

# REDESIGN OF A SEISMIC MONITOR USING CONTEXTUAL DESIGN

Ozgur Aktunc, Berat A. Erol, and John D. Garcia

Engineering Department, St. Mary's University, San Antonio, USA  
oaktunc@stmarytx.edu, berol@mail.stmarytx.edu,  
jgarcia32@mail.stmarytx.edu

## **ABSTRACT**

*This paper discusses a user-centered approach to requirements gathering and design and its application to an agile software development project. The approach used in this paper is based on Contextual Design (CD), a user-centered design technique, developed by Beyer and Holtzblatt. The benefits of using CD to requirements gathering and usability are explained using a case study, a Web-based seismic monitor, which allows a user to monitor earthquakes all around the world in real time. This case study demonstrates the benefits of CD by the improved design and usability of the application. CD shares some of the fundamental principles of agile software development processes, such as continuous user feedback and rapid prototyping. This makes CD a natural candidate to be used in agile software development.*

## **KEYWORDS**

*Software Process, Software Design, Software Requirements, Software Methodologies, Web Engineering, Graphical User Interfaces*

## **1. INTRODUCTION**

Software systems are becoming more complex and there have been a number of methodologies to deal with the inherent complexity of large software systems, such as agile development processes and component-based development. These methodologies focus on developing software modules separately and integrating loosely coupled modules into systems and using prototypes for iterative development. The core ideas in popular agile development processes, Scrum and XP, are to reduce the time spent on documentation and to defer design work until it is absolutely necessary, start implementing the core features early, and start testing the units early and frequently during the development process [1, 2]. The agile methodologies have been adopted rapidly in the last years and have been the dominant development processes.

Agile software development embraces a blend of ideas originated from iterative and incremental development methodologies. Agile has been described as an adaptive and people-focused approach in the literature [3]. Communication among developers and users and the emphasis of delivering working software over extensive documentation have been defined as characteristics of agile approaches [4]. Agile development has a more customer friendly approach compared to traditional plan-driven processes, such as Waterfall, when it comes to requirements gathering. Two of the twelve principles of Agile Manifesto [5], which was published in 2001, state that agile developers should welcome changing requirements even late in development and the most efficient and effective method of conveying information is face-to-face conversation. These two principles guided this research to find a user-centered approach for requirements gathering and

design. A user-centered approach requires strong and continuous customer developer collaboration, where users are involved with the project efforts and provide frequent feedback to developers. In fact, many of the agile methodologies put the customer in the development team as a core component. Scrum defines a Product Owner role to represent the customer within the development team [1]. XP gives the customer, the authority to make business decisions by defining the project and its goals [6].

User stories have been the primary tool to capture and document the user needs for most agile processes. In Scrum, Product Owner writes user stories and prioritizes them to be added to a product backlog [6]. User stories are collections of statements that describe the usage of a system from a user's perspective. User stories are embraced by the agile community due to their simple approach to requirements gathering, and also being good at drawing people into discussions about how they want to use the system. Even though user stories are good at capturing possible usage scenarios, there is still a need for other tools to capture other types of user requirements. These are listed by Bittner [7] as requirements that do not have a flow (such as a condition that needs to be satisfied any time if occurs, as with security requirements), or for non-functional requirements (such as technical constraints and performance requirements). Bittner suggested that a successful requirements approach should be a blend of different approaches depending on the domain and the users. Bittner provided a taxonomy that consisted of declarative, scenario-based, model-based, prototyping-based, testing-based, and backlog-based approaches [7].

In this paper, we are presenting the application of a user-centered requirements and design approach called Contextual Design in a case study, a Web-based seismic monitor, which allows a user to monitor earthquakes all around the world in real time. This work was a redesign of this application using the Contextual Design, which were described by Holtzblatt and Beyer in 1995 [8]. The requirements for the application were gathered using the Contextual Inquiry method, which is part of Contextual Design. Contextual Inquiry (CI) emphasizes face-to-face communication and collaboration with the customer. This makes CI a natural candidate to use in an agile project as user involvement is necessary for agile development. Our case study also includes the redesign effort using the CD work models that are explained in section 3.

## **2. REQUIREMENTS GATHERING**

A common issue in software development is the difficulty that customers have while articulating their needs. Many users have a hard time articulating their own work practice and they need to be guided by requirements specialists using various elicitation techniques, such as storyboarding and role playing. Work practice refers to the behaviors, goals, and intents of users in their own work environment. The term, work practice, is not only limited to a workplace but also includes to any domain where the users perform tasks to achieve their goals. A work practice can be associated with a business activity, such as writing an accounting application or with a life event, such as purchasing items from an online store. Requirement elicitation techniques can be used to describe the users' work practice in order to give an insight to the processes and tasks that take place in the work practice. The process of requirements elicitation is generally accepted to be one of the most difficult and challenging activities in software development [9]. Various techniques and tools have been proposed in this area but lack of communication and understanding between stakeholders continues to be one of the primary reasons of failed projects. There are two challenges for requirement analysts to elicit the requirements. The first is that the users are usually not fully aware of their own work practice when they are taken out of the context of their everyday environment. The second is that work practice is complex and varied to describe for users. Using an ethnographic technique, such as field interviews at the user's work settings would uncover the motivations and strategies that users cannot articulate by other means. Requirements gathering is the activity of identifying and generating the requirements for a system by obtaining

them from the system's users and stakeholders. There are many elicitation techniques defined in the literature, such as interviews, user observation, questionnaires, brainstorming, prototyping, user stories, and requirements workshops. Each of these techniques has particular strengths and drawbacks. The choice of elicitation technique will depend on the users and their work practice. The main objectives while choosing an elicitation technique are to optimize the communication between the developers and the users and assist the users to articulate their needs in the most efficient way.

There are several issues that need to be considered while gathering requirements:

- Availability of the users and stakeholders
- Communication issues between developers and users
- Knowledge of the development team about the work domain
- Understanding of the current system if it is a redesign effort

Development teams need to address these issues before choosing appropriate requirement gathering methods, activities, and tools. The authors surveyed a variety of methods and tools before deciding on using Contextual Inquiry as the requirements gathering method.

