Collaborative Designing of User Interfaces/User Experience from Different Geographical Locations

Tanmay Songade
Rochester Institute of
Technology
Rochester, USA
tms6649@rit.edu

ABSTRACT
There are many powerful software products available out there in production environment. Most of those software requires contributions from people who are at different geographical locations. And that's where concept of collaborative codding comes into picture. When it comes to field of designing, there were few attempts made in past to perform collaborative designing. Although there are no such tools which can collaborate HIFI prototypes for user experience and user interface. This paper introduces a collaborative designing method in user experience and user interface for designers who are at different geographical locations. This method is divided into two parts, in first part user will have to scan the raw file through dlint. Dlint is a checker, use to find flaws in design that does not correspond to certain style guidelines. For the purpose of study currently dlint is limited to three checks. Once dlint passes final designed raw file, it is then passed through a software where it is merged with its parent file. I have opted for qualitative research method where user interviews were conducted and feedback was taken for this approach.

INTRODUCTION

My research question is in the area for collaborative designing: *Can designers work together on designing user interfaces and user experiences from different geographical locations?* In the field of designing there are different team in the organization. Each team has more than one designers. The designers work together on a project. For the success of the project, designers work together to build final screens. This needs collaboration in different areas such as initial brainstorming, creating wireframes, creating lofi prototypes and final hifi prototypes. There are variety of tools available for design teams to work collaboratively, they can even work in parallel or independently [4]. They may be at different geographical locations and different time zones; thus resulting design process is called as distributed collaborative design [4]. The design teams may collaborate with designers in different location and same time or different location and different time [4].

While we are discussing about design teams we should also think about how software developers collaboratively contribute. Let consider the example of how open source projects works. Open source projects have thousands of contributors and all the contributors as located at different geographical locations and are at different time zones. Developers use software version control system such as git, svn et cetera to integrate code. Such tools use three-way merge technique. A three-way merge is performed after an automated difference analysis between a file "A" and a file "B" while also considering the origin, or common ancestor, of both files "C". It is a rough merging method, but widely applicable since it only requires one common ancestor to reconstruct the changes that are to be merged. The three-way merge looks for sections which are the same in two of the three files. As shown in the figure 1, there are two versions of the section, and the version which is in the common ancestor "C" is discarded, while the version that differs is preserved in the output. If "A" and "B" agree, that is what appears in the output. A section that is the same in "A" and "C" outputs the changed version in "B", and likewise a section that is the same in "B" and "C" outputs the version in "A". Sections that are different in all three files are marked as a conflict situation and left for the user to resolve.

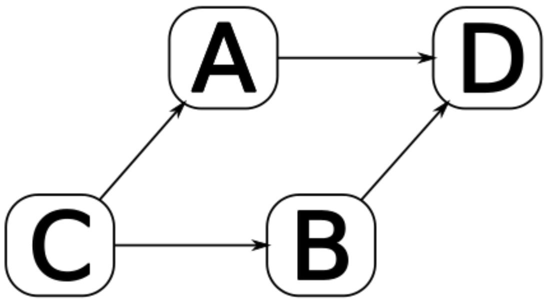

Figure 1 shows C is the origin, A and B are derivatives of C, and D is the new output version

The other tool which helps smooth functioning of software version control systems is Lint. Most of the widely used programming languages have their own linters. For example, python has pylint, javascript has jslint, ruby has rubocop and so on. Lint is a Unix utility that flags some suspicious and non-portable constructs. Lint sometimes is also referred as a tool that flags suspicious usage in software written in any computer language. The term lint-like behavior is sometimes applied to the process of flagging suspicious language usage. Once code is passed through linter then it becomes uniform. Once code is uniform then it becomes relatively easy for the version control systems to integrate.

The proposed method is divided into two parts is you can see in figure 2.

