidelity prototypes. One purpose of this study was to gather tool requirements for a prototype creation tool. The web survey featured both quantitative and qualitative questions regarding tool support and processes for Agile Usability engineering. Specifically, what tools the respondents both had used and preferred to use, and what features the respondents would find useful or hindering in a tool.

3.1.1 Objective of the Survey

The objective of the survey was simply to gain insight into the world of agile usability. I had access to specific examples of agile interaction design through a literature review, but that data lacked answers to specific questions regarding tool support and how different teams were using prototypes. For instance, it is clear from several published accounts [4,6,5] that low fidelity prototypes are used in the user interaction design cycle, however, it is not clear if

the prototypes are used to their full potential or if the teams are limited by a lack of tool support.

In addition to gathering an understanding regarding how agile teams use tools in their design process, the web survey was also aimed at understanding which tools were used in practice. The previous chapter discussed many tools that exist and can assist designers in creating and testing low fidelity prototypes, but the literature is not capable of revealing which (if any) of those tools are actually used in practical application.

3.1.2 Survey Design

Before conducting the web-based survey, ethics approval was obtained from the University of Calgary. A copy of the web survey questions has been included in Appendix 2.

A web survey seemed appropriate as the goal of the survey was to understand basic usage and characteristics of tools used in multiple industrial settings. Web surveys are an excellent method for eliciting feedback from a large group of participants (compared with typical interview style surveys). This is especially true if the nature of the data being collected is quantitative or at least limited in scope. As the target audience for the survey was agile user interaction designers, it made sense to recruit participants via the Agile Usability Yahoo Group [37]. The web survey itself was located on a publicly accessible web server located at the University of Calgary.

The survey was comprised of three sections, with respect to the aim of each question. The first section asked questions regarding the role and

methodologies of the participant. For example, was the participant responsible for Interaction Design or were they simply trained in that field. The first section also asked the participants to clarify whether or not they followed agile principles in their daily work (at least as far as interaction design is concerned). The second section asked questions related to tool support in use. This section was designed to shed some light on the kinds of tools designers/developers were using to create low fidelity prototypes. Specifically, each participant was asked to list the tools they were currently using or had used in the past as well as list the tools they preferred to use. The final section of the survey was designed to gather requirements for the construction of a user prototyping tool geared specifically toward agile interaction design. In all, the final section contained four long answer questions. The first question asked the participants to describe any feature for a tool they felt would be helpful. Next they were asked the opposite question, what features would be a hindrance. The third question, simply put, asked the participants: If you could snap your fingers and a tool would be built, what would it do? Finally, the participants were given a method to submit any final comments. In all, the agile-usability yahoo group proved to be a good avenue for recruiting participants. In total, there were 23 responses to the survey.

3.1.4 Participant Demographics

The participants of the web survey came from the Agile Usability Yahoo group. The first section of the survey inquired about the development practices and professional responsibilities of the participants. The first question (Q1) in this section asked “Are you a developer who practices Agile methodologies?”

The second question (Q2) asked “are you or have you ever been responsible for user interfaces?” Finally, the last question asked of the participants (Q3) was “Would you consider yourself a specialist in the field of usability.” The participant responses are presented in Figure 6.

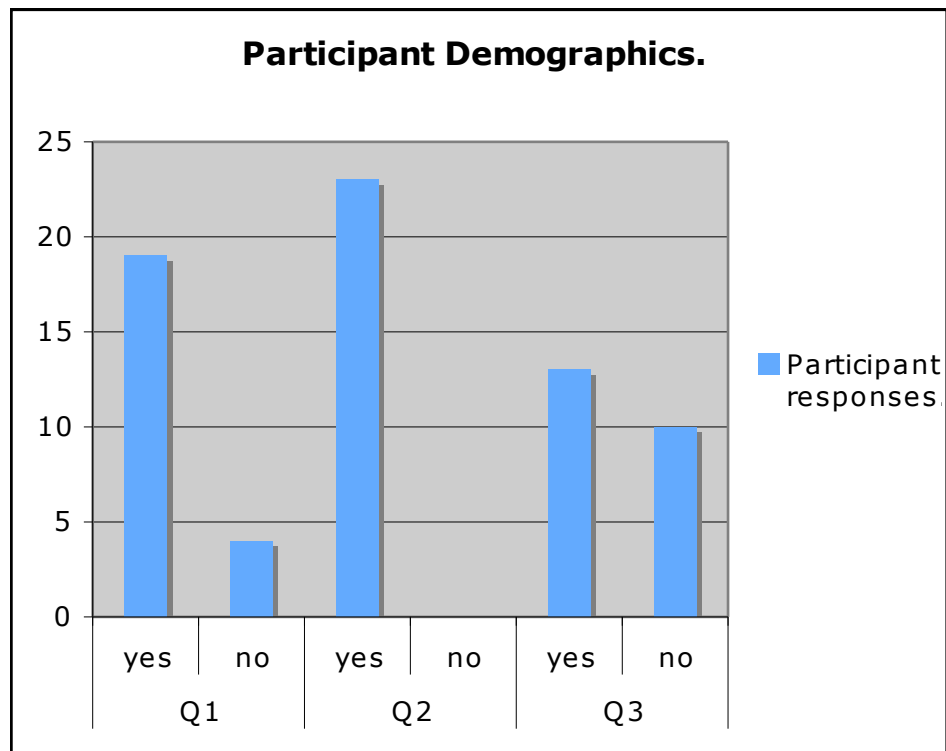


Figure 6: Participant Demographics

As can be seen from Figure 6, all of the participants of the survey were responsible for usability at one time or other. The responses are fairly split regarding whether the participants considered themselves usability experts (Q3). The vast majority of the participants did indeed practice agile methods (as should be expected from members of an Agile Usability group).

3.1.5 Results

When asked what tools the participant had used, I received many different answers, ranging from pen and paper to Microsoft PowerPoint to DENIM and many others. However, when asked what tool they preferred to use, two responses dominated the field; namely: pen and paper (21/23), and whiteboards (23/23). It cannot be said decidedly why teams seem to prefer using physical drawings to computer tools, but intuitively a possible explanation may lie in the friction between the process used by designers and the process imposed by the various tools. In other words, designers will become frustrated with a tool that restricts their creative thought process by requiring them to follow a specific process, rather than allow for process flexibility. Several (4) of the subjects in our web survey echoed this sentiment when asked about what features they would consider a hindrance: “Anything that slows me down”.

The other interesting finding from the survey relates to prototype interaction. Several of the respondents (4) mentioned prototype interaction as a useful feature or at least, a lack of interaction was a hindrance. One respondent complained of Axure “publishing required to see interactive prototype”. This suggests that this respondent certainly wanted to have an interactive prototype. Other participants asked for the “ability to make buttons and links hot so that one can click from prototype to prototype” and stated that they want to “Be able to use the low fidelity to demonstrate moving from screen to screen or things happening. Ability to move from one low fidelity to another based on a button click or some other event”. Most profound of all, a participant clearly indicated that a major

deterrent for using low fidelity prototypes was “they’re non-functional.” Clearly, low fidelity prototypes must not only be fast to create, but must also be interactive.

But why not use high fidelity prototypes instead of pen and paper drawings? Many respondents of the survey mentioned a recurring problem with high fi prototypes: “it is too easy to go to high fidelity and either a. spend too much time on the prototype (e.g. this often happens with Photoshop) or b. get the wrong kind of feedback on the prototype (why is this blue? You are missing a field here, etc.)” In other words, a hi-fidelity prototype looks too completed and thus elicits the wrong kind of feedback (feedback regarding the look and feel of the application rather than feedback about more serious usability concerns) from usability test participants. A second issue with hi-fidelity prototypes is the complexity involved with updating hi-fi prototypes. As one participant explained “you can have a group fiddling with a paper prototype - much harder to do with something on screen - especially if it needs a developer to change it”. It seems clear from this response that the prototype needs to be interactive, but should not require any actual development to create or modify it.

3.1.6 Summary and Requirements Gathered from Survey

The web survey uncovered many interesting tool requirements. It highlighted that low fidelity prototyping has many benefits in early testing over hi-fi equivalents and indicated that prototypes must be interactive. Respondents ask for preserving the pen and paper metaphor. The majority of survey

participants prefer to use pen and paper or whiteboards. Finally, the tool should not impose a design process on the designer.

3.2 Interview Based Survey

In addition to the web survey, another source of motivation and tool requirements were in the form of semi-structured interviews. In the study, interviews were in-depth, semi-structured, and performed face to face or on the telephone with participants from Canada and the United States.

A number of people responsible for interaction design on a number of projects were interviewed. Specifically a User Centered Design Specialist, P4, who had formal UCD training, an Agile developer P1, who had some informal UCD training, one business analyst, P3, with some exposure to the agile methods process and informal UCD training, and finally one information architect (IA), P2, also with UCD training, development skills and agile methods experience. All of the above participants were on different development teams working for different companies with the exception of P1 and P3 who worked for the same company but on different projects.

The interviews uncovered interesting trends regarding how designers recruit and use participants for usability tests. In many cases, bringing participants onsite to be part of a usability test seemed to be a problem. Several of the professionals interviewed related that they tended to “save” users to later in the design process to conserve resources (presumed to refer to time on the part of the participants and the designer) [P1, P3]. P4 also raised the point that participants are not “always available when currently you’re in the process of design”.

Another professional extended that same argument to a distributed scenario and expressed concern regarding how to recruit participants when “you’re in a situation where the customer and the developers are very far apart from each other”. In this instance, the developers were attempting to consult with the customer about a usability concern. Even influential agile user interaction professionals have issues recruiting participants for usability tests and as a result tend to group many features together to be tested rather than test each feature as it is designed. Clearly, a tool which can assist designers in gathering feedback from users who are not located in the same room as the designer would be a major help to many agile interaction designers.

Another concern that was raised relates to the functionality of a traditional Wizard of Oz prototype. Both participants P2 and P4 expressed the concern that a couple of drawings on pieces of paper simply do not capture user experiences very well. “Then we will build a prototype. Because it is just too hard to see a series of pictures and really understand what is going on. So we’ll get some sample content and we’ll build something, either in HTML or whatever it takes just to build the simple prototype [where] there is really nothing in the back end but at least you can feel the interaction design.” [P2]. Participant P4 summed the sentiment up perfectly by stating “some people only look at stuff on a computer screen”. In both these cases, the low fidelity design is done on paper, but a more expensive prototype is then developed to test the design. I conclude that a tool that allows interactions to be added to simple drawings would be a useful asset to these agile design teams. In addition to interaction, some form of running a

usability test without requiring test participants to be brought in would also be extremely beneficial.

3.3 Summary of Gathered Tool Requirements

A problem with standard pen and paper based Wizard of Oz usability tests is that they not only require a human oracle to “pretend” to be the computer, but also require test subjects to be present on site with the “wizard”, which can be difficult to facilitate and costly to setup, specifically when agile teams use short iterations (2-4 weeks).

Based on feedback received from the web survey and qualitative data gathered in interviews, I assembled the following list of tool requirements:

- **Easy to Use** – The tool must not only be easy and intuitive to learn and use, but must also be flexible enough to allow designers to test non-trivial interactions (at least to the level of interaction possible with pen and paper drawings).
- **The pen and paper metaphor must be preserved** - A designer using the tool should feel as though the process is very similar to what she/he has already been doing with pen and paper.
- **Prototypes must be fast to create and test** – The process of creating and testing prototypes must be fast and painless.
- **Flexible** – As the results from the web survey suggested, in order for a tool to be useful, it must be flexible in terms of the design process.
- **Allow for feedback from off-site participants** – A previously discussed, a major concern with not only Wizard of Oz testing, but usability testing

in general is the hassle of bringing in participants. The tool should allow for usability testing in a way that does not require participants to be collocated with designers/wizards.

- **Prototypes must be interactive** – The tool must provide designers with a method of incorporating interactions into the prototype without a “wizard” being present for testing.
- **Distributed Data Collection** – The tool must collect data useful in evaluating the usability of the design in question. This is not an easy task because, as our research has suggested, that the data collected during a WOZ usability test tends to be dependent on the domain and user goals of the application under test. Nevertheless, we came up with three streams of data that are valuable for determining the usability of an application. These three streams of data were derived from related work.
 - **Time frames** – how long was the user stationary at specific pages. Task duration metrics are among the most commonly used usability metrics [45], page durations are simply a more fine grained version of the same metric.
 - **Mouse tracks** – where did the participants mouse move while on a page. Research has suggested that there is a strong relation between a users mouse cursor and the position on the screen that the user is looking [44].

- **Direct feedback from participant** – the tool must allow a participant to provide direct feedback (e.g. alert the designer that a menu button doesn't seem to go where she/he imagined it would).