### **3. CONTEXTUAL INQUIRY AND DESIGN**

Contextual Design (CD) is a customer-centered design process found by Holtzblatt and Beyer [8]. It uses extensive field data as the foundation for understanding users' needs, tasks, intents, and processes in order to design products and systems that satisfy the users. Contextual Design uses an ethnographic elicitation technique, Contextual Inquiry (CI), to gather requirements from users. CI suggests observing the users in their work environment, following their work practices, recording their work processes, and asking the users questions when necessary. The main objectives of CI are to investigate the users' workflow and to understand how they are doing the work. Generally, the chosen users for CI are the end users; however CI also requires communication with indirect users, such as independent stakeholders, and managers that assign work to the users [10]. Some methods that are employed in CI include video recording, note taking, following a context and asking questions to users if necessary, but also avoiding an interview setting. Developers assume the role of the apprentice in a relationship that was defined as the master and the apprentice by Holtzblatt and Beyer [8]. The observation of users is considered to be a more reliable method than interviewing the users, because users cannot always explain their needs and do not always know what they need. The data collected from CI is used in the CD process, when the development team meets to analyze and interpret the results and generate the requirements. CD is a well-defined, structured, user-centered design process that provides the methods to analyze user data and generate design models based on this data. The key principles that shaped the development of Contextual Design and provide the key motivations for its use as a design tool were described by Holtzblatt and Beyer [11]:

- System design must support and extend the users' work practices.
- People are experts at what they do, but unable to articulate their own work practice.
- Good design requires partnership and participation with users.
- Good design is systemic.
- Design depends on explicit representation.

During the design process of the case study, described in section 4, the authors took these principles and their implications into consideration. The authors formulated these guidelines based on the principles given above:

- The requirement analyst must actively question the users and suggest interpretations of their actions and motivations. The design should be based on the collaborative work of the analyst and the users, not what the analyst understands from the work practice.
- The designer must use tangible and expressive representations to capture design considerations at every step of the development process. These representations have to be improved based on users' feedback iteratively.

In the next section, the CD process and activities are described in detail.

### 3.1. Contextual Design Process and Activities

Contextual Design activities can be grouped under two types:

- Requirements and solutions
  - Contextual Inquiry
  - Interpretation session
  - Creating work models
  - Affinity diagramming
  - Visioning
- Define and validate concepts
  - Storyboarding
  - User environment design
  - Paper mock-ups
  - Interaction and visual design

The relationship among these activities and the typical order of them taking place in a contextual design process are illustrated in Figure 1.

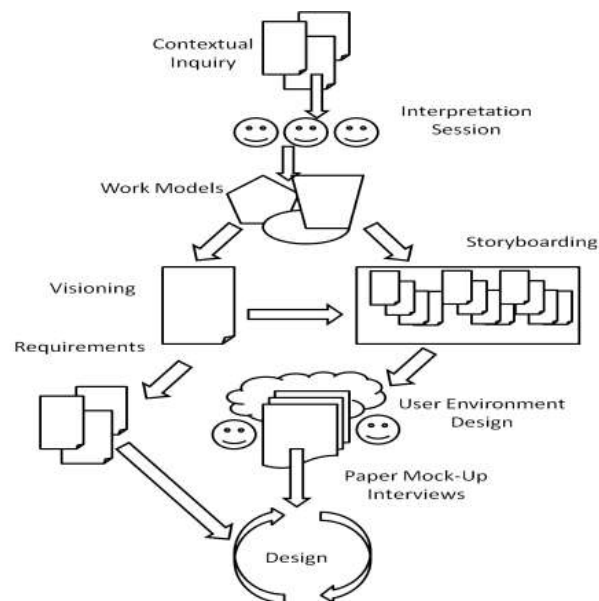


Figure 1. Contextual Design Process and Activities

The initial data is generated as a result of the CI sessions held with the users by observing their work practices in their work environments. Based on this data, design teams create work models that represent the key aspects of the system. The preliminary requirements set that developers come up at this stage offers the opportunity to receive feedback from the customers in the visioning phase. This stage up to this point can be defined as the “requirements and solutions.” In the second stage, “define and validate concepts”, the requirements generated previously become the input to the design process and the development teams start working to redesign the users’ environment to support the new system. Paper prototypes are helpful to receive feedback from the customers through an iterative design process. At this stage, the top priorities for the designers are high usability rates and user experience [10]. The requirements gathering approach is closely related to the design activities as illustrated in Figure 2. The requirements are gathered using the CI method and the requirements data is interpreted by the developers via the CD process. The data should be interpreted by a cross-disciplinary design team to have different perspectives to the data, such as business, design, and marketing. The interpretation sessions are crucial to the process, as the team starts to understand the users and their needs, while they are creating the work models.

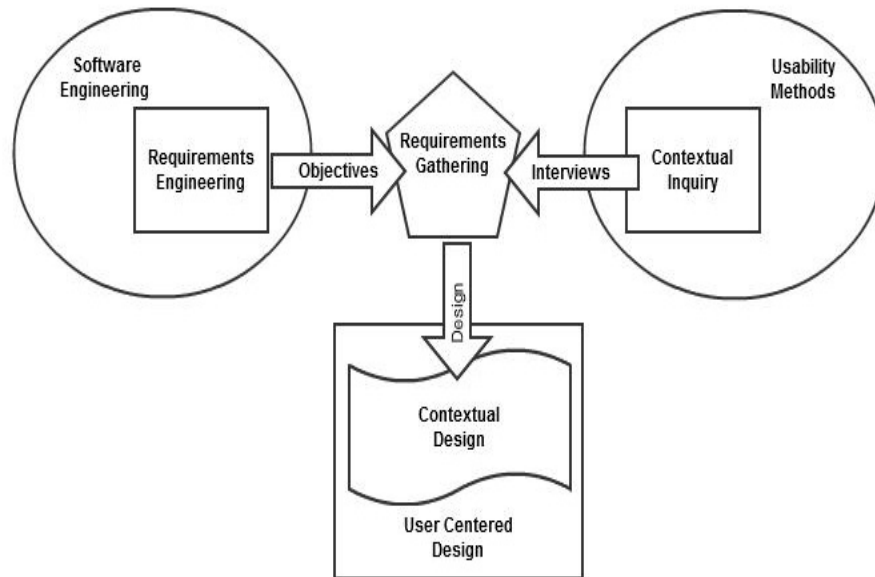


Figure 2. Requirements Gathering Process

### 3.2. Work Models

Work models describe the work practice from the users’ point of view. They illustrate the communication among the users and the system from the designers’ perspectives. There are five different work models that provide five perspectives on how work is done: flow model, cultural model, sequence model, physical model, and artifact model. They represent the key aspects of work that designers need to integrate into their design. Therefore, in CD, work models capture and illustrate the work of individual users and the workflow of the organizations.