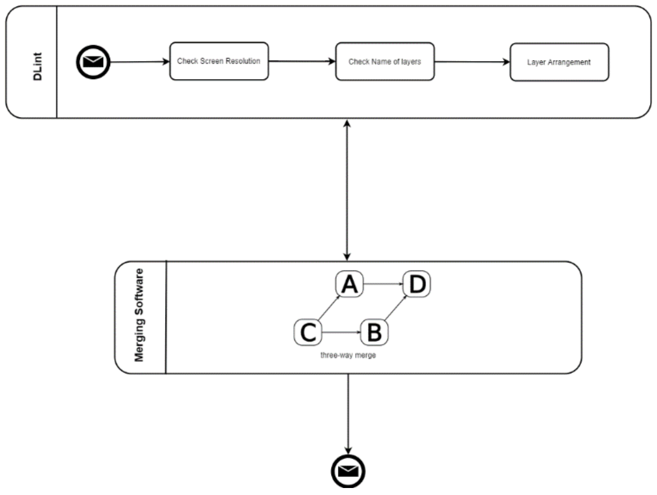

Figure 2 shows how a file is passed through Dlint and Merging Software

First, scan final raw files with dlint. Dlint is a linter tool which will perform three checks. First check will be verifying screen resolution. Second, name of layer should not be layer 0, layer 1 so on. Layer should have some meaningful names. Third, layers should be arranged in the same order as it appears on user interface. Second part is to pass scanned files through a software which will use three-way merge technique to collaborate multiple files. This process will help designer to reduce the burden of manual management and integration of screens. Designer just have to reply of dlint and merging software to collaborate all the work and produce final hifi prototype.

METHOD

For conducting interviews, I kept specific criteria for participants. Participants should have worked on designing user interfaces and user experience for at least five hundred hours till date. I was specific about requirements for participants just to make sure that they have spent considerable amount of time on designing user interfaces and user experiences and they have worked in a team which have more than one designer. I have taken four interviews out of which two interview were analyzed and coded by looking into interview transcripts. Participant 1 (P1) is a 34-year male full time employee working for an organization in India. His job title is User Interaction Designer working on creating hifi prototypes for web applications. He uses Photoshop Adobe tool to create hifi prototypes on regular basis. Being a full time employee he is involved in collaborative process in the initial phase of the project. Later he works individually while creating hifi prototypes. Participant 2 (P2) is a 23-year male student at Rochester Institute of Technology, Rochester, New York. He is also a part time employee of a company in Rochester. He uses multiple tools for creating hifi prototypes such as Sketch, Photoshop, InDesign. Participant 3 (P3) is a 26-year female who is currently student of Industrial Design in Rochester Institute of Technology. She is involved in creating 3D modeling using software like AutoCAD. She is also heavily involved in creating designs for class project using Illustrator an Adobe tool. Overall I interviewed 4 users 3 men (75%) and 1 female (25%).

The interviews conducted were open-ended, with questions focused on their background, their work pattern, their views on proposed method and feedback on who to improved proposed method. I also asked participants questions on the fly which derived from the discussion over particular question. The above mentioned questions where framed in such a way that maximum information can be gathered. Interviews where conducted over phone, skype video call and in-person. Table 1 shows the characteristics of the participants interviewed.

	Age	Sex	Occupation
P1	34	Male	Employee
P2	23	Male	Graduate Student
P3	26	Female	Graduate Student
P4	38	Male	Employee

Table 1. Characteristics of interview participants.

Method of Analysis

For this study I have followed "test-retest method" to analyzing the data collected from user interviews. Where I have coded the transcript for first interview once and then without looking at the results re-coded the same transcripts. The two set of codes were compared and percentage agreement score was 77%. Agreed coding method and codes were then used in second and third interview.

Coding System

In order to have codes all-inclusive and mutually exclusive, initially I have gone through the first transcript once. Then again re-read it looking for specific categories. After collecting all the data, I made an affinity diagram which could help me in segregating the highlighted points. Finally, the transcripts were coded with four codes – Traditional Approach, statements used by the participants explaining how do they integrate design files. Positive Comments, statements used by participants favoring use of dlint and screen merging tool. Negative Comments, statements used by participants criticizing use of dlint and screen merging tool. Suggestions, statements used by participants for modifying dlint checks and merging tool.