Chapter Four: ActiveStory

ActiveStory is a tool developed for the purpose of designing and performing usability testing on an application in a manner that is inline with agile principles. Our tool allows designers to sketch user interfaces, add interactions and finally administer the usability test over the Internet via a built in web Wizard of Oz system. The remainder of this section is divided into four sub sections. Firstly, a brief overview of ActiveStory is presented, describing the purpose and high-level goals of the tool. The second section describes the design tool, the third section provides details about the Wizard of Oz online usability-testing tool, The final section contains a discussion regarding the data collected by the ActiveStory usability-testing tool is provided. The final section describes some implementation details about the tool.

4.1 Tool Overview

ActiveStory is a stand-alone application that allows user experience designers in agile teams to create low fidelity prototypes digitally and then assist in administering usability testing on these prototypes. The tool has two major components that can operate independently of one another: the prototype design tool, and the usability test tool. The focus of this research was placed on the online usability testing components as there are many different prototype design tools and, while none of these tools meet all of the requirements laid out in the previous chapter, there are no tools that assist in distributed low fidelity usability testing.

4.2 Prototype Design Tool Overview

The ActiveStory design tool was created out of necessity. A tool was required to produce a prototype that could be executed in a distributed nature and no other prototype design tool both fulfilled the requirements gathered from the web survey or the qualitative interviews, and was capable of producing a prototype that could be executed in a distributed manner. As a result, the ActiveStory design tool was created to serve these purposes.

ActiveStory prototypes are extremely simple state machines that accept a single form of user input, mouse clicks. Another way to think of it is to visualize a prototype as an interactive storyboard. This may seem too trivial to create a working prototype with but in reality, a low-fidelity prototype is meant to be just that: low-fi. It should not attempt to evaluate complex user interactions, but rather focus on basic layouts and navigation. A click on a specific location in one frame of the storyboard takes the user to another frame.

The foundation of the prototype is a series of images. These images are displayed to the user during the usability test and the tool does not try to discern if there is any semantic meaning in the images. In other words, ActiveStory does not attempt to recognize widgets or any other components of the drawing. The design tool must be expressly told what parts of the image are “clickable”. ActiveStory represents a link as a region in which a mouse click will trigger a state change. Thus links can be thought of as a bounding box coupled with a

reference to an image entity. It should be noted that multiple links could point to the same image.

The following subsections describe the ActiveStory design tool in more detail.

4.2.1 Drawing the Prototype

ActiveStory's design tool allows interaction designers to quickly create a prototype that can be tested in a distributed nature. The ActiveStory design tool allows designers to quickly draw user interfaces and link them together. It is important to note that there are two forms of input for the design tool: tablet and importing an existing image. The usual form of input is a pen tablet, either a side tablet or a tablet PC. These technologies are becoming very affordable and are thus within a reasonable budget for an Agile team.

When using a tablet, the pen and paper metaphor is strictly maintained in that there are literally two modes for drawing, pen and eraser. This decision was based on the results of the web survey, which suggested that a tool needed to be fast, easy to use and not have features which slow down design. The user interface of the design tool is specifically catered toward a user with a tablet. The buttons are large for easy clicking. There are no drop down menus or complex forms of interaction. Figure 7 shows a screen shot of the design tool. The second form of input is geared towards users who either do not like using a tablet, or have already drawn a low fidelity prototype on physical pen and paper. The tool allows designers to import existing images into the prototype and then further

enhance them (if they so wish). If the user has access to a scanner, physical paper images can be scanned and then imported into the system.



Figure 7: An annotated screenshot of the design tool.

A flipchart metaphor is used to manage the many interfaces in a design. A flipchart is simply a large pad of paper that is positioned vertically (ie, the pages fold over the top of the pad) and then pages of the pad can be drawn onto. It is important to note that the order of the many pages is maintained, rather than loosely organized as would be the case with a pile of poster pages. Many sketching applications (i.e. SMART notebook [16]) use this same metaphor, so I decided not to stray from it. The ActiveStory's flipchart has one extra feature that is not found in the physical equivalent, the ability to make a copy of the current

page and place it either before or after the page currently being viewed. Figure 8 details the flipchart controls as well.

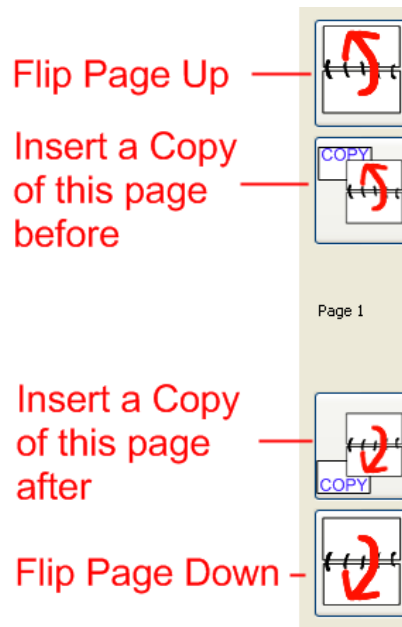


Figure 8: Screenshot of the flip chart controls.

4.2.2 Adding Interactivity

Once page designs are created, interactions can be added to the design. In ActiveStory, an interaction is simply a region on a page that, once clicked upon, causes the system to load a new page. The ‘Activate’ button is provided to specify the region of the current page that is to be activated. Figure 9 shows the whereabouts of this button.

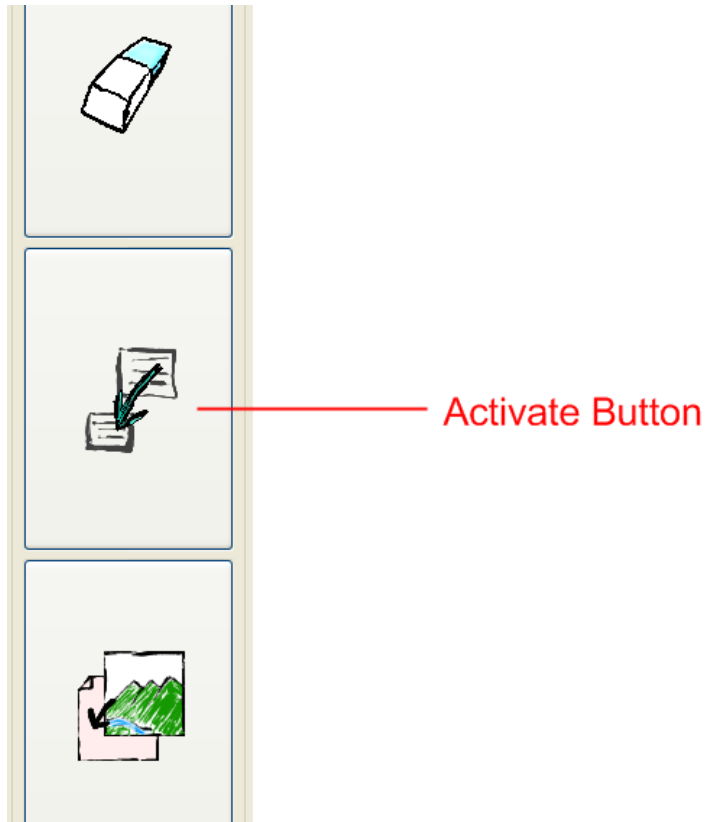


Figure 9: A screenshot of the main window detailing the Activate button.

For example, suppose a designer draws two pages. The first page is a layout of an online catalogue, complete with a title, sketched image, description and price. The second page is a drawing of the shopping cart. To add an interaction to the catalogue page, we simply select the boundary of the product with the Activate tool, and finally select the destination page. This procedure sounds complicated, but in reality is quite simple. Figure 10 shows the full series of actions required to perform this task.

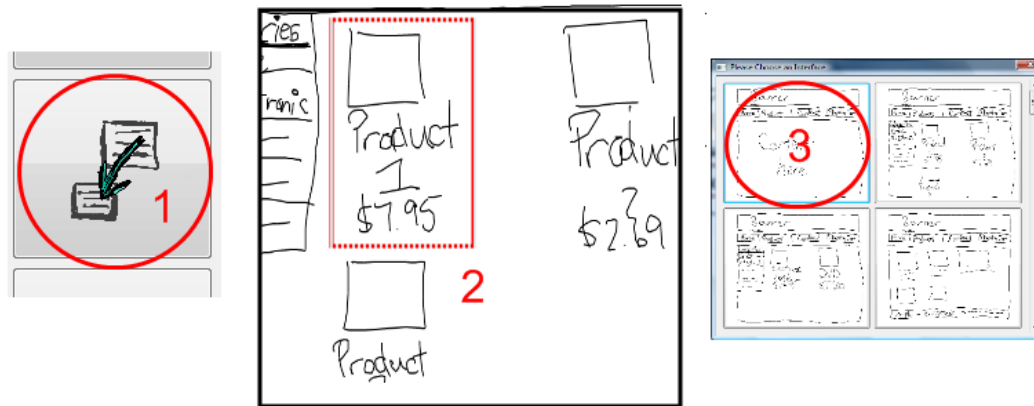


Figure 10: Sequence of events required to create an interaction.

Active Story does not attempt to recognize widgets as it was felt that this feature could become a hindrance as ActiveStory does not provide support for increasing the fidelity of a prototype. In other words, widget recognition is only useful if the designer wishes to turn a low fi prototype into a high fi prototype, a feature that flies in the face of both strict Agile principles as well as usability engineering principles.

4.2.3 Exporting the Design

Once the design is complete and all interactions have been added, the designer is ready to export the design into a prototype that is both interactive and can be evaluated in a distributed environment. The Internet is the perfect medium for providing a system to multiple, distributed individuals. It is not hard to imagine the benefits a distributed Wizard of Oz usability testing application could bring to the table. For example, usability test participants would no longer be required to be on site with the designer, thus usability testing can still happen in situations where a company can not afford to bring participants on site.

ActiveStory sports a built in web server to make the process of exporting faster and more convenient. To export the current design to the web, the designer clicks on the export button on the lower right hand side of the main window application

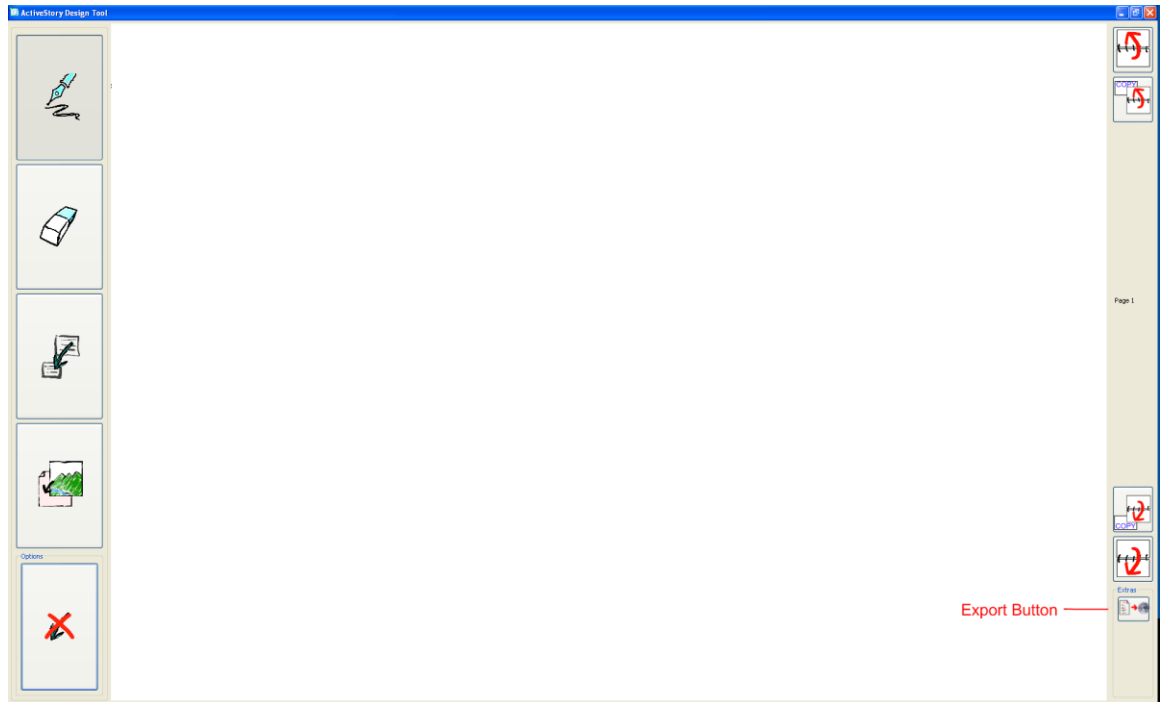


Figure 11: A screenshot of the ActiveStory Design Tool with the export button highlighted.