#### 3.2.1 Flow Model

As Hartson and Pyla stated, “Flow Model is a picture of work domain, its components, and interconnections among them.” [10] We can see who does what, and who communicates with

whom to accomplish the work. The flow model also shows breakdowns, obstacles that interrupt the communication flow among the users, the system, and other elements. People or groups are shown as a bubble. Flow is shown as arrows between individuals, artifacts are shown as small boxes, and breakdowns are shown as red lightning bolts. An example flow model is shown in Figure 3 for the case study explained in this paper.

### 3.2.2 Cultural Model

The cultural model illustrates the culture of the organization, such as expectations, policies, values, culture of the country, laws, and legal regulations. So, it is important to illustrate the cultural context of an organization's formal and informal work policies. It shows how people are constrained and how they work around those constraints. As Holtzblatt and Beyer described, "Cultural Models do not map to organization charts." [11] They show work experiences of the users and their intention to accomplish the work. An example cultural model is illustrated in Figure 4.

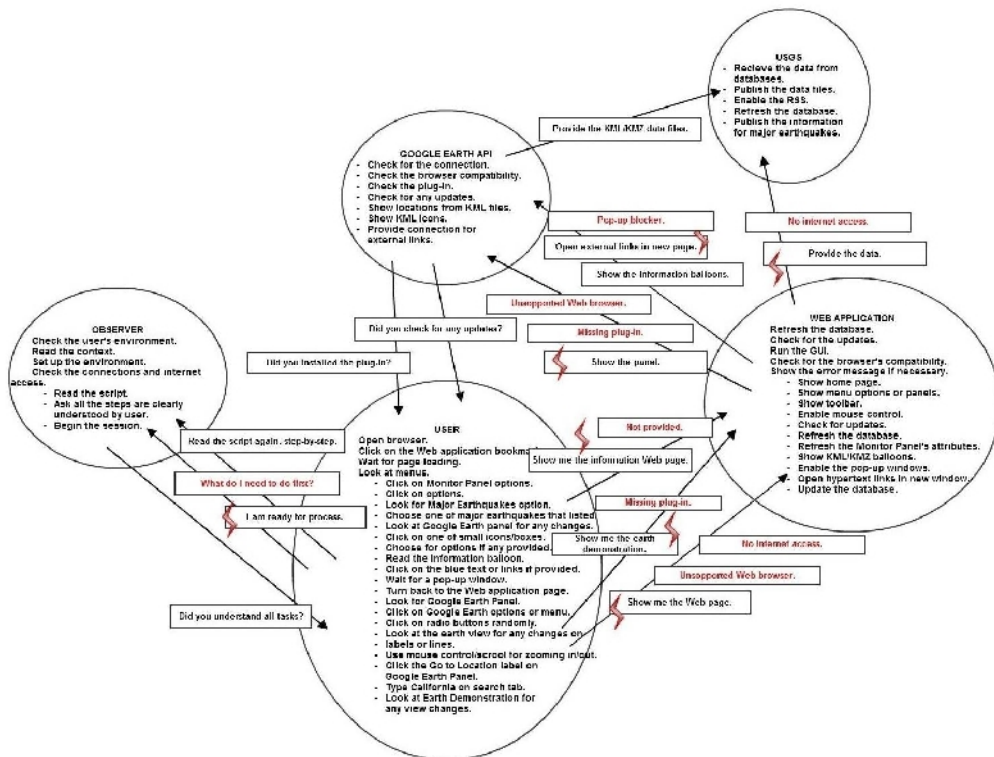


Figure 3. Flow Model for the Seismic Monitor

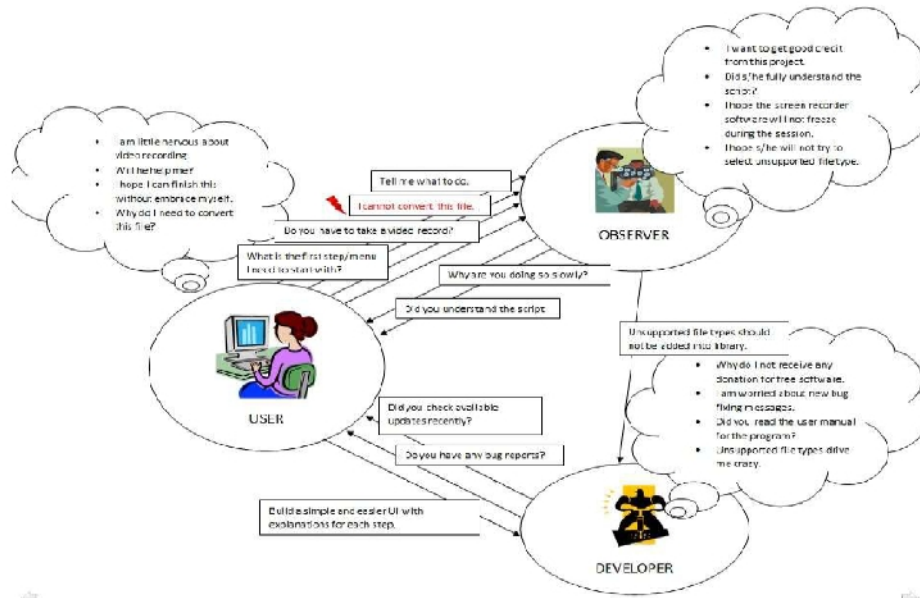


Figure 4. Cultural Model

### 3.2.3 Sequence Model

As Holtblatt and Beyer stated “The actions people take in doing their work reveal their strategy, their intent, and what matters to them.” [11] The sequence model shows those intents and strategies, or ways of doing the work, in a systematic structure. In addition, it explains triggers and orders for those intents based on the workflow and the breakdowns. The main objective of the sequence model is to analyze the current workflow; then to describe and design a better path for the same process more efficiently. An example sequence model is illustrated in Figure 5.