RESULTS

Interview 1

Traditional System

My first participant stated that he was developer before he started designing. As a developer he has contributed into the many open source projects. And with the help of decentralized versioning tools he was able to seamlessly collaborate to open source projects.

> *Whenever I use to code I have followed some practices such as to write the documentation for every function. And that is the reason why my code was more readable and when people were using my code they could relate to it quickly.*

Positive Comments
This participant gave really good comments to me when I explained him the concept of dlint. And I asked him if he is interested in using it.

> *I will pass my file through the dlint. It is always helpful for me to know if my design is matching the standards or not. I am also interested in knowing how can dlint show my where are the errors in file.*

He also gave positive comments on software which will be used for merging scanned screens. He liked it because somehow it linked this concept with his past as developer.

> *I am interested in knowing the feedback provided by this software. It will be time saver for me if the software scans the screens itself and mergers it with parent screen.*

Negative Comments
This participant gave me negative comment to the question that the designer always gives a description explaining how he derived to a particular design decision. In this current proposed system where will designer explain his/her design.

> *There is absolutely no room for the designer to give the description of the design he has created. This may lead to the failure of system.*

Suggestions
The feedback given by this user was mostly focused on how software can determine whether the merged file is the expected result or not.

> *There should be a way for the user to write a description about the design which he/she has designed.*

Interview 2

Traditional System
My second participant has explored many applications while he was working with clients on campus. He stated that he likes how collaborative documentation works.

> *Siebel software for documenting client interactions and technical details of on-site hardware diagnostics in corporate environments like Dell International Services and Thomson Reuters.*

Positive Comments
He liked the idea of collaborative designing. He said that it is a good channel or platform for the designers who have a sprouting idea of design. And once he uploaded his rough idea of concept then he can ask other designers to contribute to his idea and make it real.

> *Creates a good open source environment to seed ideas into the common psyche of designers and artists.*

Negative Comments
His concern was what if the software is central and everyone is merging their designs into a central server then there are chances that unwanted data can replaced meaningful data in the central server. This will then become virus, as later all the derived screens will have same unwanted data.

> *The design updates that would be uploaded to the common designs from which everyone can derive needs to have a control group of round table users who can approve the design by voting.*

Another flow which he mentioned that will not work here is in case of complex design. As in complex designs there is also a third dimension. The elements are over one another, then how will dlint will check such designs.

> *If the designs included background layer or overlapping* images *of multiple layers then this system will fail to check such three dimensions designs.*

Suggestions
The feedback was mostly related to merging files and creation of master file. What he suggested that designers should not completely depend on software to generate the final screen. There should be review process in between which will be group of people who will review new design and then approve the merge.

> *Create a hierarchy of pre-approved round table users who are trusted as designers and artists who can vote to approve design updates.*

Another suggestion by this user was that the system should be peer to peer. There should not be any central data, it should be distributed amongst authorized users.

> *Create the user interface on a peer-to-peer platform to increase exposure for budding artists who can seed ideas that can then be added onto by others by leaving its proprietary to the community.*

DISCUSSION
There are couple of limitations of the study. One of them is that the proposed system does not have the process of reviewing the designs before merging it into the software. As shown in the new flow diagram software should have a process of reviewing the design by expert uses.

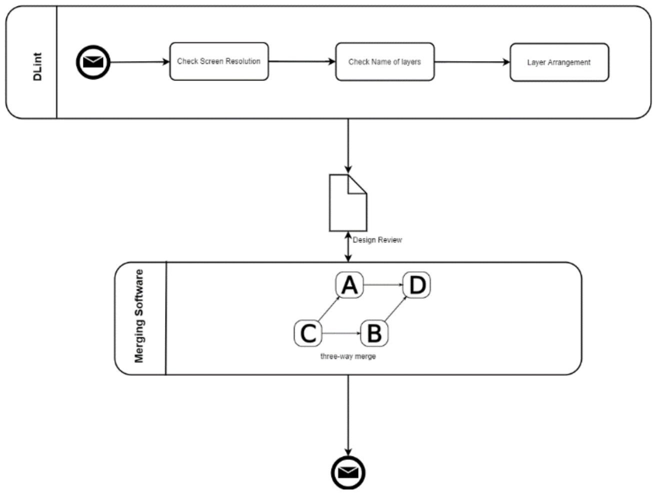

Figure 2 shows how a file is first passed through Dlint then it is reviewed and then Merging Software merges the file