ActiveStory allows the designer to provide a task she/he wishes the participants to attempt to perform. This task description is the designer's way of controlling what features the usability test attempts to evaluate. The literature [6] suggests that some agile interaction designers ask the test participant(s) to attempt a specific task. An example might be "Please try to log in to the system" rather than simply "use the system." However, a task is not required in the case that an interaction designer does not wish to evaluate a specific task but rather the

prototype as a whole. The ActiveStory usability-testing tool will display the task to the test participants during the test.

ActiveStory also requires knowledge of the ip address/dns domain where the prototype will be hosted. This is due to problems AJAX applications face with Cross Site Scripting (XSS) security limitations. Most browsers require an AJAX connection to be opened to a specific domain, so the tool needs to know ahead of time what that domain will be. Once exported, the prototype can be accessed publicly (assuming the hosting computer has public access to the internet) and will begin collecting data. The process of exporting was designed to be simple and quick to perform, with a minimum of configuration required. In ActiveStory, once the prototype is built, there is literally two button clicks required to export the prototype. At that point, the designer need only provide the address of the prototype to her/his test participants.

4.3 Distributed Wizard of Oz Tool Overview

Once the designer or design team has completed a low fidelity prototype, they can conduct usability tests on the design and make revisions if necessary. As has already been discussed, it is much better to catch usability problems earlier in the development cycle, as they are generally less expensive to fix. ActiveStory allows designers to create low fidelity prototypes and run usability tests on them without requiring test participants to be collocated with the designer. As a result, travel time and expenses are reduced as study participants can access the test from any computer that is connected to the Internet.

4.3.1 Gathering Participants

Before a Wizard of Oz test can be administered, the participants must be recruited. As we have already hinted at, recruitment for a usability test with ActiveStory is much easier than a traditional test because the participants do not need to be brought in. All a participant needs to start testing a project is knowledge of the web address where the test is deployed.

4.3.2 Evaluating the Prototype

The start page describes the process involved with the test, the task that the participant is to accomplish and basic directions to providing feedback to the designer. Figure 12 provides a screen shot of what the participant is first shown when they access the ActiveStory Wizard of Oz system.

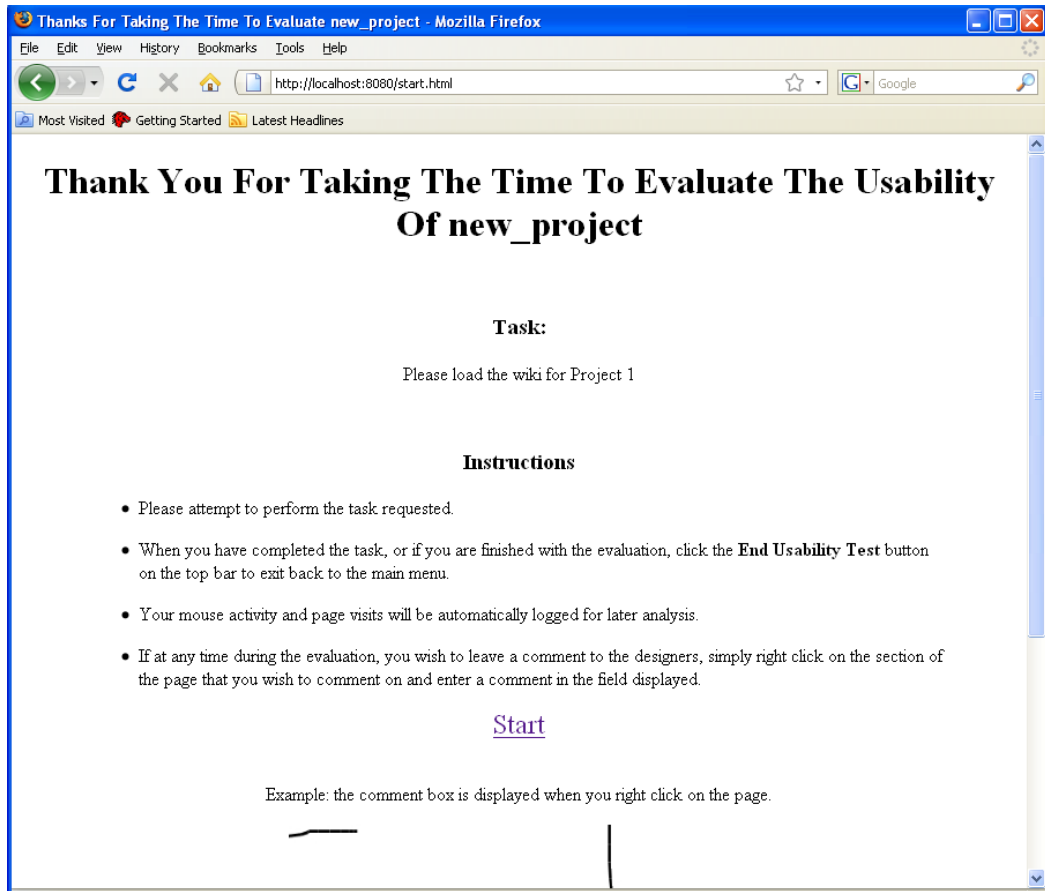


Figure 12: The ActiveStory Wizard of Oz tool initial page.

When the participant decides to actually test the usability of the application, she/he simply clicks the “Run Usability Test” link on the start page and the wire frame application is started. The participant then attempts to complete the task using the low fidelity application. At any time, the participant may leave a comment to the designer, by right clicking on the interface and entering some text in the textbox provided. ActiveStory will then attach the comment to the page and remember the coordinates so that the comment can be

rendered later in the proper context on the page. Figure 13 provides an example of the comment box.

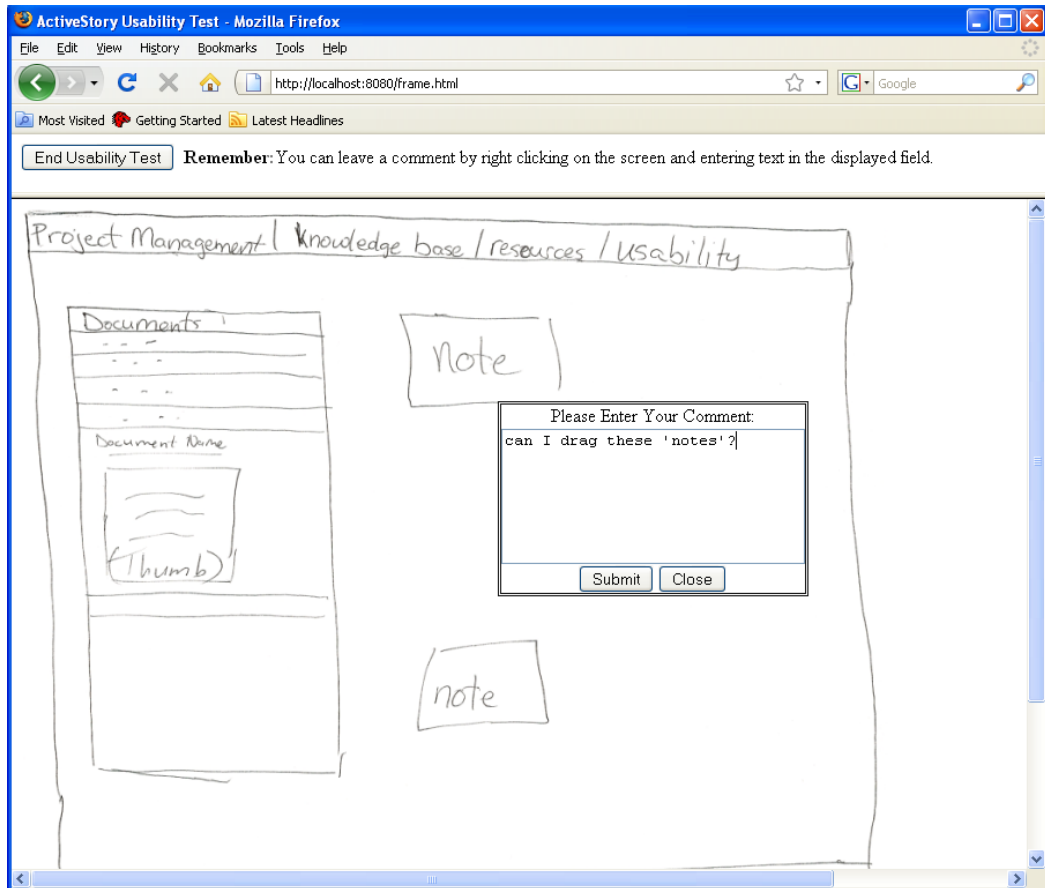


Figure 13: Entering comments in ActiveStory.

While the test is under way, the ActiveStory server is constantly collecting data about application usage. Specifically, ActiveStory stores all mouse activity on the part of the test participants as well as page timeframes (how long was a participant viewing a page before moving on to another page).

4.4 Reviewing the Usability Data

Once all the participants have used the application, the designer can begin analyzing the design based on the feedback provided as well as the data collected

by the ActiveStory server. As previously discussed, ActiveStory collects three major categories of data: mouse data, page time data, and comments. I will now discuss how a designer can interpret each of these categories assisted by ActiveStory.

4.4.1 Mouse Trails

Firstly, lets look at mouse data. ActiveStory allows designers to get a glimpse of how a user is using a mouse to complete the required task using the wire frame application. ActiveStory generates an image for each interface and participant that show where the participant's mouse was located at any given time. Figure 14 shows a snapshot of a mouse trail generated by ActiveStory. This information can be useful in discovering usability problems as labels that appear to be functional (eg: a label that tricks users into thinking something will happen if they mouse over or click on it).

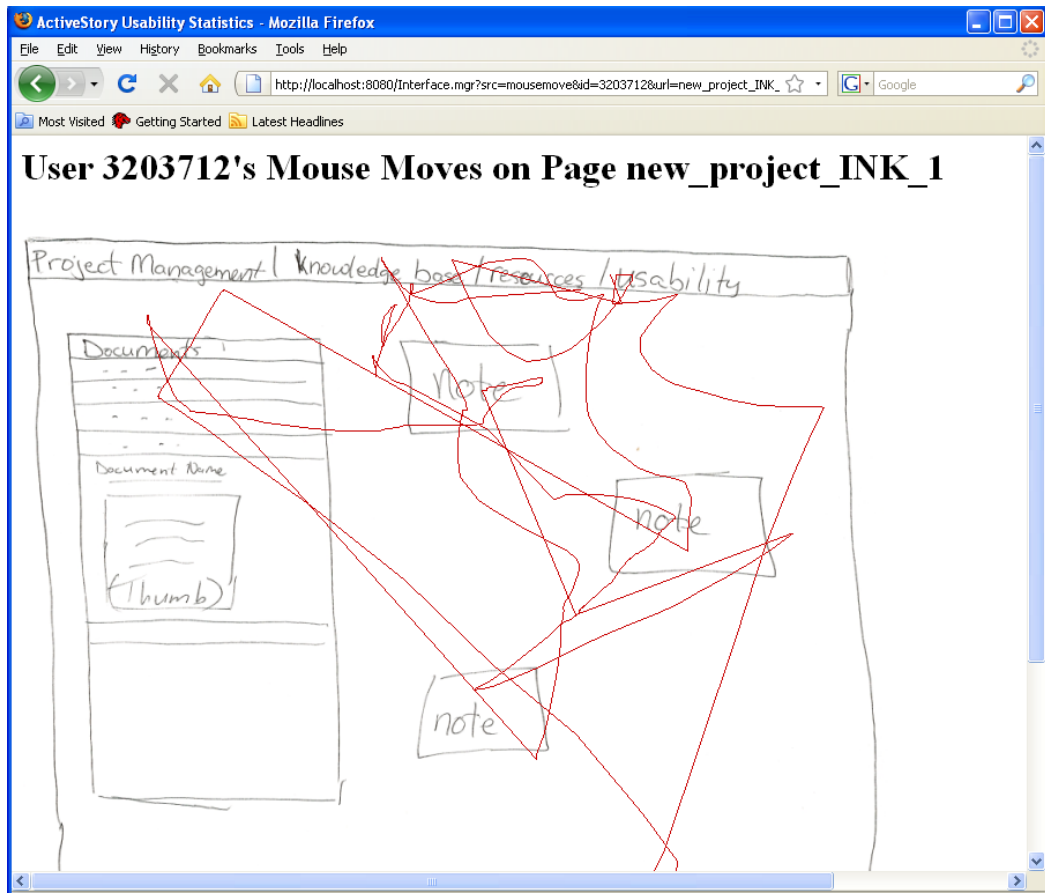


Figure 14: A mouse trail collected by ActiveStory.