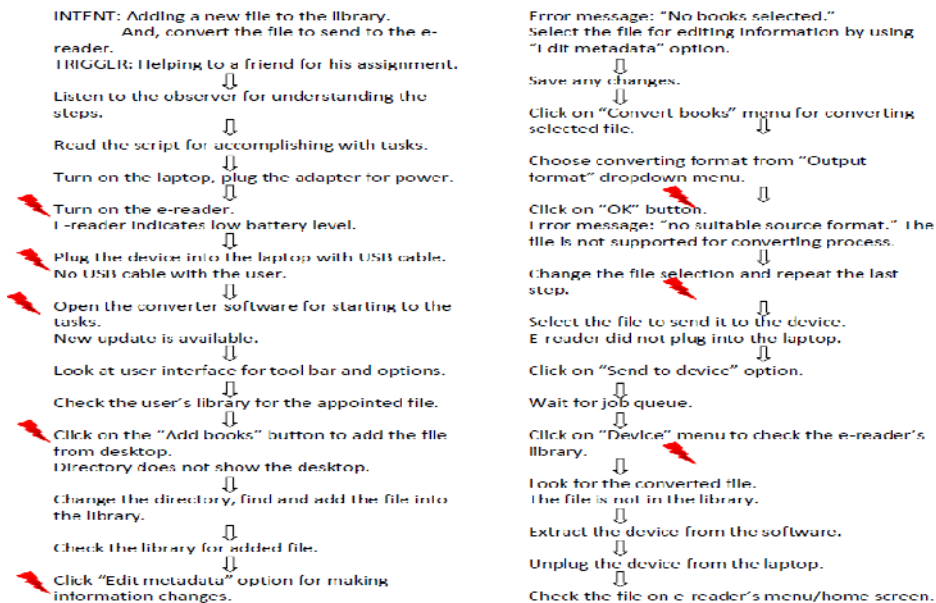


Figure 5. Sequence Model

### 3.2.4 Physical Model

It shows the physical environment of the work place. The physical model shows how the users organize their work environment to make the workflow easier and to accomplish the work in a short time. Their moves, the tools they use, such as simple office equipment, hardware and software used in the work context, and communication lines among other people and things are described in the physical model. An example physical model is illustrated in Figure 6.

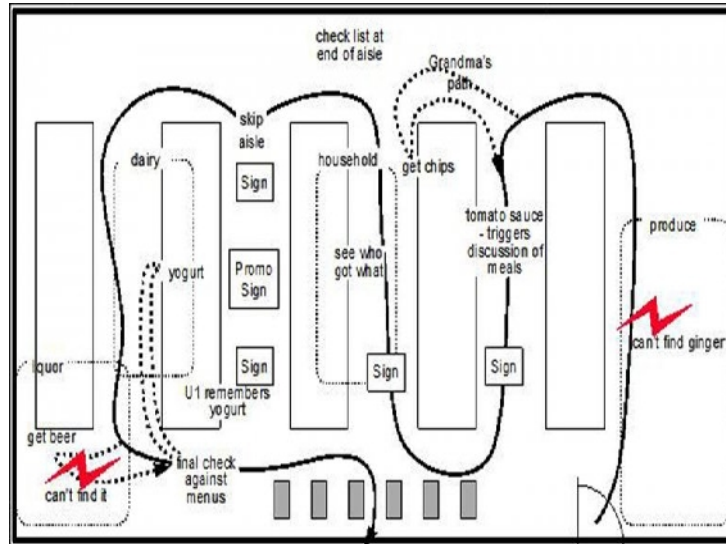


Figure 6. Physical Model [11]

### 3.2.5 Artifact Model

The Artifact Model shows the artifacts that are affecting the work flow, and used or created during the work process by the users. Its main objective is to collect the information for objects and their parts that are most used, structure the parts, such as an agenda or to do list, annotate to indicate the informal usage styles, and, finally, to present those objects as color, shape, or layout. An example artifact model is illustrated in Figure 7.

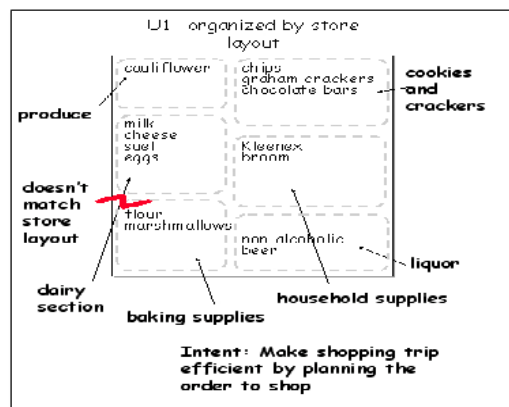


Figure 7. Artifact Model [11]



### 3.3 Consolidation

During consolidation, the design team consolidates the work models that were created for many users so the team can see the associations and dependencies of the work practices for these users. The consolidated work models should reveal common strategies, workflows, users' environments, and constraints. Figure 8 illustrates the consolidation of four work models. Designers can also use affinity diagrams to bring the issues and insights of all customers into a large, hierarchical diagram. Affinity diagrams are used by integrating large amounts of data to find the relationships among concepts. Designers can discover previously unseen associations and uncover patterns. Figure 9 illustrates the affinity diagramming technique.