Another limitation was related to considering the parameters while calculating the coded data from user transcripts. In this paper percentage of acceptance is calculated but it has two drawbacks. First, it cannot give a valid comparison of the reliability of coding one interview with the reliability of another interview on a different topic with a different number of coding categories [7]. Second, it does not give a valid clue to whether a particular percentage is acceptable or not [7]. In order to overcome these drawback, I should have calculated coefficient of reliability. It can be calculated by following formula:

$$r = \frac{N\Sigma XY - (\Sigma X)(\Sigma Y)}{\sqrt{[N\Sigma X^2 - (\Sigma X)^2][N\Sigma Y^2 - (\Sigma Y)^2]}}$$

Figure 3 shows formula for calculating coefficient of reliability.

N is the total number of pairs of test and retest scores. XY means to multiply X by Y, where X and Y are the test and retest scores. The Greek symbol Sigma means 'the sum of'. So SigmaXY means to sum all the pairs of test scores (X) multiplied by retest scores (Y).

My hypothesis for this study was that the proposed study is helpful for collaboration of user interface designs from different geographical locations. I was looking for answers from user which will prove this hypothesis. I got positive responses from the user saying they will use this system for collaboration when it comes to UI/UX design.

REFERENCES

1. George Chin, Jr. , Mary Beth Rosson, Progressive design: staged evolution of scenarios in the design of a collaborative science learning environment, Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, p.611-618, April 18-23, 1998, Los Angeles, California, USA [doi>10.1145/274644.274726]
2. Sachs, P. (1995). Transforming work: Collaboration. learning and design. Communications of the A CAl 38.9, 46-55.
3. Aggelos Liapis, Synergy: a prototype collaborative environment to support the conceptual stages of the design process, Proceedings of the 3rd international conference on Digital Interactive Media in Entertainment and Arts, September 10-12, 2008, Athens, Greece [doi>10.1145/1413634.1413665].
4. Verbeke, G. 2001. A Future Focus on Collaborative Design", Hogeschool voor Wetenschap & Kunst, Saint Lucas Architecture, Brussels, Belgium, Technical Design & Informatics, Faculty of Architecture, Delft University of Technology, the Netherlands.
5. Mitchell, W. J. 2004. Challenges and Opportunities for Remote Collaborative Design, Massachusetts Institute of Technology.
6. Scrivener, S. A. R., Ball, L. J., and Woodcock, W. 2001. Collaborative Design, Co-Designing 2000 Coventry University.
7. Gorden, Raymond (1992). Basic Inteviewing Skills. Itasca, IL: F. E. Peacock http://www.indiana.edu/~educy520/sec5982/week_5/qual_data_analy_ex2.pdf
8. J. Tate and T. C. Clancy. Secure and tamper proof code management. In Workshop on Cyber Security Analytics; Intelligence and Automation (SafeConfig), Nov. 2014.
9. Mohammed Elkoutbi , Ismaïl Khriss , Rudolf K. Keller, Automated Prototyping of User Interfaces Based on UML Scenarios, Automated Software Engineering, v.13 n.1, p.5-40, January 2006 [doi>10.1007/s10515-006-5465-5].
10. John M. Carroll , Mary Beth Rosson , George Chin , Jürgen Koenemann, Requirements development: stages of opportunity for collaborative needs discovery, Proceedings of the 2nd conference on Designing interactive systems: processes, practices, methods, and techniques, p.55-64, August 18-20, 1997, Amsterdam, The Netherlands [doi>10.1145/263552.263577]