If there is a consistent mouse trail for every user spending time over an element that is not a button or link, there may be a problem.

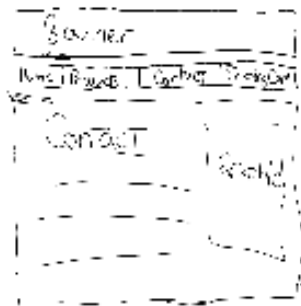
4.4.2 Page Durations

ActiveStory also allows designers to see how long users are spending on specific pages. For instance, suppose a web site prototype has five pages, a home page, a contact page, a products page, a shopping cart page and a checkout page. If the users are consistently spending 30 seconds on the shopping cart page, where no actions are really required, that might be a sign that the “**continue shopping**”

button is badly placed. ActiveStory presents the page time data in tabular form, with a table for each page in the application under test. The first line of each table provides an average time frame, and every following line provides the time frame for each visit to the page (including a unique number that represents each participant in the usability test).



1. 2.49 Seconds
2. 1.88 Seconds
3. 1.55 Seconds
4. 0.59 Seconds
5. **Average Time: 1.63 Seconds**



1. 0.59 Seconds
2. 1.56 Seconds
3. 0.78 Seconds
4. 0.35 Seconds
5. **Average Time: 0.82 Seconds**

Figure 15: Page durations in ActiveStory.

Figure 15 shows an example of what this table might look like.

4.4.3 Participant Comments

Finally, ActiveStory allows designers to see comments in the context in which they were submitted. When a participant submits a comment during the test, she/he does so by right clicking on the interface, and entering text in the text box provided. For example, suppose a participant is confused by the text in a title, she/he can right click on the title and enter a comment explaining the problem. When the designer analyses the data, the comment will be superimposed over the title and thus the designer knows that the comment has something to do with the title (even if the comment does not specifically mention the title). Figure 16 shows an example of a comment superimposed over the original page.

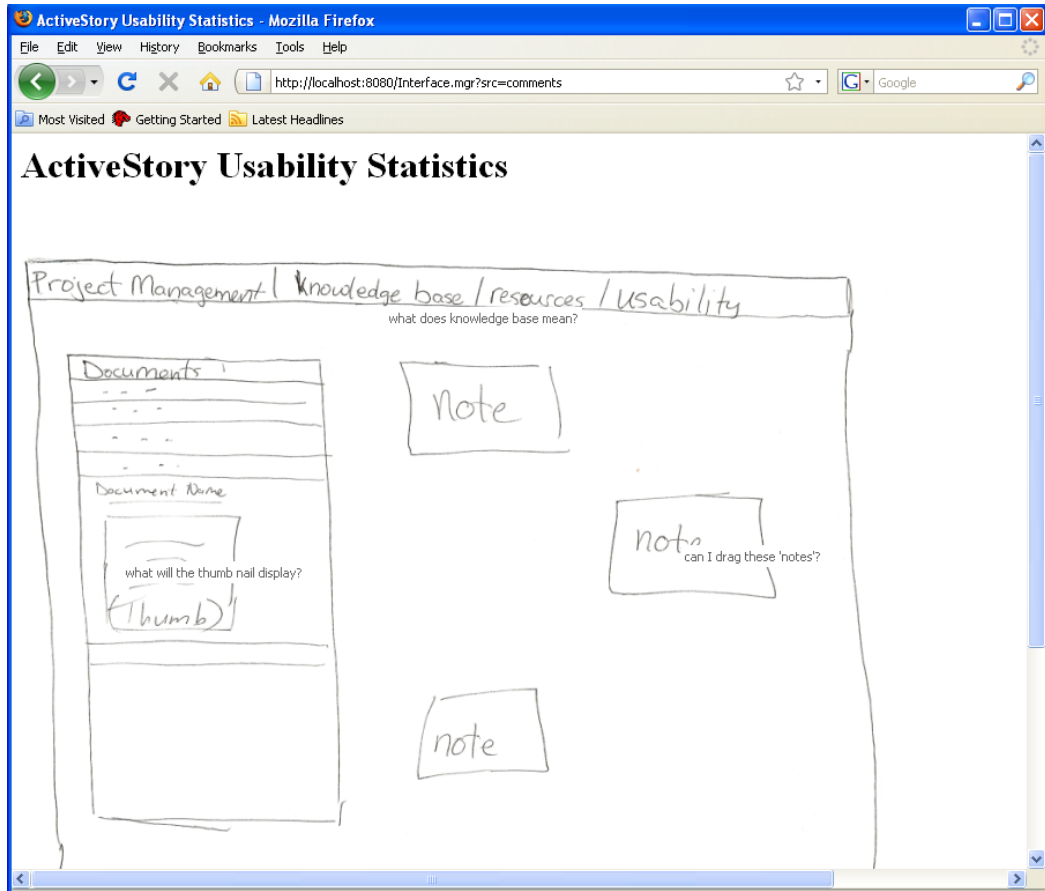


Figure 16: A comment in ActiveStory.

It is important to note that while a comment is being entered, the page timer is stopped so that the recorded time spent on that page is not artificially high.

4.5 Implementation Details

The following section describes the implementation details of ActiveStory. It is designed to provide the reader the basics of how ActiveStory is put together and will also serve to outline the cause of some of the tool limitations presented later in the thesis.

4.5.1 Design Tool

The ActiveStory design tool was written in Java/SWT. The decision to use Java and SWT was based off Java's well known cross platform compatibility. In a world where OS diversity is expanding, it simply did not make sense to use a framework that is limited to one platform (like .NET). ActiveStory persists its project data in the form of XML. Every project in ActiveStory is stored as a separate XML file. The choice to use XML was based in the spirit of open access to data.

The logical data model of the application is very similar to the flipchart metaphor used as the basis of the drawing application. The root object is a flipchart object, which in turn consists of one to many flipchart page objects. Each of these flipchart pages contains an image and a set of active regions that represent the outbound links from the page. The data structure of these active regions is simply a bounding box and the filename of the image the link (instead of a reference to a flipchart page). It was simply easier to convert to html using a reference to an image (whose filename is not changed in the conversion process) than to dereference an image from the flipchart. (Figure 17 shows a class diagram of the model of ActiveStory).

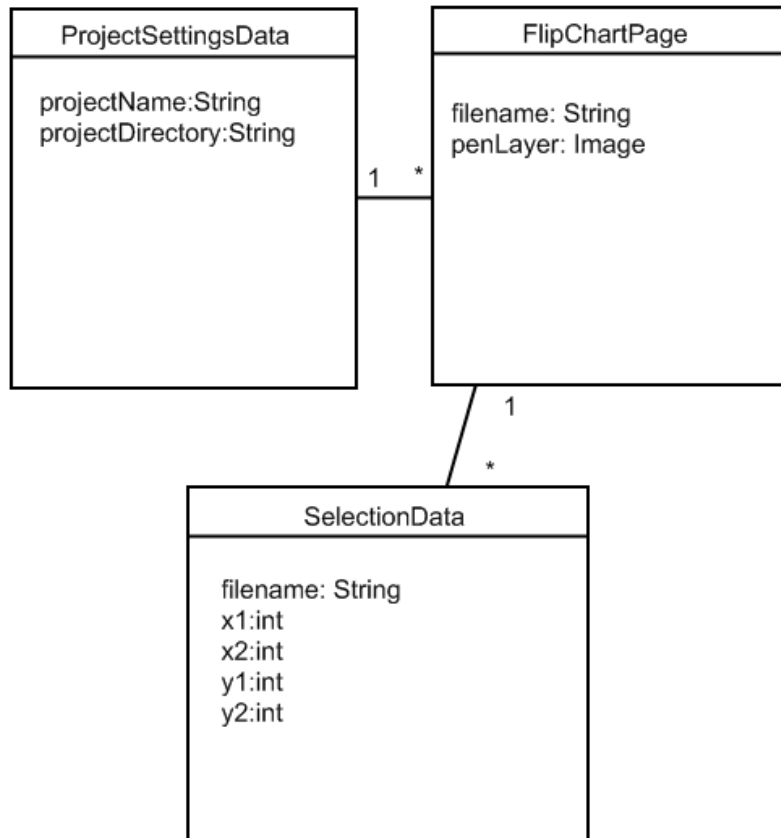


Figure 17: The model of ActiveStory.

The pen and paper metaphor is of central importance to ActiveStory, so it was important to allow for direct tablet input. It was decided early on that, as tablet drivers control the mouse cursor, an acceptable solution to the direct input problem would be to simply capture mouse events and update the current flipchart page image with the data. To accommodate this, ActiveStory streams mouse data directly to the current flipchart page object, which is responsible for updating its internal image representation.

An alternative to using a tablet is simply importing an image directly into the current flipchart page. If this option is utilized, ActiveStory simply replaces the old image with a copy of the imported image, and then reloads the internal image in memory.

ActiveStory stores image data as an image in the project directory as well as an absolute path in the project XML file. This solution has caused some problems. As the paths are stored absolutely, the project is not very portable. As a result, if the project is moved, the reference to the images is easily lost. Figure [[XX]] details the typical project file structure and demonstrates the problem described.

```
- <project>
  <project-name>new_project</project-name>
  - <flipchart>
    - <page>
      - <links>
        <link x1="410" y1="504" x2="720" y2="726" href="C:\ActiveStory\demo\new_project_INK_2.bmp" />
        <link x1="982" y1="310" x2="1290" y2="547" href="C:\ActiveStory\demo\new_project_INK_3.bmp" />
      </links>
      <filename>C:\ActiveStory\demo\new_project_INK_1.bmp</filename>
    </page>
    - <page>
      <links />
      <filename>C:\ActiveStory\demo\new_project_INK_2.bmp</filename>
    </page>
    - <page>
      <links />
      <filename>C:\ActiveStory\demo\new_project_INK_3.bmp</filename>
    </page>
  </flipchart>
</project>
```

Figure 18: Typical file structure showing an imported file link.

Navigation inside the project flipchart is accomplished via the right hand side navigation buttons. When a navigation button is activated, the reference to the current flipchart page is updated accordingly. At the foundation of the

ActiveStory design tool lies the flipchart object. References to the specific current page in the flipchart are used to manipulate the appropriate internal image. Navigation is performed by maintaining a current page index into the flipchart object.

4.5.2 Wizard of Oz Usability Test Tool

The Wizard of Oz usability testing tool is primarily written in Java, JavaScript and HTML. The exported prototype is a series of HTML files that include the images created or imported into the design tool. As soon as a page is loaded, a JavaScript timer is started. This timer is used to track the duration of a page visit. The active regions are DIV HTML elements, which are bound to the appropriate size and laid over the background image. Each of these DIV regions have JavaScript mouse click listeners registered to them. When a region is clicked, an AJAX request is made to the ActiveStory built in web server which logs the duration of time the user was at the page (from the duration timer) and navigates to the appropriate new page.

Similarly, mouse move events are captured by registering a mouse move listener to the background image. Any time the mouse is moved, an AJAX request is sent to the server containing: the mouse X,Y location and the page name. The server then logs the information in a file with the session id (so that separate mouse trails can be maintained.)

Finally, user comments are posted by opening a text field over the location of a right mouse button click. Once the comment text is entered, an AJAX

request sends the comment to the server along with the X and Y location so that the comment can be reconstructed in context.

The web component of the system is relatively small compared to the design tool, but the online usability-testing tool is really what gives ActiveStory the edge over other simple prototyping tools. ActiveStory is currently the only low fidelity prototyping tool that can automatically collect usability statistics via the Internet from many distributed test participants.

4.5.3 Integration via the ActiveStory Exporter

The bridge component between the ActiveStory Design tool and the ActiveStory Wizard of Oz testing tool is the exporter. The exporter essentially creates the Wizard of Oz testing tool on the fly. The exporter is a sub component of the design tool and is written in Java. Inside the inner workings of the Wizard of Oz tool lie two basic categories of sub components: pre-compiled components and dynamically created ones. The pre-compiled components include the servlets for handling the AJAX requests and some of the standard html pages. The majority of the components, on the other hand, need to be created when the prototype is exported. This is because as so much of the behavior of the prototype is different for every prototype, it simply did not make sense to try to abstract it. Figure 19 demonstrates the directory structure of the exported prototype and shows which components are dynamic and which are static.

css folder
client.js
frame.html
frame_top.html
help.html
index.html
link.js

PROJECT_HTML_FILES
PROJECT_HTML_FILE_IMAGES

remoting.js
screen1.jpg
start.html

Static Resource Dynamic Resource

Figure 19: ActiveStory prototype directory structure.