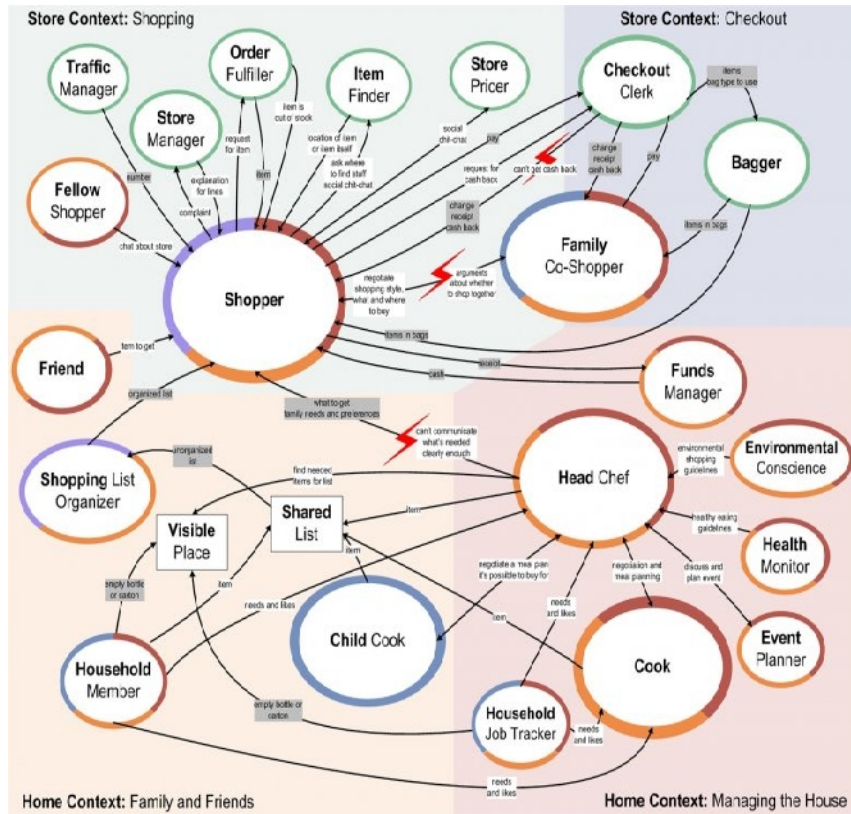


Figure 8. Consolidated Flow Model [11]



Figure 9. Affinity diagramming

The consolidated work models and affinity diagram together represent the work practice of the user population, and provides a reference for designers to understand users' goals and intentions.

### 3.4 Visioning

During visioning, the design team brainstorms by using the consolidated data to envision how to improve the users' workflow and environment with the help of technology and by transforming the work practices. The main purpose is to illustrate the users' behaviors in the newly designed work environment. The vision includes the new system, how it will be delivered, and how it will be supported to make the new work practice successful. The team can start using high-level diagrams, paper prototypes, or wireframes to simulate the new work practice. These low fidelity diagrams allows the designers to use their creative skills and serve as prototypes for future work practice.

### 3.5 Storyboarding

Storyboards illustrate how redesigned work flows and how tasks will be achieved in the new work environment. Storyboards can be captured as use cases diagrams, which explain the communication and interaction among users and system components around the system domain. Storyboards illustrate all the steps required to accomplish the redesigned work.

### 3.6 User Environment Design

Based on work models, consolidated data, and storyboards, the new user environment is designed. It shows how the new environment will support the way users accomplish the work. Each storyboard scenario captures one specific part of the overall tasks. Figure 10 illustrates the interactions of CD processes that have been discussed so far.

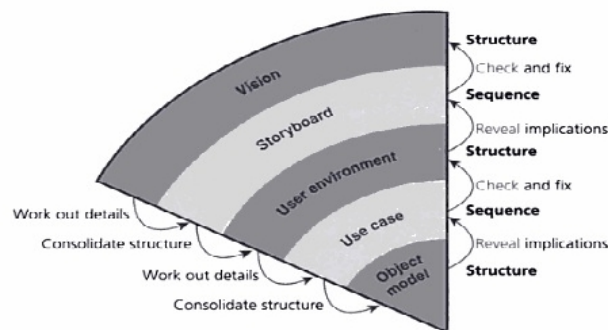


Figure 10. Contextual Design: From Design to Development

### 3.7 Paper Prototyping

Early prototyping is crucial for validating the requirements, as well as for testing and usability purposes. The earlier the problems are found, the less it will cost to fix and solve them. Also, prototyping helps us to receive user feedback for the main design and implementation phases. Paper prototyping has been described as the cheapest and easiest way to generate user comments for early design sketches by Nielsen [12]. We can also get the users involved with the testing of early design ideas at a very low cost.

### **3.8 Interaction and Visual Design**

Interaction and visual design process is an iterative stage that runs together with prototyping. The main purposes of this stage are to build an iterative process to increase user experience with new design and to test the new interface with an incremental prototyping process. It is crucial to get the users involved in this process and receive feedback. Interaction design facilitates the interaction between users and their environment. At this stage, development team needs to add breadth to the initial prototypes, including high fidelity graphic elements and functional prototypes. Although this appears to be the final stage of CD, the visual design activities can continue concurrently with the system implementation. In fact, during agile development process, CD practices continue to support the development team in several ways. One of them is to provide confidence while prioritizing the user stories. As the team has a good understanding of what the user needs are, user stories can be prioritized while they are places in the product backlog. Another benefit of CD is the increased levels of system usability, as the users have already approved many of the system features during prototyping.

## **4. CASE STUDY: SEISMIC MONITOR – A WEB BASED REAL TIME EARTHQUAKE TRACKER**

The user-centered approach summarized in this paper was applied to a Web application that tracks the seismic activities around the world in real time. The initial version of this application was developed two years ago by one of the authors of this paper, Garcia. The core functions of the system were implemented successfully; however there were a number of usability issues as the requirements were gathered in an ad-hoc way without any user feedback. This application was chosen as a test bed to measure the success of our approach by comparing the usability of the old version to the new one. In this section, the case study will be summarized by describing our implementation process and the outputs of our application.

### **4.1 Overview of the Application**

The seismic monitor application receives the earthquake data from the United States Geographic Survey (USGS) [13]. The web application enables earthquake tracking for a rolling 30-day period. The system displays earthquake data provided by USGS on a map using the Google Earth API [14] with a global perspective of past and current earthquake activity. Along with earthquake location, earthquake depth and magnitude also are provided to users via a KML (Keyhole Markup Language) icon on the map. The applications usage is informational only, providing limited information for non-emergency purposes. It was redesigned as a fully modifiable and open application, as the users can customize the user interface; add additional technologies, such as support for Open Layers, ESRI (Environmental System Research Institute) technologies. The application is currently running on the institution's servers and available for public use: <http://cl.stmarytx.edu/~jgarcia/earth/geoquake.html>.

### **4.2 The Initial Requirements and Design**

The initial design of the application was done by a single developer that followed a traditional design process. The developer determined the key high level goals for the system and designed the application based on the use cases generated from these goals. The use cases are beyond the scope of this paper, thus only the high level features and the initial prototype for the application will be given in this section.

### 4.2.1 High Level Features List

The Earthquake Alert real-time system offers the following features:

- Allow the user to view the latest earthquake information
- Allow the user to view more detailed earthquake information if the magnitude of the event is great enough (exceeds a magnitude threshold determined by the USGS)
- Allow the user to display the road names and town names using Google Earth layers
- Allow the user to display the buildings layer in high and low resolution
- Allow the user to navigate around the Google Earth globe with all the same controls that are currently available on the Google Earth Desktop application
- Allow the user to refresh the screen as needed
- Provide the user with a navigation text field that allows for destinations in coordinates and destination names
- Allow the user to display the Google Earth overview map
- Display to the user the most recent earthquake activity on the application user interface via an icon that will signify the location of the seismic activity. The icon can be clicked to display a Google Earth KML balloon that will contain additional information about the seismic event

### 4.2.2 Early Design Stages

The initial user interface design was generated rapidly using wireframes. Simplicity and easy access to core functions were the highest priorities. The developer chose to place all the menus and selections on the left hand side of the screen for simplicity. The border layout was chosen and the central panel was used for the displaying the globe. A wireframe of the early version of this application is illustrated with Figure 11.

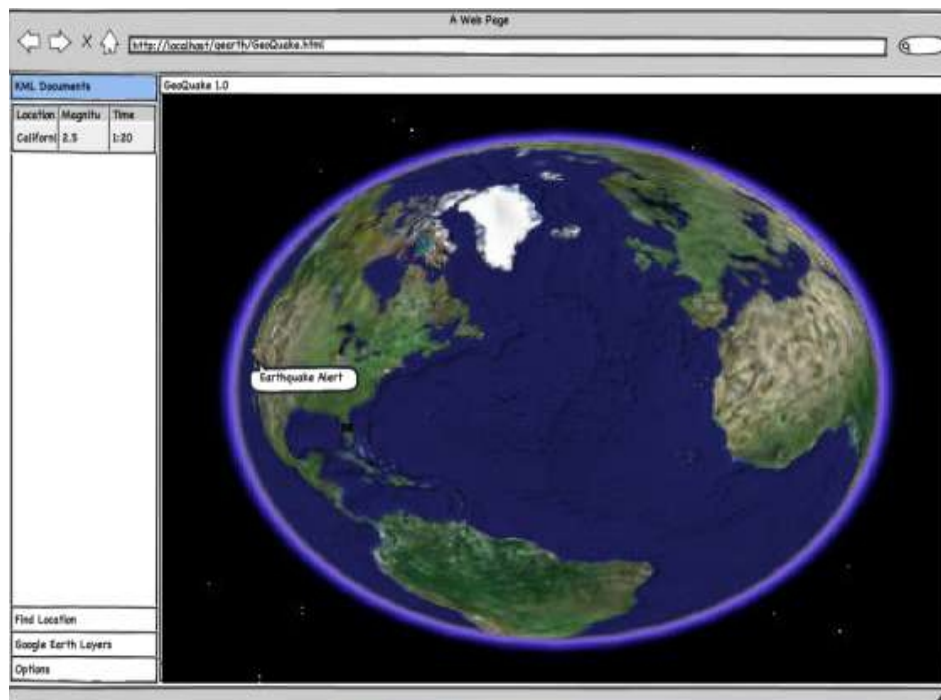


Figure 11. Early User Interface Design

### 4.3 The Redesign of the Application

The first step to the redesign process was to prepare for the contextual inquiry sessions that will be held with the users. The users were selected from a diverse group of college students that were going to use this application the first time. The CI sessions had to be recorded so the developers installed a desktop capturing software to monitor users' cursor moves, responses, and actions with the software and also used a camcorder to record their physical responses. The basic instructions given to the users were prepared in advance and each user received the same instructions. The tasks given to the users included finding out the seismic activities in a specific country and finding out the details of the earthquakes. The observer followed these guidelines during the sessions:

- User is not given any negative or positive signals verbally or by the observer's body language
- User is not prompted to carry out tasks differently
- User is to be listened and observed attentively
- User is provided basic help only if he/she is not able to proceed with the current task at all
- Users actions are only recorded with his/her approval
- User is not given any feedback based on his/her success or failure

The results of the contextual inquiry were discussed in the "interpretation session" and these conclusions were made:

- The application's options menu needed to change
- Google Earth API's options menu had to be represented in a separate layout
- The monitoring options menu had to be redesigned utilizing an Accordion Layout
- Gestalt principles had to be considered during redesign
- A faster job scheduler was needed to receive the data from the agency's database to avoid lag
- Language support for popular worldwide languages had to be added
- Mouse navigation option for the API had to be provided
- Some language changes in the toolbar had to be made
- Failures based on the browser types and versions had to be fixed
- Color and font sizes had to be changed to increase usability
- More user feedback had to be collected for future iterations
- Objectives for future versions and design renovations had to be documented

Based on the communication and interaction results from the CI sessions, developers created work models for the new design. Some of the work models are not included in the publication due to their large size and high level of detail, but they are available upon request. The flow model can be observed in Figure 3. The use case model was recreated based on the new requirements as shown in Figure 12. As the developer was following an agile process, the basic use cases were written in brief formats. The focus of the development effort was the functionality of the system more than the documentation. However, a product vision document was written to give a high level overview of the system. The high level features of the system that were given in section 4.2.1 were part of the vision document. The vision document also included a system context diagram to visualize the interaction of the system components. The system context diagram is illustrated in Figure 13.