As the prototype is generated at export time, it is fairly easy to write different exporters for the design tool. It would be useful for the design tool to be able to export for different web frameworks, ex: J2EE or PHP. To take further advantage of this fact, I structured the exporter as an abstract factory. This allows for different exporters to be utilized in the same fashion. It should be noted however, that at the present time, there is only one exporter in ActiveStory, (the exporter for the built in web server).

4.6 Summary

We described the tool ActiveStory. ActiveStory supports all of the objective requirements we outlined in Section 2.3. It preserves the pen and paper metaphor, it allows for feedback from non-collocated usability tests and it produces interactive prototypes. More information regarding the ActiveStory's fulfillment of the objective requirements laid out previously will be discussed in the next chapter. To evaluate if the tool supports the more subjective requirements, a user evaluation is required which is described in Chapter 6.

Chapter Five: Tool Analysis

5.1 State of the Art

In Chapter 3, we laid out several tool requirements, gathered from both a web survey and qualitative interviews. Specifically, we discovered that Agile – Usability practitioners are interested in a tool that is easy to use, preserves the pen and paper metaphor, is fast for creating designs and export them, allows for feedback from usability test participants not collocated with the designer, allows for interactive prototypes, and finally the exported prototype must collect data while the test is being administered. Of these requirements, four are objective in nature.

Chapter 2 discussed some tools that have been used or at least could be used by interaction designers in Agile teams. Table 1 shows a comparison between ActiveStory and these other tools. The table is divided into two parts, the top section contains prototype design tools, and the bottom section contains usability testing tools. The tools are rated in accordance with how well they fulfill a particular requirement. They can be rated as either fully meets requirement (F), partially meets requirement (P) or does not meet requirement (N).

The requirement that a pen and paper metaphor be preserved is considered fulfilled if

- a) A tablet interface is supported or

b) There is a method for importing images initially drawn with pen and paper.

The requirement that a tool must support usability testing with participants who may not be co-located with the design team is considered fulfilled if an offsite participant can test the prototype without requiring additional software that they would most likely not already have. The requirement is considered partially fulfilled if an offsite participant is able to evaluate the prototype with access to additional software.

The third requirement, a tool must produce interactive prototypes is considered fulfilled if the tool in question is fully interactive. By fully interactive I refer to not just state change interactions but also widget interactions. In other words, can a participant enter text into a text box or are they simply presented with the image of a text box. State change interactions partially fulfill this requirement.

Finally, the requirement that a tool must collect data in a distributed manner is considered fulfilled if the tool in question is capable of collecting usability data from multiple remote participants simultaneously. The requirement is partially fulfilled if the tool is capable of collecting data remotely, except only for a single participant at a time.

Table 1: A comparison of the related tools to tool requirements.

Requirements / Tools	Pen and Paper Metaphore	Non Co-located Participants	Can Produce Interactive Prototypes	Distributed Data Collection

Pen and Paper Prototypes	F	N	N	N
SILK	F	P	F	N
DENIM	F	P	F	N
Serena Composer	N	P	F	N
MS PowerPoint	N	F	P	N
MS Visio	N	F	P	N
Intuiect	N	F	F	N
Axure RP	N	F	F	N
Basic Wizard of Oz	N	N	N	P
Oz Lab	N	P	F	P
Neimo	N	P	N	N
Morae	N	P	N	N
ActiveStory	F	F	P	F

As can be seen from Table 1, ActiveStory fulfils all of the requirements, at least partially. Oz lab scores surprisingly well based on these four requirements, however, it requires each participant to be evaluated separately so the fourth requirement is not fully fulfilled, and the pen and paper metaphor is not embraced at all.

5.2 ActiveStory Supported Requirements

ActiveStory supports the four objective requirements laid out in the previous section:

- 1) Pen and paper metaphor is preserved,
- 2) support of remote test participants,
- 3) capacity for creating prototypes which are interactive, and
- 4) distributed data collection.

As the ActiveStory design tool provides support for tablets, it is considered to fulfill the first requirement. The fact that ActiveStory can also import existing pen and paper drawings only strengthens this claim.

ActiveStory easily claims the requirement of support for remote test participants without requiring additional software; the testing framework is written with HTML and JavaScript, which are available on the vast majority of personal computers.

The third requirement cannot be fully claimed as ActiveStory does not support widget recognition and thus participants are not able to interact with most types of data presentation widgets. It was decided early on that the prototypes created by ActiveStory would reflect an automated Wizard of Oz usability test and thus only simple mouse click user input is accepted. However, ActiveStory does support state change type interactions so the requirement is considered partially fulfilled.

Finally, ActiveStory is the only tool that fulfills the fourth requirement. This requirement is considered fulfilled because ActiveStory is able to collect mouse data, participant comments and page durations from multiple participants simultaneously.

5.3 Limitations of ActiveStory

ActiveStory either completely or partially fulfills every objective requirement gathered in Chapter 3. However, there are limitations with ActiveStory. As already discussed, one of the areas that ActiveStory falls short is interactivity. Interactions in ActiveStory are simple state changes. Perhaps a method of allowing specific types of widgets (text boxes, combo boxes...) would improve ActiveStory's support of this requirement. Authentication for designers is also missing in ActiveStory. Intuitively the issue of security, although it was never

brought up as a requirement by any of the initial requirements survey participants, needs to be addressed before the tool could be used in sensitive industrial projects where new designs shouldn't be leaked to the public early. As ActiveStory is really a proof of concept tool, there was little demand for usability data privacy, however, if the tool were to be used in industry, there would have to be authentication mechanisms in place to protect usability data from being viewed by the wrong people.

5.4 Summary

In summary, ActiveStory was compared with the tools presented in Chapter 2. The comparison was based on the requirements presented in Chapter 3. Of the requirements gathered, only four are objective in nature. ActiveStory is the only tool presented, which supports all the requirements (at least partially). ActiveStory completely fulfills the requirements that a tool be using a pen and paper metaphor, support remote test participants, and collects remote usability data automatically. ActiveStory partially fulfills the interactivity tool requirement.

Chapter Six: Qualitative Evaluation

The previous chapter discussed how ActiveStory compared with several other related tools based on objective requirements that were gathered via a web survey and some qualitative interviews. The rest of the tool requirements, on the other hand, are not objective and as a result require an evaluation to determine if ActiveStory fulfills those requirements. To assist in this purpose, a pilot study was conducted to determine if there is a chance that ActiveStory fulfills these subjective requirements.

6.1 Objectives

The objectives of this pilot study are as follows:

1. To evaluate the ease of use involved with using ActiveStory
2. To evaluate whether or not ActiveStory is similar to pen and paper prototyping.
3. To evaluate the efficiency of ActiveStory, from a prototype creation and testing perspective.
4. To evaluate the flexibility of ActiveStory. In other words, to determine if ActiveStory assisted in performing existing processes instead of requiring those processes to change in order to use ActiveStory.

Thus, the research questions that this study was attempting to answer for are as follows:

1. Is ActiveStory easy to use?

2. Does ActiveStory support prototyping with a simple pen and paper metaphor?
3. Is it fast to create and export prototypes using ActiveStory?
4. Is ActiveStory flexible?

In addition to these core objectives, the pilot study was also aimed at uncovering shortcomings in the tool. These findings could then be applied back to the tool to improve the accuracy of feedback in later evaluations.

6.2 Study Methodology

Ethics approval for this study was obtained from the University of Calgary (Appendix 1).

The study was conducted with the assistance of seven undergraduate students at the University of Calgary, who acted as participants. The students were enrolled in a Web Based Systems course and they were divided into two development teams, with each team consisting of three or four members. Each team was assigned a final term project, the implementation and design details were left up to the teams' discretion. The projects lasted a little over three months each in duration.

The course required the use of agile development methodologies, however, the students were given some liberties with regards to which practices they chose to follow. Their final grades did however reflect the amount of agility the specific teams displayed during the course of development. In addition to agile development, the teams were also encouraged to pay attention to the user

experience of their respective projects. Their final grade did not specifically rely on a good user experience, but they were told that a “wow” factor certainly couldn’t hurt. To assist the students in designing for user experience, an hour and a half presentation was given which covered basic usability concepts as well as a description of how agile – interaction design was being applied in industry, based predominantly on the work of Jeff Patton [4]. As a result, both development teams practiced Agile Methodologies and Interaction design approaches.

The projects themselves consisted of three milestone releases and a final presentation / demo. At the end of each release, the students were expected to give the evaluators a brief demonstration of the work that had been completed. It is important to note that code was not looked at during these presentations and thus the students needed to create working software in order for progress to be seen as being made. For the first release, the students were encouraged to use pen and paper prototypes to design and evaluate their user interfaces. Both groups presented these pen and paper prototypes to the evaluators as part of the milestone release presentation. For the second release, ActiveStory was introduced to the students and they were encouraged to use the tool for future user interaction design. Throughout the development process, the researcher was available to the students to provide guidance with usability decisions and to answer questions regarding the use of ActiveStory. At the completion of the project, the members of the development teams were interviewed and the recordings were transcribed for analysis.

6.3 Participants

The participants of the pilot study were students taking a senior level web based systems course at the University of Calgary. Based on the data gathered in the follow-up interviews, it can be safely said that all of the students were familiar with basic programming and software engineering skills. All of the students were exposed to Agile Methods for the course and were instructed to make use of them during the development of their term projects.

It is more difficult to determine how many of the students had previous experience with Usability engineering or the precepts of good usability. This lack of data is due to a research oversight during the interview process. However, the students were given a 75-minute lecture about usability during the course and were constantly guided by the teaching assistants in good usability practices. Further, it seems that in industry (at least in agile teams) that under half of those responsible for UI design consider themselves experts (based on the results of the web survey discussed in Chapter 3). It cannot be said whether or not the skill level of a non expert UI designer in industry is comparable to a student who has been trained in simple usability engineering.

6.4 Development of Projects

The procedure being used by the students is important to the validity of the projects used in the evaluation of ActiveStory. Both the projects in the study were term projects for a web based systems course at the University of Calgary. Both projects used in the evaluation of ActiveStory were non trivial enterprise

web systems. Group 1's project was a social networking website which allowed for creation/modification/deletion of groups and group members and allowed group members to attach audio artifacts to their group. Registered users were also able to leave comments on both the groups and audio files. Group 2 created an online research collaboration tool which allowed users to share information regarding research being conducted around the university. The nature of the projects is important to the validity of this study as it is important to demonstrate that the projects were not trivial in nature.

The process used by the student design teams to create the prototypes is also important to clarify. The students followed a process similar to the industry standard used by agile user interaction designers [46]. Students followed Feature Driven Development (FDD) for the duration of the projects. Students began by creating a vision of the project and then began nailing down the details of each feature during the appropriate iteration. As a result, the user interactions for the features were designed in detail at the beginning of their iterations. This is significant because most agile interaction designers also follow this process, thus the size of the low fidelity prototypes, while probably smaller than an industrial prototype, is still comparable.

6.5 Discussion

The objective of this evaluation was to evaluate the usefulness of ActiveStory as a tool to support agile interaction designers in a situation where attention to usability might have otherwise been ignored. Such situations include small companies that cannot afford to bring in external usability testing

participants. Six research questions were posed to the ActiveStory evaluation participants.

1. Did the participant find the tool was easy to use?
2. Did the participants find that the ActiveStory design tool metaphored Pen and paper?
3. Did the participants find that it was fast to create and export prototypes in ActiveStory?
4. Did the participants find that ActiveStory was flexible in terms of design process flexibility?
5. Did the participants find that the data collected by the ActiveStory Wizard of Oz tool was useful for uncovering usability problems?
6. Did the participants find that prototyping with ActiveStory is better than using pen and paper on its own. The final subsection of the discussion deals with just that issue.











The following sub sections describe the results of the research questions in more detail.

6.5.1 Ease of Use

The first research question seeks to determine whether or not ActiveStory fulfils the requirement that a low fidelity prototyping tool must be easy to use. In order to gather evidence for determining if ActiveStory fulfils this requirement,

the question “Did you find that ActiveStory was easy to use” was posed to the study participants in the follow up interview. The verbal responses of the participants were analyzed and categorized into either: the participant agreed the ActiveStory was easy to use, the participant somewhat agreed that ActiveStory was easy to use or the participant disagreed and considered ActiveStory difficult to use. Table 2 shows the result of this evaluation.

Table 2: Evaluation of direct verbal responses to the question "Did you find ActiveStory easy to use?"

P1	P2	P3	P4	P5	P6	P7
						
Agree 		Agree, but had issues 		Disagree 		

However, there are many facets to usability. In order to better understand the usability of ActiveStory, Jakob Nielsen definition of usability was further explored. Nielsen suggests that usability is comprised of several facets, specifically a usable tool must be: easy to learn, efficient, easy to remember, cause few errors, and be subjectively pleasing [38]. Most of the participants (except P3 and P5) reported that overall they found the tool easy to use, but they found themselves hindered by one or two usability issues. In terms of learnability P5 actually suggested that the tool was learnable

“I was actually playing around, just Ok, what does this do, what does that do. I was like figuring out how to make it flip between pages and stuff” – P5

P5 was not the only participant to describe how the use of the tool was learned.

P7 described his experiences as:

*“...I had an issue with the tablet pc. I can't see the tool tip when I am [using the pen] like this...So at the beginning I was a bit WOW. Which button does this? The icon on the eraser is obvious but this one and this one [pointing to **import image** and **create active region** buttons] ... are a bit difficult without the tooltip.” – P7*

“Something that would be to actually label things. A first time user, you know, they don't quite understand.” –P2

Clearly, P7 found that the use of icons with the buttons in ActiveStory to be problematic. But P7 finishes by saying “but it was quite easy to use”. Other than these three participants, no other mention was made that there were issues learning how to use the tool. The bigger category of usability problems seems to related to limiting errors. A number of participants mentioned that ActiveStory's lack of an undo feature was a major hindrance.

“You would draw something and you would have to erase it because you know you message and drew the line wrong and then you get this

huge eraser that doesn't allow for much precision. There wasn't an undo so you couldn't just go back.” - P3

“I found there was a little bit of a problem with undo... If you did something wrong it is hard to get back” – P2

P3 touched on another error causing feature of ActiveStory's design tool, a single sized eraser that had a tendency to clobber a sketched interface. P7 also echoed this sentiment, “I would say to use the eraser tool [is difficult], I would say it is a bit too big”. Another cause of error turned out to be the flip chart metaphor. Several test participants mentioned that they didn't feel they understood how to navigate using the flip chart, specifically when to use the copy buttons and when to use the page up / page down buttons. Figure XX displays the flip chart controls. *“Sometimes when I tried to go to the next slide, I was clicking on Copy. They're a bit too similar.”* Comments P4.

On the more positive side, the online wizard of Oz tool received much better usability praise from at least one participant *“The online wizard of oz part of the tool was extremely easy to use”* (P3).











Given the feedback received regarding the ease of use of ActiveStory, it can be easy to read the results too negatively. It should be noted that all of the usability issues reported were in the design tool, and not in the distributed wizard of oz tool. In fact, several of the reported usability issues were a direct result of the interpretation of the tool requirements. For example, the requirement that a low fidelity prototyping tool should metaphor pen and paper leaves some room

for interpretation. Should the tool provide an undo feature? True pen and paper would not. However, the use of a metaphor should not be an excuse for poor usability. However, I believe the metaphor would still be intact if an undo feature were to be provided. These usability issues should not, however, be shrugged off. It may simply be that the design of the tool needs to be improved to mitigate some of these problems.

6.5.2 Pen and Paper Metaphor

The second research question that the pilot study aimed to answer was whether or not ActiveStory used a transparent pen and paper metaphor. This question needed to be explored in a more subjective manner because it is possible that a tool designed to meet a specific metaphor may not actually present that metaphor in a very transparent manner. In the case of ActiveStory, the participants of the pilot study were asked the question “did you find the tool metaphored what you did with pen and paper?” Table 3 provides a quick glance at how the participants answered that question.

Table 3: Did the participants feel ActiveStory metaphored pen and paper.

P1	P2	P3	P4	P5	P6	P7
						
Agree 		Agree, but had issues 		Disagree 		

Interestingly, the participants were much less undecided with their responses to this question. It was either “yes we totally agree” or “no, we don’t agree”, there was no middle ground. What is even more interesting is that the justification for disagreeing with the statement “ActiveStory metaphors prototyping with pen and paper” were predominantly due to the usability issues already discussed.

“I decided to go with [Microsoft] Paint [instead of drawing using the ActiveStory design tool]. One of the things I think would be helpful would be some kind of template you could create. Cuz for most of your pages its usually you have the skeleton... I found I had to replicate most of those [templates] in Paint, just to get the same thing” –P2

P2 felt that the pen and paper metaphor broke down because he was not able to use the design tool provided by ActiveStory and as a result decided to use Microsoft Paint. It is not clear from the interview whether P2 found that the wizard of oz tool to be reflecting a metaphor for running a usability test with sheets of paper.

Participant P3 also decided to disagree with the research question. Again, his justification was based mostly in usability errors.

“We ended up having to take it a different way because you draw something and then you mess it up, now I’m going to have to redraw all these lines. It [the drawing tool] left too much to be desired and we basically had to let it go.” –P3

Due to the usability problems, P2 and P3 decided not to use the tool and as a result felt that the tool did not reflect working with pen and paper. However, despite these two negative cases, the rest of the participants responded extremely favorably toward the transparency of the metaphor.

“Yes [I agree that ActiveStory metaphors pen and paper] because it does all the organization for you. You don’t have to worry about what comes next because you already predetermined that” – P1

P1 is referring to the ActiveStory wizard of oz tool, rather than the design tool. Paraphrased, using the bigger context of the interview, P1 is suggesting that the manner of creating and executing usability tests strongly metaphors pen and paper wizard of oz usability tests.

All of the other participants replied in a similar manner:

“that [draw with tablet in a manner similar to drawing with pen and paper] was my first inclination of what I wanted to do. I wanted to draw with the tablet. The paper prototypes were kind of like the backup.” – P5

Clearly P5 attempted to use the tablet drawing interface first because it felt like the natural way to both prototype and use the tool. He eventually gave up on the tablet due to usability issues associated with using a side tablet (not related to ActiveStory).

“Ya it was quite the same. Ya, I would say it was even easier because you can erase things...It [ActiveStory] is very useful because you are not lost with all your pages, because you have to just click on the next

button and switch to the next screen, you don't have to search for the right page to display. This is very good" – P7

P7 is suggesting that although ActiveStory has some benefits over pen and paper, the metaphor is still preserved. This participant believes that ActiveStory simply takes the chaos of a pen and paper metaphor and organizes it.










In all, I think it can be safely assumed that ActiveStory fulfills a pen and paper metaphor for creating low fidelity prototypes.

6.5.3 Efficiency

The question regarding efficiency is a very important question for this pilot study. If ActiveStory is not saving designers time nor effort in gathering participants then it is not fulfilling its primary purpose. This means that ActiveStory must be fast to create prototypes and export them, otherwise designers may be spending more time creating a prototype than they are saving by not using pen and paper to administer the usability test. To discover an answer to this research problem, the question “Did you find that it was fast to create and export prototypes in ActiveStory?” was posed to the participants of the pilot study. Table 4 demonstrates a quick view of the responses gathered from this question.

Table 4: the responses to the efficiency research question.

P1	P2	P3	P4	P5	P6	P7
----	----	----	----	----	----	----

				No data		
Agree 		Agree, but had issues 		Disagree 		

Here, there is a majority (5/6) that agrees that ActiveStory is efficient. The participant which had some issues are included in this count because they still found that ActiveStory was faster than using pen and paper to create and execute a usability test. P6 suggested that the initial learning curve slowed down the first attempt at a prototype too much, however he also claimed that once the learning curve was overcome it was better.

“Initially it was a bit slow, because it was the first time I was using this type of thing [a tablet pc] for prototyping. But afterwards it improved.”

– P6

The two participants who disagreed with the notion that ActiveStory is efficient were P2 and P5. P2 felt that trying to remember where all the links where pointing was difficult:

“It is more interactive for sure, using ActiveStory than paper. I thought it was a bit troublesome getting all the links together. We had so many links linking everywhere, you kind of have to remember which links go where” –P2

P5 also had some issues relating to usability. P5 issues were so severe they skewed this data point too far to be safely used. P5 was not able to export his

design to Internet Explorer 6 due to a bug in ActiveStory. For this reason he said that the process was too slow (because he spent far too much time attempting to figure out why his prototype was not working). However, his option improved when he used Firefox, however it is safer to drop his data point in this particular question.

“Just to get it to work in the browser, I had a lot of problems with it. I had a lot of problems with IE 6...Once I upgraded to Firefox, it was much smoother.” –P5

The rest of the participants had no reservations claiming that ActiveStory was more efficient than using Pen and Paper.

“Ya, I would say it was as fast as drawing them on paper. And then it is faster when you want to perform the test because you don’t have to search for pages.” –P7

“I thought it was better than with paper. Just that there was a lot more freedom with what I could work with then on paper.” –P4

Participant P3 added to this sentiment by suggesting that the time required to create a prototype in ActiveStory is about the time that would be required to run one usability test, so if more than one test is planned, ActiveStory is much better.

“I think realistically it took probably just as long to link it as the first time you actually present it. So if you have to do it to more than one person, you have a huge time saver.” –P3











In all, it can be safely said that once the usability issues with ActiveStory are overcome, the tool will be more efficient than pen and paper prototyping.

6.5.4 Flexibility

The fourth research question that the pilot study aimed to answer related to the flexibility of the ActiveStory tool. The participants were asked “Did you find that ActiveStory was flexible”. This first question was then immediately explained in more detail to make sure the participants answered the correct question. The participants were asked to answer the question assuming flexibility referred to process flexibility. Essentially the participants were asked “did the tool adapt itself to your process or did you need to adapt the process to meet the tool?”

Table 5 shows the participant responses.

Table 5: Participant responses for the flexibility research question.

P1	P2	P3	P4	P5	P6	P7
						
Agree 		Agree, but had issues 		Disagree 		

Once again, some usability issues tainted the results. However, the participants who agreed that ActiveStory was flexible agreed strongly.

“Ya, it allows us to be very flexible in how we actually do it, I mean we could scan in bitmap, do it in photoshop as a bitmap. We’ve got

numerous ways we could have created the images and then [the tool] was flexible enough to allow us to choose.” –P3

P7 simply answered “exactly” to the question “so you did not have to radically change your process to use the tool?”

The participants who had usability issues interfering with their process still agreed that overall the process left intact.

“I find that I had to do it in Paint to get it into the tool, but other than that I think it was ok” –P2

Interestingly enough, one of the participants (P1) claimed that the tool changed the process by forcing the team to place more focus on usability engineering. As this is a change for the good I assume that P1 would agree that ActiveStory is not a hindrance.

P7 states that the process was not really changed, but that a usability issue with removing pages caused some havoc and forced him to change how he managed pages. He admits that if the usability issue did not exist, the process would not have been altered.

In summary, only one participant outright disagreed and claimed that ActiveStory caused him to change his process to use the tool. The other participants either did not have to change their processes at all or encountered some usability issues and had to make minor process changes as a result.







6.5.5 Usefulness of the Data Gathered

The final research question that the pilot study aim to answer was whether or not the usability data collected by the online wizard of oz tool was useful. This was a

harder question for the participants to answer as many of them did not have much experience with usability prior to the course. However, many of the participants were able to uncover usability errors as a result of using ActiveStory. For example, one participant discovered that his user interaction require a lot of mouse movement to complete. A more detailed look at some specific usability problems the participants uncovered is discussed later in this section.

The final research question was phrased to the participants similar to “did you find that the usability data gathered by ActiveStory was useful for uncovering usability problems?” Table 6 shows the responses to this question.

Table 6: Responses to the question regarding the usefulness of the data.

P1	P2	P3	P4	P5	P6	P7
No data		No data	No data		No data	
Agree 		Agree, but had issues 		Disagree 		

Unfortunately, several of the pilot study participants did not actually attempt to analyze any of the data collected during their usability studies. If this was the case, the participants were then asked the follow up question “If you had collected some usability data, would you have found that data useful.” In the end, these responses were not used, as they would taint the data set too much. However, it should be noted that none of the hypothetical responses were negative. In the case of P1 and P3, ActiveStory was used during a demo of their system. The

instructor of the course tested the application and the data collected may have been useful, but neither P1 nor P3 analyzed the data as they had already witnessed the usability errors, as all of the group members were present at the demo. In the case of P4 and P6, the design tool was used, the designer tested the prototype but the collected data was never analyzed. The rest of this section will discuss the findings of the remaining four participants in more detail.

Of the three participants who did actually analyze some data gathered by ActiveStory, all found that the mouse trail data was very useful.

“I found it [the data collected] quite useful...I brought up those pictures and saw those lines tracing where the mouse is, and where most of the activity is. I have it in the back of my head that I wanted to [design] so the user won't have to move the mouse back and forth.

That picture helped” - P2

Another participant (P5) stated, “I really like the mouse feature... I found I am catching myself on a couple of things. It is going from side to side too many times.” In all, all the participants who analyzed the data agreed that mouse trails were useful usability data.