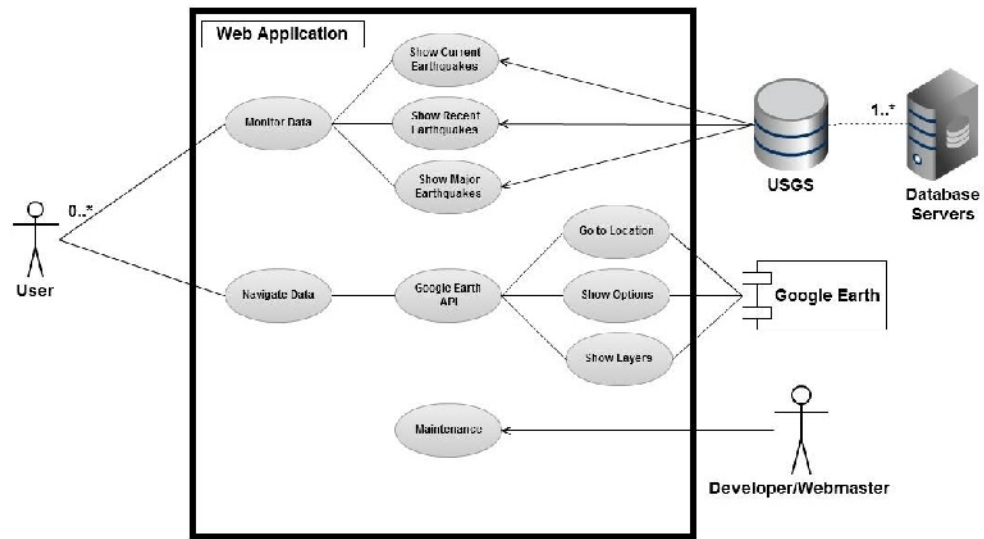


Figure 12. Use Case Model

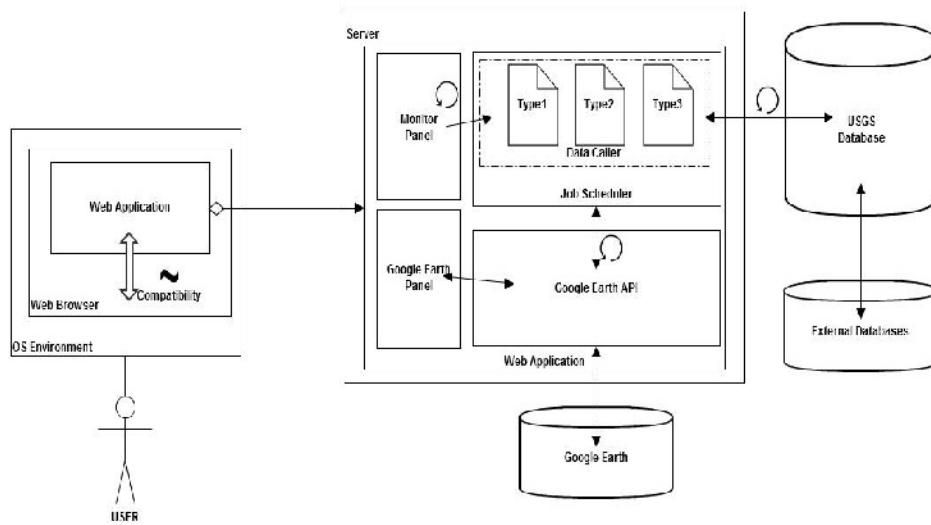


Figure 13. System Context Diagram

The current design was initially generated using the wireframe shown in Figure 14.



Figure 14. Redesigned User Interface

The developers documented the following changes with the new interface: “The first thing to notice is that there has been a clear separation of functionality. We use a typical border layout that is popular among various applications. The “Western” panel makes use of the accordion layout and contains historical earthquake data that is presented via a Grid object. The “Northern” and “Southern” panel are not used in this implementation but may be used in the future if the need arises. The “Center” panel is where the Google Earth plug-in widget resides and is the main component of the application. The “East” panel is where we again make use of the accordion layout and group all the Google Earth options that are currently supported. In addition to these, we also added a language support option for the application that will be implemented in future releases.” These changes were implemented consequently and figures 15 and 16 display the current version of the Seismic Monitor application. The application is available at <http://cl.stmarytx.edu/~jgarcia/gearth/geoquake.html>. It supports all popular browsers and may require the installation of Google Earth plug-in.

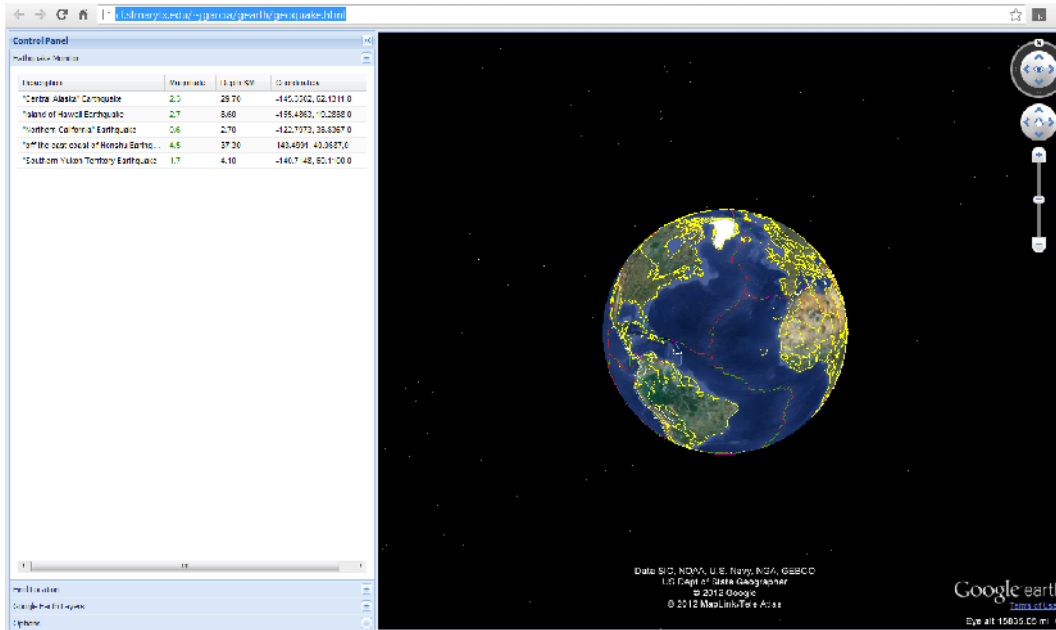


Figure 15. Home Page

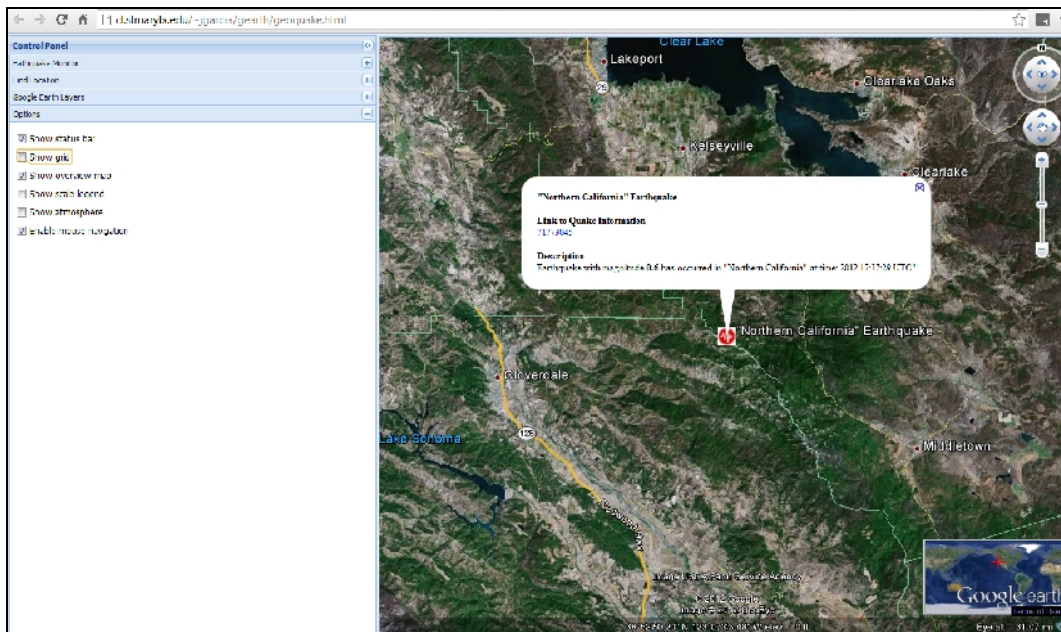


Figure 16. Information Balloons and Data Buttons

## 5. CONCLUSION

In this paper, we explained a user-centered approach to requirements gathering and design, that is applicable to agile projects. Contextual Inquiry and Design were explained and their relationship with Requirements Engineering and User Centered Design were established. We observed that CD provides reliable data from users' points of view, and helps the designer to generate design



ideas. CI is a resource intensive technique, where users had to be interviewed and recorded followed by interpretation sessions. The authors gained invaluable experience from the CI and CD activities during the project. After applying CD to our case study, we concluded CD not only provides us reliable user data from the users' work environment, but also increases usability rates for the final product. Meanwhile, it reduces the time for generating the system requirements and raises communication skills among the users, developers, and other stakeholders. CD provides invaluable knowledge to the development team to write accurate user stories. The interviews and the work models provide a deep understanding of the user and the work practice. In agile projects, the teams usually work on all aspects of development in parallel, such as writing user stories and writing test cases. CD techniques can also take place concurrently with other activities during development. It has to be noted that CI is a qualitative research technique. The data collected during CI may not be considered statistically significant. A future work for this study would be to perform follow up surveys with a larger number of users to provide quantitative data. The authors intend to validate the results with more data collected from the users of this application.

## REFERENCES

- [1] K. Schwaber and M. Beedle. 2001. Agile Software Development with Scrum (1st Ed.). Prentice Hall PTR, Upper Saddle River, NJ, USA.
- [2] K. Beck and C. Andres. 2004. Extreme Programming Explained: Embrace Change (2nd Ed.). Addison-Wesley Professional.
- [3] V. Lalsing, S. Kishnah, and S. Pudaruth. 2012. People Factors in Agile Software Development and Project Management. International Journal of Software Engineering & Applications (IJSEA), Vol.3, No.1, pp. 117-137.
- [4] V. Dattatreya and K.V.C. Rao. 2012. Agile Programming and Design Patterns in Web Development – A Case Study. International Journal of Software Engineering & Applications (IJSEA), Vol.3, No.1, pp. 37-45.
- [5] Agile Alliance, Manifesto for Agile Software Development. 2001. retrieved from Agile Alliance: <http://www.agilemanifesto.org>.
- [6] Mike Cohn. 2004. User Stories Applied: For Agile Software Development. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA.
- [7] K. Bittner. 2008. The Requirements Landscape. An Ivar Jacobson International White Paper.
- [8] K. Holtzblatt and H. R. Beyer. 1995. Requirements gathering: The human factor. Communications of the ACM, Vol. 38, No. 5, pp. 31-32.
- [9] D. Zowghi and C. Coulin. 2005. Requirements Elicitation: A Survey of Techniques, Approaches, and Tools: Engineering and Managing Software Requirements, pp. 19-46.
- [10] R. Hartson and P. Pyla 2012. The UX Book. Morgan Kaufmann , Burlington, MA.
- [11] H. R. Beyer and K. Holtzblatt. 1997. Contextual Design: Defining Customer-Centered Systems. Morgan Kaufmann. San Francisco, CA.
- [12] J. Nielsen, Paper Prototyping: Getting User Data Before You Code. retrieved from Jakob Nielsen's website: <http://www.useit.com/alertbox/20030414.html>
- [13] United States Geographic Survey, <http://www.usgs.gov/>
- [14] Google Inc., 2012, Google Earth (Version 6.2), 2012.

## Authors

Dr. Ozgur Aktunc is an Assistant Professor of Software Engineering at St. Mary's University Engineering Department. Dr. Aktunc received his Ph.D. in Computer Engineering from the University of Alabama at Birmingham in 2007. Dr. Aktunc's main research area is Software Engineering with an emphasis on component-based software development. In particular his research relates to software analysis, testing of web applications, and software metrics. He also performs outreach activities to improve computing education in San Antonio through summer programs for high school students.



Berat Alper Erol received his M.S. in Software Engineering from St. Mary's University, San Antonio in 2012. He has a B.S. degree in mathematics from Kocaeli University, Turkey. He is currently a senior student in Aeronautical Engineering at Istanbul Technical University, while he is applying to pursue a Ph.D. degree in Computer Science or engineering programs in the US. His research interests include software engineering, requirements engineering, contextual design, user interface design, and usability.



John D. Garcia is a software developer at General Dynamic C4 Systems. He received his M.S. in Computer Science from St. Mary's University, San Antonio in 2012. He has over 15 years' experience in software development. He is currently working on trusted software applications to operate in a multi-level operating system or on a trusted web server.