In practice, it seemed that the commenting feature was not used as heavily as the mouse trails. However, this could be explained by the sample. As already stated, none of the participating groups attempted to use real usability participants (due to the time and scope of the school project) and thus there was never a need to look at comments.

The page durations feature was never specifically categorized as useful by any of the participants, however, one participant (P7) stated that he had used the feature and the tone of his voice certainly indicated that he found the feature useful.

Interviewer: “Did you find the view the page durations useful at all?”

P7: “Ya [the duration feature is useful], I saw the duration on each page, how long I took to click on the next button.”

In summary, the mouse trail feature was far and away the most popular usability data gathered. Surprisingly, the participants did not view page duration information as overly useful, however the data was viewed by some of them. It may be the case that real usability test participants leaving comments would have changed the responses regarding how useful user comments were.

6.5.6 ActiveStory vs Pen and Paper

The final question that the pilot study aimed to answer was whether or not prototyping with ActiveStory was better than prototyping with pen and paper. The participants of the survey all agreed that ActiveStory certainly has many added benefits to pen and paper type prototyping.

“I think ActiveStory really helped out with the workings, like how to transition. ... The interactivity and it makes it easy for the user... The user I think feels that they are in the system more, rather than having someone [fake it]. ... This way it flows a lot nicer.” –P1

Participant P6 also agreed that ActiveStory was an improvement over pen and paper prototyping. When asked “is ActiveStory an improvement over pen

and paper” he responded, “Yes, it’s a new idea of course, but it was easy to use.”

P7 responded to the same question with:

“I think that ActiveStory is a good tool for horizontal prototypes...If you started from scratch to show all the flow in your program it is a good tool. It avoids the pain of flipping the pages and everything.” – P7

Participant P5 suggested that if the tablet technology was better supported by ActiveStory he would fully agree that in no way is paper prototyping better than ActiveStory. However, as he had issues with the tablet he said:

“I just like to draw. I like to sketch. What I like about ActiveStory was the navigation. That was the best part of it... But the mouse features, the comments, it’s a good way to share if your going to send the project to someone else...It’s like ‘here what do you think of this?’ ” – P5

In summary, all of the participants felt that the tool was at least a bit of an improvement over pen and paper prototyping. It is clear that there are several usability problems that need to be addressed before the tool can really prove to be a major benefit over pen and paper prototyping.

6.6 Study Limitations

The external validity of the pilot study is limited. However, the purpose of the pilot study is really to determine if there is a **chance** that ActiveStory is a useful tool for assisting agile interaction designers in producing more usable software as well as working out issues with the tool itself. These can be addressed

before a more comprehensive study is conducted. I think the study succeeded in its task. However, as is often the case with a pilot study, a more formal empirical evaluation will be required to really demonstrate that ActiveStory fulfills its purpose. There are two major limitations with this pilot study. Firstly, the study was conducted with students. Academic environments allow researchers to have more control over the experiment, which improves the internal validity. However, students are not industry workers. Students do not have the development and design experience that designers in industry have. This may skew the data, as students are likely to use the tool differently than industrial workers would. It is usually the case that students do not put the same care and attention to design and development practices as industrial workers due to time constraints.

The second limitation of the study involves the experiment sample. Firstly, there were only seven participants in the study, which limits the statistical validity of the sample. The scope of the projects is also a limitation. In industry projects are usually much bigger than four months with three or four developers working part time. Finally, although an effort was made to distant the researcher from the participant during the course, it is often the case that students attempt to be kind to teaching assistants, even though the TA was not in a position of power over the students. This may also have tainted the results.

6.7 Independent Feedback

ActiveStory was given to some members of industry during the final phases of the research process. The version of ActiveStory was the same version

that was used in the pilot study, as there had not yet been time to implement any of the suggested changes. The purpose of this follow up study was to provide this thesis with some external validity.

Two members of industry used ActiveStory. Both participants are in the field of web design and development. One participant (P8) is a web developer with a computer science background. She has much experience in the field of web design and has been involved with many small to medium sized web projects (based on the follow up interview). The other participant (P9) is a graphic artist who is currently working for the design department of a large company. He has experience with designing web sites but not much development training or experience. He noted:

“[I] did usability testing for band websites that we built. I think we had about 15 people try out our paper prototype. There were two web designers. We didn't use any screens because it was strictly on paper, but we videotaped the entire process to analyze the usability. This method was effective but very time consuming. We basically had to interview each participant as well as have them voice their thoughts and actions as they went through the site.” – P9

In both participant cases, the tool was used on a more contrived project. The participants were told to create a project similar in scope to the types of projects they would create in their day-to-day jobs. In the case of P8, the project created was a travel blog web site. In the case of the graphic artist P9, the site

was a simple navigable site with a handful of pages in it. He then got two participants to use the tool and analyzed the data collected.

Both of the participants were then questioned, using the same questions as in the pilot study with students. In the case of P9, an interview was not possible and thus the questions were emailed to him with instructions to answer each question with a brief paragraph. P8 was interviewed face to face and the interview was then transcribed.

The feedback received from these two industrial participants was very positive. P8 identified many of the same usability problems that were earlier brought up in the pilot study. However, she commented that she “would definitely use this instead of pen and paper because it is right there and I don’t need to scan anything in and it is hand drawn.”

Both independent participants answered the first research question (was ActiveStory easy to use) positively. P8 stated: “It was very easy to use, very intuitive to just figure out where things go.” My initial expectation was that P9 would find the tool difficult to use, based off the feedback from the pilot study and the fact that he is not a computer scientist (as all the other participants had been). This was not the case. P9 found the tool to be extremely easy to use:

“After a few tests with the program I found it super easy to use. It worked almost perfectly with my Wacom pen tablet. The eraser did not work however, and some of the movements weren't very smooth, but this was not a big deal. Once I was up and running, I easily created a navigation. The interface is simple and extremely easy.” – P9

Participant P8 focused most of her feedback on the prototype design tool and thus her feedback is very similar to the feedback received from the student participants. She found that using the tool was similar to using pen and paper “ya [I think that it was a metaphor for pen and paper], the fact that you had a tablet and not a mouse was a huge thing for me”. She also noted that although pen and paper is simpler to use, the benefits of ActiveStory far out weigh pen and paper prototyping. “I would say that I would definitely use this instead of pen and paper because it is right there [already entered into the computer via tablet]. I don’t need to scan anything in and it is hand drawn and I don’t have to have all those papers with me.”

Participant P8 did not actually collect any usability statistics, but when shown a screen shot of mouse trails, she felt that this information would be very useful “definitely [useful], most people, especially if their novice computer users, will follow their mouse, will look at their mouse. And so I think it is a really good indication at where they are looking.”

Participant P9 was the only participant who actually got others to try out a design. In total, he recruited two participants for his usability study. He commented that the usability data collected was very useful.

“Yes I did. For the most part, I used the information that pertained to time spent on each page, as well as mouse tracking. Time spent on the page was really interesting and useful because it allowed me to see how much time was being spent figuring out the navigation. Since my site was strictly navigation and no content, I could see how much time

was being spent navigating, as opposed to reading etc. This information is very useful, and will be good for creating effective and efficient page navigation. People want to get to where they want to go as fast as possible without wasting time on other pages. This tool could effectively help to achieve this.”-P9

Interestingly, P9 noted that “None of my usability testers took advantage of the comments function”, which seems to be a trend noted by most of the participants that were interviewed. However, he noted “but I know this is a very good feature and could prove to be extremely helpful.”

6.8 Summary

In order to determine if there is a chance that ActiveStory fulfills the more subjective requirements, a pilot study was conducted. The results of the study were very promising. However, a few bugs and some usability issues inherent to the design decisions used to create the tool hampered the study. It was found there is a chance that ActiveStory fulfills all of the subjective requirements, specifically: ActiveStory was found to be easy to use, metaphored pen and paper, was efficient, was flexible, and collected useful usability data. The study had several limitations however, including a small sample size and the fact that the test participants were students. Additional independent feedback received from two industrial users supported the findings in the pilot study while adding external validity to the evaluation of ActiveStory.

Chapter Seven: Conclusion

The following conclusion is a summarization of the contributions of this thesis as well as a discussion of some possible future work. The research motivations are first revisited to provide a basis for the research contributions in the field of Agile – Interaction Design.

7.1 Research Motivations

In Chapter 1, several motivations for this research were presented.

Specifically the motivations are:

- 1. There is little published work regarding agile / usability tool support.** The research field of agile interaction design is a maturing topic. However, there is not very much published work that details the types of tool designers are using to perform their tasks with.
- 2. There are currently no tools that allow for distributed usability data collection.** Traditional paper based prototypes require usability test participants to be collocated with the usability designer / evaluator. This collocation is often not financially possible in small agile teams.
- 3. Once such a tool is built, there needs to be an evaluation of that tool to determine if it actually assists designers to develop more usable software.** A tool needs to be properly evaluated with study participants to determine if it meets its purpose. In this case, the tool needs to be evaluated against the requirements gathered in Chapter 3.

7.2 Research Contributions

Based on the motivations of this thesis, several contributions can be derived.

- 1. For this thesis, a web survey and qualitative interviews were conducted to gain insight into both what tools are being used by agile interaction designers, and gather tool requirements that they feel would be useful in a novel tool.** A web survey was constructed to gather an understanding of the state of the art in terms of agile interaction design and of tool requirements for a tool that supports agile interaction designers specifically. In tandem, several leading agile interaction practitioners were interviewed and the transcripts were analyzed to uncover details on how agile and interaction design are being used together.
- 2. Based on the results of the web survey and qualitative interviews, a novel tool ActiveStory was created.** A tool called ActiveStory was created to fulfill the requirements gathered in the initial web survey and qualitative interviews. The tool consists of two components: a prototype design tool and a web based wizard of oz evaluation tool.
- 3. The ActiveStory tool was evaluated using academic students with respect to the tool requirements. The tool was also compared against the requirements gathered in Chapter 3.** The first tool

requirement, that the tool be easy to use, was partially satisfied. The second tool requirement was that the tool must metaphor pen and paper prototyping. This requirement was fully satisfied by ActiveStory. The third requirement, that a tool must be efficient, was found to be true with ActiveStory. The fourth requirement stated that a tool must be flexible. This was also found to be true in ActiveStory, although there were some usability issues that tainted the data. The final requirement was that the remote data being collected by the tool should be useful in uncovering usability problems. There were some problems with the study in this aspect, but the study data that was collected pointed to ActiveStory's remote data collection being useful. Finally, it was also clear from the study that ActiveStory was an improvement over prototyping with pen and paper.

7.3 Future Work

This thesis has opened the road for much other follow up research. Firstly, it is necessary to perform a better, industry based, evaluation of ActiveStory to truly understand how agile interaction designers would use it and as a result, better understand what future work may be required to improve the tool. However, there are plenty of future tool enhancements that have been uncovered by the pilot study. First and foremost, several usability concerns must be addressed. These include, rethinking the flipchart metaphor for user interface navigation inside the design tool, providing better support for tablet technology,

and providing an UNDO functionality so that designers do not have to rely as heavily on the eraser to correct mistakes.

In terms of future research not directly related to ActiveStory but rather to the field of tool support for Agile Interaction design, there is still plenty of work required to determine if there exist other tools that can support this process. For example, ActiveStory strives to support agile interaction design in situations where it is difficult to bring in test participants. It does not attempt to replace the need for more traditional methods of usability testing. However, an interesting research topic might be to explore the effects of replacing collocated forms of usability testing with distributed forms. It might also be worthwhile exploring whether or not a tool such as ActiveStory actually streamlines the process of interaction design from within an agile context.

References

1. Principles behind the Agile Manifesto, <http://agilemanifesto.org/principles.html>
2. Constantine, L. L.: Process Agility and Software Usability: Toward Lightweight Usage-Centered Design. Information Age, August 2002.
3. Barnum, C.: Usability Testing and Research. Pearson Education, New York (2002).
4. Patton, J.: Hitting the Target: Adding Interaction Design to Agile Software Development. , In. OOPSLA 2002, pp. 1-ff, ACM Press, 2002
5. Sy, D.: Adapting Usability Investigations for Agile User-centered Design, Journal of Usability Studies, vol. 2, issue 3, pp. 112-132, ACM Press, 2002.
6. Meszaros, G., Aston, J.: Adding Usability Testing to an Agile Project, In Agile 2006, pp. 289-294, IEEE Press, 2006
7. Landay, J.A., Myers, B.A. "Interactive Sketching for the Early Stages of User Interface Design", CHI, ACM Press, Denver , 1995, pp 43-50
8. James Lin, Mark W. Newman, Jason I. Hong, James A. Landay, "DENIM: An Informal Tool for Early Stage Web Site Design." Video poster in *Extended Abstracts of Human Factors in Computing Systems: CHI 2001*, Seattle, WA, March 31-April 5, 2001, pp. 205-206.
9. Visual Studio: <http://msdn.microsoft.com/vstudio>
10. Visio Home Page- Microsoft Office Online: <http://office.microsoft.com/visio>
11. Fiedler, A., Gabsdil, M.: Supporting Progressive Refinement of Wizard-of-Oz Experiments, In Sixth International Conference on Intelligent Tutoring Systems – Workshop W6: Empirical Methods for Tutorial Dialogue Systems, 2002.

12. Ricki's Wizard of Oz Tool, <http://www.softdoc.de/woz>
13. Munteanu, C., Marian, B.: MDWOZ: A Wizard of Oz Environment for Dialog Systems Development, In 2nd International Conference on Language Resources and Evaluation, Greece (2000).
14. Li, Y., Hong, J. I., and Landay, J. A. 2004. Topiary: a tool for prototyping location-enhanced applications. In Proceedings of the 17th Annual ACM Symposium on User interface Software and Technology (Santa Fe, NM, USA, October 24 - 27, 2004). UIST '04. ACM, New York, NY, 217-226. DOI=<http://doi.acm.org/10.1145/1029632.1029671>
15. DUB – DENIM: <http://dub.washington.edu/denim/>
16. SMART Technologies: <http://smarttech.com/>
17. AXURE RP: <http://www.axure.com>
18. AXURE RP – Prototypes: <http://www.axure.com/prototypes.aspx>
19. Paul, S.: Usability Success Stories,
http://www.gowerpub.com/pdf/Usability_Success_Stories_Intro.pdf
20. User Centered Design Works: <http://www.ucdworks.org/ucdworks/opencms/>
21. Constantine, L.: What do Users Want?, In Windows Tech Journal, 1995.
22. Ferreira, J., "Interaction Design and Agile Development: A Real-World Perspective", M.S. thesis, Victoria University of Wellington, New Zealand, 2007 pp. 63
23. Landay, James A. and Brad A. Myers. "Sketching Storyboards to Illustrate Interface Behaviors." In CHI'96 Conference Companion: Human Factors in Computing Systems. 1996. <http://citeseer.ist.psu.edu/landay96sketching.html>
24. PowerPoint Home Page – Microsoft Office Online: <http://office.microsoft.com/powerpoint>

25. Kavanagh, R., Soety, J., "Prototyping With Visio", In Usability Interface, vol 7, No. 1, July 2000
26. Intuitect – Intuitect Professional: <http://www.intuitect.com/products/intuitect-professional.php>
27. Serena Dimensions Composer – Serena Software – Application Lifecycle Management – Software Development Lifecycle:
<http://www.serena.com/Products/composer/index.html>
28. IA Think: Seeking the Silver Bullet: http://www.iathink.com/2005/09/seeking_the_sil.html
29. Kelley, J.F., "An empirical methodology for writing user-friendly natural language computer applications". Proceedings of ACM SIG-CHI '83 Human Factors in Computing systems (Boston, 12-15 December 1983), New York: ACM, pp. 193-196.
30. Kelley, J.F., "An iterative design methodology for user-friendly natural language office information applications". ACM Transactions on Office Information Systems, March 1984, 2:1, pp. 26-41.
31. A Wizard of Oz framework for collecting spoken human-computer dialogs: An experiment procedure for the design and testing of natural language in-vehicle technology systems. Brian Lathrop, Hua Cheng, Fuliang Weng, Rohit Mishra, Joyce Chen, Harry Bratt, Lawrence Cavedon, Carsten Bergmann, Tess Hand-Bender, Heather Pon-Barry, Benjamin Bei, Madhuri Raya, Liz Shriberg. In the 12th International Congress on Intelligent Transportation Systems, San Francisco CA, USA.
32. OzLab: <http://www.cs.kau.se/~jsp/ozlab/>
33. Pettersson, J., "OzLab – A System Overview with an Account of Two Years Experiences", HumanIT, 2003: http://www.humanit.org/pdf/HumanIT_2003_Ch10_Pettersson.pdf

34. Coutaz, J., Salber, D., Carraux, E., and Portolan, N. 1996. NEIMO, a multiworkstation usability lab for observing and analyzing multimodal interaction. In Conference Companion on Human Factors in Computing Systems: Common Ground (Vancouver, British Columbia, Canada, April 13 - 18, 1996). M. J. Tauber, Ed. CHI '96. ACM, New York, NY, 402-403. DOI=
<http://doi.acm.org/10.1145/257089.257904>
35. Calvary, G. and Coutaz, J. 2002. CatchIt, a Development Environment for Transparent Usability Testing. In Proceedings of the First international Workshop on Task Models and Diagrams For User interface Design (July 18 - 19, 2002). C. Pribeanu and J. Vanderdonckt, Eds. INFOREC Publishing House Bucharest, 151-160.
36. Morae: Usability Testing Solution for Websites and Software:
<http://www.techsmith.com/morae.asp>
37. Agile Usability Yahoo Group: <http://groups.yahoo.com/group/agile-usability/>
38. Jakob Nielsen, *Usability Engineering*. San Diego, CA: Academic Press, 1993, pp. 25
39. Landay, J. A. and Myers, B. A. 1996. Sketching storyboards to illustrate interface behaviors. In Conference Companion on Human Factors in Computing Systems: Common Ground (Vancouver, British Columbia, Canada, April 13 - 18, 1996). M. J. Tauber, Ed. CHI '96. ACM, New York, NY, 193-194. DOI=
<http://doi.acm.org/10.1145/257089.257257>
40. <http://www.theclayman.com/images/GiantBoards.jpg>

41. Nielsen, J. 1994. Guerrilla HCI: using discount usability engineering to penetrate the intimidation barrier. In Cost-Justifying Usability, R. G. Bias and D. J. Mayhew, Eds. Academic Press, Orlando, FL, 245-272.
42. Patton, J. 2003. Improving on Agility: Adding Usage-Centered Design to a Typical Agile Software Development. DOI=
http://agileproductdesign.com/writing/adding_ucd_to_agile_development.pdf
43. Nielsen, J. and Landauer, T. K. 1993. A mathematical model of the finding of usability problems. In Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems (Amsterdam, The Netherlands, April 24 - 29, 1993). CHI '93. ACM, New York, NY, 206-213.
DOI= <http://doi.acm.org/10.1145/169059.169166>
44. Chen, M. C., Anderson, J. R., and Sohn, M. H. 2001. What can a mouse cursor tell us more?: correlation of eye/mouse movements on web browsing. In CHI '01 Extended Abstracts on Human Factors in Computing Systems (Seattle, Washington, March 31 - April 05, 2001). CHI '01. ACM, New York, NY, 281-282. DOI= <http://doi.acm.org/10.1145/634067.634234>
45. Sauro, J & Kindlund E. (In Press) Making Sense of Usability Metrics: Usability and Six Sigma, in Proceedings of the 14th Annual Conference of the Usability Professionals Association, Montreal, Canada.
46. Fox, D., Maurer, F., Sillitto, J. Agile Methods and User-Centered Design: How These Two Methodologies Are Being Successfully Integrated In Industry. In Proceedings of Agile 2008, (To Appear)

APPENDIX A: ETHICS CERTIFICATIONS



CERTIFICATION OF INSTITUTIONAL ETHICS REVIEW


This is to certify that the Conjoint Faculties Research Ethics Board at the University of Calgary has examined the following research proposal and found the proposed research involving human subjects to be in accordance with University of Calgary Guidelines and the Tri-Council Policy Statement on "Ethical Conduct in Research Using Human Subjects". This form and accompanying letter constitute the Certification of Institutional Ethics Review.

File no: **5296**
Applicant(s): **Patrick I. Wilson**
Frank Maurer
Brian D. Fox
Department: **Computer Science**
Project Title: **An Exploration into the Use of Low Fidelity Prototypes in Agile Teams**
Sponsor (if applicable):

Restrictions:

This Certification is subject to the following conditions:

1. Approval is granted only for the project and purposes described in the application.
2. Any modifications to the authorized protocol must be submitted to the Chair, Conjoint Faculties Research Ethics Board for approval.
3. A progress report must be submitted 12 months from the date of this Certification, and should provide the expected completion date for the project.
4. Written notification must be sent to the Board when the project is complete or terminated.



Janice Dickin, Ph.D., LL.B.,
Chair
Conjoint Faculties Research Ethics Board


Date:

Distribution: (1) Applicant, (2) Supervisor (if applicable), (3) Chair, Department Faculty Research Ethics Committee, (4) Sponsor, (5) Conjoint Faculties Research Ethics Board (6) Research Services.



UNIVERSITY OF
CALGARY

CERTIFICATION OF INSTITUTIONAL ETHICS REVIEW

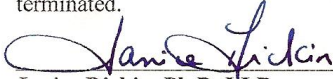
This is to certify that the Conjoint Faculties Research Ethics Board at the University of Calgary has examined the following research proposal and found the proposed research involving human subjects to be in accordance with University of Calgary Guidelines and the Tri-Council Policy Statement on "*Ethical Conduct in Research Using Human Subjects*". This form and accompanying letter constitute the Certification of Institutional Ethics Review.

File no: **5417**
Applicant(s): **Patrick I. Wilson**
Frank Maurer
Department: **Computer Science**
Project Title: **Examination of the Impact of a Software Tool to Facilitate the
Creation and Testing of Low Fidelity User Interface Prototypes**
Sponsor (if
applicable):

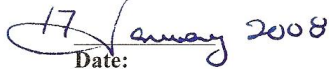
Restrictions:

This Certification is subject to the following conditions:

1. Approval is granted only for the project and purposes described in the application.
2. Any modifications to the authorized protocol must be submitted to the Chair, Conjoint Faculties Research Ethics Board for approval.
3. A progress report must be submitted 12 months from the date of this Certification, and should provide the expected completion date for the project.
4. Written notification must be sent to the Board when the project is complete or terminated.



Janice Dickin, Ph.D, LLB,
Chair
Conjoint Faculties Research Ethics Board


Date:

Distribution: (1) Applicant, (2) Supervisor (if applicable), (3) Chair, Department/Faculty Research Ethics Committee, (4) Sponsor, (5) Conjoint Faculties Research Ethics Board (6) Research Services.

APPENDIX B: WEBSURVEY QUESTIONS

Traditional Agile Methods (AM) strives to deliver working software into the hands of the customer as quickly as possible. In doing so, small feature sets and developed in short iterations concentrate on the immediate rather than the overall project vision. User Centered Design (UCD) on the other hand, strives to uncover users characteristics, context, habits etc.c... before any working software is developed. Both approaches have their strengths and weaknesses. AM is are clearly concerned with delivering the immediate features, the overall vision of the project is might sometimes be lost. Conversely, UCD tends to see all the necessary criteria for the user experience and hence does not concentrate on immediate future (waterfall) (and might we slow in producing tangible results). As a resultTo combine the benefits of both approaches, several Agile teams have begun attempting to incorporate the basics of UCD into their software engineering processes. The purpose of this web survey is to gain insight into how current Agile teams (both traditional and user centered) incorporate low fidelity user interface prototypes (which is a UCD practice) into their methodology, which is a UCD practice.

- 1.) Are you a developer who practices Agile methodologies? (yes / no)
- 2.) Are you or have you ever been responsible for designing User Interfaces? (yes / no)
- 3.) Would you consider yourself a specialist in the field of usability? (yes / no)

The following questions are referring specifically to designing low-fidelity prototypes in an Agile context.

- 4.) Please check each tool you have used for designing low-fidelity user interface prototypes.
 - a. I/My organization does not use low-fidelity prototypes.
 - b. A whiteboard
 - c. A piece of paper
 - d. Microsoft PowerPoint
 - e. A Graphics Drawing Tool (ie: Photoshop or MS Paint)
 - f. Denim
 - g. Microsoft Visio
 - h. Other: _____
- 5.) Which tools do you prefer to use for designing low-fidelity user interface prototypes.
 - a. I/My organization does not use low-fidelity prototypes.
 - b. A whiteboard
 - c. A piece of paper
 - d. Microsoft PowerPoint

- e. A Graphics Drawing Tool (ie: Photoshop or MS Paint)
- f. Denim
- g. Microsoft Visio
- h. Other: _____

6.) What features in any low-fidelity prototype design tool do you find helpful?

7.) What features in any low-fidelity prototype design tool do you find a hindrance?

8.) What would you personally like to see included in a low-fidelity prototype design tool?

Additional Comments: