Proposal for Team TerraUser

The Web-based User Management Project

Michelle Harr
Naoko Tsunekawa
Daniel Wallace

20 December 2001

TABLE OF CONTENTS

LIST OF TABLES AND FIGURES

1. INTRODUCTION

This document is an official proposal of the TerraUser web-based user management software. This project is part of Northern Arizona University (NAU) College of Engineering and Technology's (CET) Senior Capstone Design 2001-2002, with sponsorship provided by Deborah Lee Soltesz from the U.S. Geological Survey and advisement provided by Dr. Eck Doerry, Professor of Computer Science at NAU.

Our client is Deborah Lee Soltesz from US Geological Survey (USGS) Terrestrial Remote Sensing Group at the Flagstaff Field Center. The team has set up TerraWeb as a way for people to access this information along with a way to organize and manage some of their data. However, currently USGS TerraWeb applications have minimal security. Users are not required to log on to access these web applications. No current user management system is in place. Data management and data analysis and manipulation is the main function of many of these applications, and it is imperative that if work is going to be done using these systems that there be some sort of security standards.

The objective of the project is to design and implement an efficient and secure interface to other USGS TerraWeb applications, along with a stand-alone application used to administer the user management system. The software will allow users to securely and easily access other interactive TerraWeb applications.

This document provides the problem statement and solution statement of the product, along with requirements and specifications, cost analysis, risk assessment, organization and project management, high-level design, and resources, schedule, and budget.

2.1 Background / History

Congress created the U.S. Geological Survey in 1879 as a science agency for the Department of the Interior. The USGS serves as a national science provider and fact-finding agency that provides a scientific understanding about natural resource conditions, issues, and problems. USGS scientists collect, monitor and analyze large amounts of data about the Earth and solar system. The government and citizens in all walks of life use the information the USGS produces for various reasons including to address pressing social issues. The USGS uses the vast scientific expertise for a wide range of products such as maps and scientific solutions such as wetlands restoration and hazardous waste disposal.

The USGS Terrestrial Remote Sensing Team at the Flagstaff Field Center consists of a four-member group: Pat Chavez *(Remote Sensing Scientist and Group Leader),* Stuart Sides *(Computer Scientist),* Deborah Lee Soltesz *(Web Mistress), and* Miguel Velasco *(Image Processing Specialist)*. They work with satellite multispectral, airborne photos, shipborne sidescan sonar, and DEM digital images. This team does such things as digital mosaicking, extraction and mapping of earth science information, geometric and radiometric calibration and corrections, and multitemporal change detection. The team has set up TerraWeb as a way for people to access this information along with a way to organize and manage some of their data.

Currently USGS TerraWeb applications have minimal security. Users are not required to log on to access these web applications. No current user management system is in place. Data management and data analysis/manipulation is the main function of many of these applications, and it is imperative that if work is going to be done using these systems that there be some sort of security standards. These TerraWeb applications are fairly new, therefore application uses and functions are evolving for the groups needs.

2.2 Business Issues

- Security is the biggest business issue that is faced in this project.
- There needs to be a user-friendly interface.
- There will be NO browser specific tags, and NO cookies.
- The project must comply with the Rehabilitation Act of 1973, Amendments of 1998, section 508.
- Since the project is all web based it is important that the server is accessible and fast enough to handle a given number of users.

Some of the other important business issues related to this project are how to allow access to multiple web-applications, and how to effectively manage users and the data that they have access to.

2.3 Value of a Technology Solution

This project will facilitate/support a way for users to securely and easily have access to TerraWeb applications. The client will have a generic interface to all of the current and future web applications. The TerraUser solution will be a cost effective and easily modifiable solution to current and future needs. It does not have the extreme cost or confusing complexity of commercially available solutions to this problem.

2.4 Competitive Products

At the moment we are not aware of competitive products that meets the specialized needs of the project.

2.5 Business Environment

Since the technologies that the project is being developed on are cutting edge, there are always going to be changes to the solution, thus we expect changes in the requirements. Since this project is designed as an interface to applications that deal with scientific data management and analysis, we will use a modular design. User expectations are always high when it comes to security and user management. There is a demand for our product in many areas within the user market and beyond.

2.6 Assumptions and Dependencies

Listed below are assumptions that have been considered.

- All TerraWeb applications will be using same technologies so that the interfacing will not be a problem.
- A server exists.
- Users have access to standard browser.
- Direct access to server is available.
- Server has direct access to TerraWeb server (i.e. not going through firewalls).

3.1 Goals

In this project we are to design and implement a system that will:

- Create a secure interface to currently existing web applications
- Centralize the user management system
- Provide a way for different users to have different access levels
- Set different user priority levels
- Have an application to manage all the users
- Allow for basic user customizations

The objective of the project is to design and implement an efficient, secure interface to other USGS TerraWeb applications, along with a stand-alone application used to administer the user management system. The software will allow users to securely and easily access other interactive TerraWeb applications.

3.2 Product Description

There will be two parts to this TerraUser interactive web application: a stand-alone part for administrators to manage users and permissions, and an invisible application that lets other applications connect to get specific user information. There will be a variety of information about the user that will be stored including, but not limited to:

- Who the user is
- What the user's personal preferences are (look and feel of application)
- What team user belongs to
- Priority level for running processes
- Applications the user has access to
- Level of access

The system will be built on a SuSE Linux server running an Apache web server. The system will use a MySQL database to store user information in.

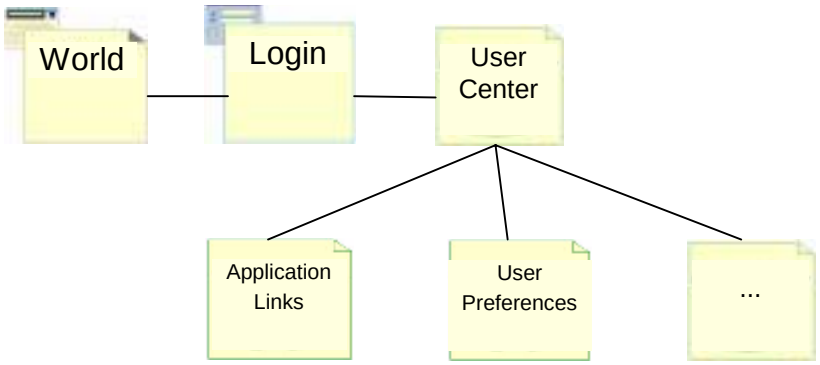

Figure 3.1: High-level Interface Overview

Figure 3.1. Above shows a rough diagram of the TerraUser interface. Users will have to log in through a browser. They will then have access to TerraWeb applications, a user preference page, etc.

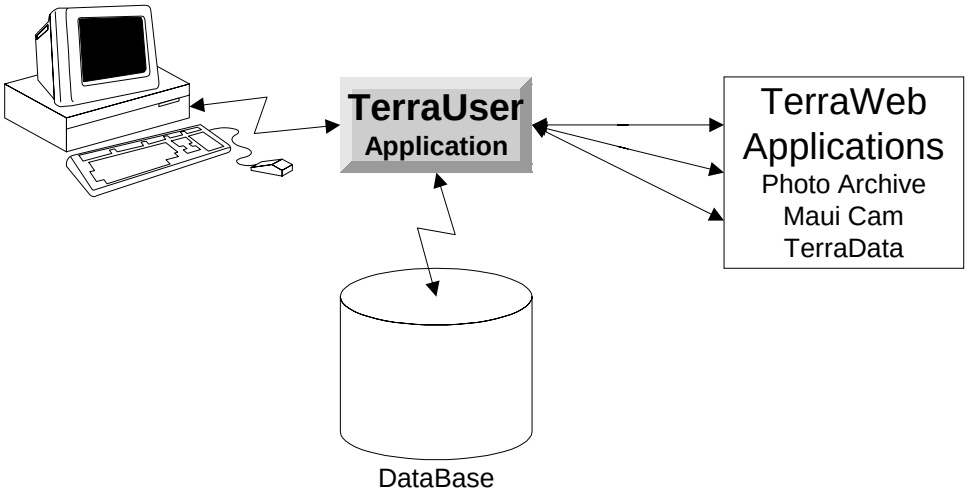

Figure 3.2: High-level Application Overview

Figure 3.2. Above shows a rough overview of the TerraUser application. Users will use a secure socket layer to login to the TerraUser application. The TerraUser application will communicate to a MySQL database using JDBC. The TerraUser application will be able to send requested information to the TerraWeb applications.

High-level architecture will be discussed in section 9 with more details.

4.1 Functional Requirements

The overall functionality must:

- Provide a secure interface for users to login to TerraWeb applications
- Provide a centralized way to manage users and their access/priority level
- Use a web-based interface
- Allow the user to:
  - Supply a user name and password to be granted access to applications
  - Supply personal preferences
- Allow the administrator to specify users, access levels, priority levels, and available applications
- Store all the data in its database

The following specific functions must be provided:

1. **User Accounts:**
   - Shall provide storage of user information including but not limited to: user login name, password, priority level, access rights, group membership, and interface preferences.
   - Will be administered through an administrator interface.

2. **Centralized User Login:**
   - Will require use of user login name and password to access the software.
   - Shall provide users the ability to change the password at any time.
   - Shall use encryption to send data.
   - May have expiration times for passwords set by administrators and will force users to change their passwords after the expiration time.

3. **Interactive Web Application for Administrators:**
   - Will be an independent application available exclusively to administrators.
   - Shall provide the ability to add or delete information that can be stored for users.
   - Shall provide the ability to alter information stored for any given user.
   - Shall provide the ability to add or delete users.
   - Shall provide options to set the password expiration time.
   - Shall provide monitoring of user activities and the option to enable outputting of activities to a log file.

4. **Interactive Web Application for Users:**
    - Shall use secure login.
    - Shall provide access to the Maui Cam, TerraData, and Photo Archive applications.
    - Shall provide user interface customization stored with their user data.
    - Shall allow applications to retrieve the user's customization.
    - Shall allow users to view and manage their system accounts in one place.
    - Shall allow users to use TerraWeb applications they have been granted access to.
    - Shall allow users the option to change their password.

4.1.1 Functional Client-side requirements

There are three kinds of users:

- *Administrators:* Users who use the administration application to edit user information in the database. They enter data such as who the user is, what privileges the user has and the priority level of the user.
- *Editors:* Users belonging to a specific work group or multiple work groups who have access to all information belonging to their group. These users also have read access to information that is marked public by other groups.
- *Guests:* Users who are allowed very limited access and information to applications. These users can only search and read information that is marked as public.

Administrators

Administrators manage users and teams and what information they can access. The following table is a list of the identified requirements for the administrators.

Requirements prioritized with 1 = Crucial, 2 = Very high, 3 = High, 4 = Average, 5 = Low, 6 = Very low.
Difficulty is rated from 1 to 10 where 10 = very difficult and 1 = very easy.

Requirement	Priority	Difficulty
Add a new user	1	2
Delete existing user	1	2
Update user information	1	2
Add/Update/Delete team information	1	2
Log off all users	3	8
Post Message of the Day (MOTD)	5	1
View active users or logs	2	2
Add information fields	1	4
Set password reset/expire	1	5

Table 4.1: Administrator Requirements

Editors

Editors are the main users of the interface and applications. These users do the actual work. The following table is a list of the identified requirements for the editor.

Requirements prioritized with 1 = Crucial, 2 = Very high, 3 = High, 4 = Average, 5 = Low, 6 = Very low.
Difficulty is rated from 1 to 10 where 10 = very difficult and 1 = very easy.

Requirement	Priority	Difficulty
Change password	1	2
Change preference	1	4
Access to applications	1	8
Search option	5	8
E-mail to administrator	2	1

Table 4.2: Editor Requirements

Guest

Guests are just able to view certain information and have limited access to applications. The following table is a list of the identified requirements for the Guest.

Requirements prioritized with 1 = Crucial, 2 = Very high, 3 = High, 4 = Average, 5 = Low, 6 = Very low.
Difficulty is rated from 1 to 10 where 10 = very difficult and 1 = very easy.

Requirement	Priority	Difficulty
View application data that is marked public	1	8
Search for public information	1	5
Post messages to administrators or teams	5	1

Table 4.3: Guest Requirements

4.1.2 External Interface Requirements

User Interfaces

The TerraUser software will be accessed on the Internet through a web-browser. Users must have an Internet connection and a standard browser to access the software.

Hardware Interfaces

The TerraUser software is a database driven web-application. The software will use the most current standards for implementing JSP pages accessing a MySQL database via JDBC. All HTTP transfers (data over the internet) and SQL access (data from a local database) will be handled by internal functions of the technologies being used.

Software Interfaces

The TerraUser software will act as an interface between the user and applications they have access to. The applications can be only be accessed through the software. When the user selects an application to run, the software will send required information. At any time the application is running, a request from the application can be made back to the software to gain any new or different information needed. The request can be made from any of the applications. A single feature in the software will handle all requests. When new applications need to interface to the software, the form of the request will be used in the new application and the software will be ready for the new application. The requests for information can be requested only with valid user login.

Communications Interface

The TerraUser software will use a web-based interface. The software will generate dynamic web pages viewed via HTTP/HTTPS. The software will communicate with the users web browser using the most current HTTP/HTTPS and HTML standards. All communication from the software to the user will be prompted by user input. All communication from the user will be done through entering data and selecting items on the web pages dynamically created by the software.

4.2 Performance Requirements

1. System development must use the technical tools/languages as specified by USGS sponsor. See section 3.5 for technical constraints.
2. System must be user-friendly for non-technical users.
    - The system will use a simple web-based interface.
    - It shall be accessible by all standard methods of Internet access.
    - It should require little or no training to use.
3. System must be maintainable.
    - The code will be well commented.
    - Documentation will be created about the software design and implementation.
    - System should be accessible from any machine.
    - The system will reside on a network visible to the Internet.
4. System must be highly secured.
    - The system will enforce user login, which will determine level of accessibility.
    - It shall use encryption to send information across networks.
5. System must be scalable.
    - The system will use a modular design.
    - The system will use a simple and well-documented interface between modules and external software.

4.3 Non-Functional Requirements

These are constraints on the services that do not impact the functionality of the product. The system must follow design guidelines presented in the capstone design course. The system must be designed using technical constraints listed in table 4.4 in section 4.4.2

Delivery

- Documents in electronic forms: PDF, HTML, or MS Word.
- Code should be delivered well commented in electronic form.

Training

- Users should require little to no training to use interface.
- Administrators should require at most an hour worth of training to use application.
- Application developers that want to use interface should require less than an hour of training to interface (Reading documentation should be enough).

Usability

- User logins should take no longer than 30 seconds.
- Access to applications should be seamless.
- Errors should give a friendly error message and direct the user toward help.
- Basic help should be available.
- Users can only be logged in once.
- Messages will be displayed when server is unavailable for logins.

4.4 Design and Implementation Constraints

Below are the constraints that have been proposed by the client as well as those which reflect project domain specifications.

4.4.1 General security constraint

The system will have to secure user information sent through the Internet. This will be achieved by using the secure HTTPS protocol.

4.4.2 Technical constraints

The TerraUser software must meet the following minimal requirements:

- The system will be designed to be scalable to meet future needs of the client.
- The implementation must utilize specific technologies provided on the server.

The following is a brief summary of the technology required by the project and available on the server:

Category	Technology Used
Operating System	SuSE Linux
Web Server	Apache
Java Server	Apache Tomcat
Server Side Interfacing	Java, JDBC, JSP, JavaScript
Database	MySQL
Security	SSL

Table 4.4: Technical Requirements

The design must provide a completely web-based interface. All interfaces must meet with HTML 4.0 minimum standards and be in compliance with the Rehabilitation Act of 1973, Amendments of 1998, section 508.

4.5 Business Rules

1. System must comply with the Rehabilitation Act of 1973, Amendments of 1998, Section 508
2. System must adhere to accessibility and government guidelines (System must not use cookies, etc)
3. System must not require specific browser to be run.
4. System development/integration/testing must be completed by April 26, 2002, for the Capstone Project Conference.

4.6 User Documentation

1. All the documentation must be in PDF, HTML, or Microsoft Word.
2. All deliverable documentation should be submitted electronically.
3. Documentation and reports should be formatted so they will make usable two-sided hardcopies.
4. Documentation should be readable on a computer screen.
5. Documentation on setup/implementation of system, database, and security must be provided.
6. Code must be well commented.
7. Manuals for programmers, system/database/web administrators must be provided.
8. Online, context-sensitive help must be provided.

5.1 Estimated Life Cycle Costs

Technology changes are very fast, and it is significant to maintain and update the product. The product development is based on the assumptions listed in section 1.6. The assumption is made for current technology, and it is possible that the assumption may not fit future technology. A summary of estimated cost for maintenance over the next 5 years is in Table 5.1 below.

Lists	Time	Estimated Cost **
Maintenance cost for administrators hours	Ξ 6 hours	$150
Operating System change	Ξ 5 hours	$125
New version of Java programming language	Ξ 2 hours	$50
Security standard change	Ξ 20 hours	$500
Web server change	Ξ 1 hours	$25
Database change (updates)	Ξ 2.5 hours	$62.5
Total	Ξ 36.5 hours	$912.5

*Note: Costs are estimated based on $40K/yr+30% overhead (= $25/hr) Estimated costs are basically for man-hours since updates are free.

Table 5.1: Estimated Cost for Maintenance

5.2 Benefits

There are many benefits that this software will bring. The following list is some of them.

- Professional look and feel of application/database access
- Flexibility (customization)
- Security of network, database, and application access
- Easy use of applications
- Easy database management

This section characterizes the possible risks during the design, implementation, and testing phase in three different categories: project risks, product risks, and business risks. The risks are also analyzed for their probability and overall effect to identify possible strategies for success of the project. Risk analysis will help us to focus on events that have a reasonable chance of occurring and that would directly affect the success of our project.

6.1 Risks

After analyzing the problem, Team TerraUser had identified the following as being potential risks.

6.1.1 Project Risks

Project risks are risks that affect the projects resources or schedule.

#	Risk	Risk Description
1.	Time Management	Group members all have different and busy schedules so, finding time to meet might be difficult.
2.	Hardware Issues	Hardware, which is essential for the project, might fail or be unavailable.
3.	Requirements Change	Numerous changes on the requirements that were not anticipated might occur.
4.	Security	There might be changing security standards in the field.
5.	Compatibility	There might be compatibility issues with the web server, operating system, browser, and database.
6.	Interface Specification Availability	Essential interface specifications not available on schedule.
7.	Size Underestimation	We might underestimate the size and complexity of the system.
8.	Lack of Experience	There is a lot of technology that group members need to learn and master in order to successfully complete the project.

9.	Major Form of Communication Blocked	Example: Thursday December 6th, 2001 we were unable to reach our sponsor by email because of the USGS network being cut off from the rest of the world. The whole of the US Department of the Interior had been forced off of the Internet as a result of a court case Cobell v. Babbit. The BLM, USGS and Park Service were also offline.
10.	Management Change	There might be an organizational change where the sponsor's management changes.
11.	Interface Specification	There might be different specifications interfacing to a variety of USGS TerraWeb applications.
12.	Security	There might be security issues with bugs, viruses, hackers.
13.	Speed Issue	The network cannot process as many transactions in a reasonable speed as expected.
14.	Flexibility	The product is not flexible enough for future modification.
15.	Bugs	There might be an overwhelming amount of bugs found in software.

Table 6.1: Project Risks

6.1.2 Product Risks

Product risks cover risks related to the products that we rely on to do this project. Some of the products that we are relying on to do this project are: SuSE Linux, Apache Web Server, Apache Tomcat, MySQL, SSL, Java, JDBC, HTML, JSP, Java Script.

#	Risk	Risk Description
16.	Security	Security hole in products we rely on to do project.
17.	Bug or Missing Functionality	Hard to implement a specific feature because of a bug in one of the products that we rely on.
18.	Changes in Software	It takes time to do upgrades when changes are made, and you don't know if everything will upgrade smoothly, could be compatibility issues also.
29.	Browser Incompatibilities	Inconsistency between how different browsers function.

Table 6.2: Product Risks

6.1.3 Business Risks

#	Risk	Risk Description
20.	Changes in Technology	Some new technology might come out that supersedes the technology on which the system is built.
21.	Product Competition	Product might already exist, or be developed.
22.	Data Management	Different TerraWeb applications might need TerraUser to keep track of some unique piece of information.

Table 6.3: Business Risks

6.2 Risk Analysis

The risks we have identified are the most significant and threatening of the many risks the project may face. To establish a risk assessment framework we have projected the probability of each risk happening and the overall effect of the problem actually occurring. The following table shows the major risks along with the probability of each happening and the corresponding effect.

Risk	Probability	Effects
Time to develop software is underestimated.	High	Serious
Learning curve is high on required technology.	Moderate	Tolerable
The size of the software is underestimated.	High	Tolerable
The network or database or server cannot process as many transactions as expected.	Low	Catastrophic
Cannot perform open searches on database.	Low	Serious
Unexpectedly high number of bugs to fix during test phase.	Moderate	Tolerable
Key team members are unavailable or ill at critical time in project.	Moderate	Serious
Changes to the requirements are made causing major design and implementation changes.	Moderate	Serious

Table 6.4: Risk Probability & Effect

The only critical risk we have identified is the underestimation of development time. We are confident, however, that this risk will be closely monitored and that the team is devoted to overcoming any obstacle to prevent it from happening.

6.3 Risk Management and Mitigation

For each of the risks identified in sections 2.1–2.3 possible strategies have been identified to manage these risks. Solutions to risks follow:

#	Risk	**Risk Analysis/Mitigation**
1.	Time Management (Underestimated development time)	Work harder on prioritizing important tasks to be completed.
2.	Hardware Failure	Could lose valuable data and code if there was a major hardware failure on server. Make a back up plan of critical files and document the setup.
3.	Requirements Change	Continuous communication with sponsor and all design team members.
4.	Security	Read security news sites, to keep up to date.
5.	Compatibility	Pick a standard to support and stick to it.
6.	Interface Specification Availability	Have a back up plan of how to solve interface problems.
7.	Size Underestimation	Prioritize the most important things to be completed during the time allotted.
8.	Lack of Experience	Team members will research and try to get up to date on interfacing issues.
9.	Major Form of Communication Blocked	It is hard to know ahead of time whether a major network problem or communication breakdown will happen, so always have more than one way to communicate and a back up plan.
10.	Management Change	Politics are part of business life, so expect a reasonable amount of change and be able to adapt.
11.	Interface Specification	Research alternatives, in case the path chosen fails to work.
12.	Security	Make sure that the latest patches are installed.
13.	Speed Issue	During testing look at the performance.
14.	Flexibility	Project will be designed using a modular design.
15.	Bugs	If testing of software is done consistently and intensely through out the process, major bugs can be minimized and avoided.
16.	Security	Make sure that all the latest patches are applied.
17.	Bug or Missing Functionality	Check for latest patches, or check web for a work around.
18.	Changes in Software	Add some extra time to the schedule for things such as upgrades.
19.	Browser Incompatibilities	Decide on some sort of standards and stick to them.

20.	Changes in Technology	By making sure the system has the latest and greatest, this risk should be minimized.
21.	Product Competition	Do research on competitive products.
22.	Data management	Sometimes data management can get messy and confusing. One way to avoid this is to follow guidelines and standards set forth by the industry.

Table 6.5: Risk Mitigation Strategies

By monitoring the risks listed above we hope to minimize their occurrence. Mitigation strategies are listed in section 4.2, should we encounter these risks.

6.3.2 Risk Monitoring Strategies

Throughout the project the risks will be reassessed and updates will be made to risk mitigation strategies. The risks will be discussed during periodic project reviews and status reports.

Risk monitoring will consist of the following:

- **Risk 1:** To monitor how we are doing with regards to the schedule, we will look at the current schedule and calendar and make updates as needed.
- **Risk 2:** By constant use of development box we will know if a hardware failure occurs. To be safe we will make a weekly backup.
- **Risk 3:** With constant communication with sponsor and group members we will keep on top of changes that are made in the requirements.
- **Risk 4:** We will do monitoring of security news websites weekly.
- **Risk 5:** By configuring and using required project technologies, we will find compatibility issues that arise.
- **Risk 6:** We will place a high priority level on interface specification and allocate a few extra resources, and checking the schedule regularly.
- **Risk 7:** We will check the complexity and progress of the project during weekly status reports.
- **Risk 8:** To keep up to date on what each team member is doing, each team member will write a weekly status report and email it to the team.
- **Risk 9:** We will use constant communication through many channels.
- **Risk 10:** This risk can't be monitored until it happens.
- **Risk 11:** We will monitor interface through progress reports and schedule.
- **Risk 12:** Creating a log of events that occur will help with security monitoring.
- **Risk 13:** During the testing cycle, performance testing can be done.
- **Risk 14:** There will be intensive testing and checking during the review process.
- **Risk 15:** To keep track of bugs and software problems, we will use a bug track log and then have other group members verify that problem is corrected after the bug fix. We will test related functionality to make sure other things did not break once the bug fix was applied. Verifications will be logged too.

- **Risk 16:** Logging events that occur both on the web server and in the application can help track down security problems.
- **Risk 17:** We will do a weekly check of product websites for patches and make sure that everything is up to date.
- **Risk 18:** We will add some extra time to the schedule for such things as upgrades.
- **Risk 19:**To track changes in the technologies that we are using, weekly monitoring of technology news websites will be done.
- **Risk 20:** Individuals will educate themselves on what is going on in this field.
- **Risk 21:** We will do research at the beginning of the project on competitive products, and keep an eye out for things that we may have overlooked.

6.4 Summary

Through detailed documentation, constant communication, quality tracking and constant research we hope to be able to avoid most of the identified risks.

When we identified the risks, we recognized that time management is significant since the project must follow the college capstone curriculum timeline. Also, changes in technology are fast and it is important to be aware of and to keep up with them. Security issues are always a major issue in computer networking, and we should have a good knowledge of it to be confident with our product.

By identifying the possible risks associated with the project and ways for minimizing or mitigating them we feel confident that we will not have any major surprises. Although we have identified many risks we feel better prepared to deal with problems should they arise. The overall project seems to have a moderate to low risk associated with it.

7.1 Team Organization

The TerraUser team is composed of three members, Michelle Harr, Naoko Tsunekawa, and Daniel Wallace. All of the team members are seniors at Northern Arizona University in the Computer Science and Engineering program. Assigned responsibilities of the team are as follows:

- Michelle - Team Leader / Communications Coordinator
- Daniel - Website Coordinator and Facilitator
- Naoko - Document Coordinator and Secretary

Figure 3 shows the TerraUser organization.

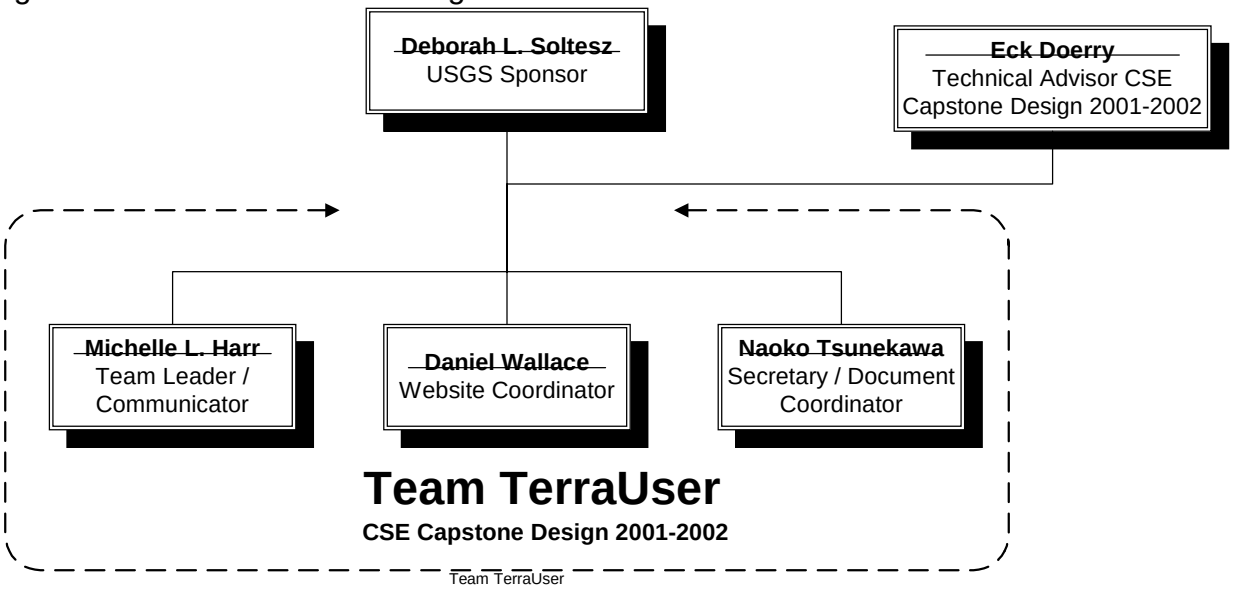

Figure 7.1: TerraUser Organizational Chart

7.2 Project Roles

Each member will focus on specific categories on the project. Project roles are as follows:

- Michelle – MySQL, JDBC
- Daniel – JSP, HTML, CSS
- Naoko – SSL, Web-Server Administrator

7.3 Project Management

Each group member is in charge of managing a certain aspect of the project as defined above in section 7.2. These group members are in charge of dividing tasks and checking on the progress of the tasks. The team leader will keep track of progress and if progress is lacking or not being made, will try to correct the problem.

7.3.1 Reporting Process

Meeting Minutes

Meeting minutes will be recorded by secretary and distributed via e-mail no later than 48 hours after the meeting. Meeting minutes should include the items discussed at the meeting and action items to be taken.

Individual Weekly Status Report

An individual weekly status report will be sent to the group leader every Friday by 4pm. It should include the items completed this week and the tasks for the next week.

Weekly Status Report

A weekly status report will be distributed to sponsors and group members every Friday by 5pm via e-mail with following categories:

- Assignment completed
- Other tasks to be completed this week
- Meeting held this week
- Tasks for next week (action items are also in meeting minutes)

7.3.2 Monitoring Process

Project will be monitored in following categories:

- Schedules and timelines – updated weekly; make changes as needed
- Risks – see section 6.3.2
- Group members must be informed of changes through e-mail
- Sponsor contact log must be updated regularly to monitor the communication with sponsor

7.3.3 Review Process

Documents Review

All document drafts must be completed in time for a team review before the final versions are submitted. Documents must be reviewed and approved by all individuals before submission. Individual components must be delivered to the coordinator at least a week in advance of the due date. The team will then review and finalize the compiled document.

Self-Evaluation

A self-evaluation allows us to reflect upon and examine our work. Following should be evaluated:

- What were our goals going into the project?
- Did the work move us toward these goals?
- Did our goals change during this experience?
- How can we improve the process for next time?

A self-evaluation will be conducted at the regular meetings following a major assignment deadlines. We will write up a brief summary of areas examined, problems identified, and steps taken/planned to correct the problems. Evaluation results will be typed and posted on the team website and team notebook.

Individual Reviews

To evaluate individual performance so we can improve our teamwork, all group members will do an individual review each semester.

7.3.4 Team Decision Process

Decisions will be made by the following process:

1. Group member makes motion
2. Discussion on item
3. Majority vote

For major design changes to be made, they need to be discussed, debated and voted on by the group. A team member can make minor design changes for the project component they are working on as long as they inform the other group members of the changes at meetings or by e-mails. This is assuming that the changes will not have any major impact on the other group member's work.

In a situation of a divided team, non-participating members, team members who change the design without team consent, etc, the conflict resolution strategies must be taken to solve the problems. The process is as follows:

1. Try to communicate with the member who is creating interpersonal disputes
2. Communicate with facilitator
3. Set group meeting(s) with manager

8.1 Design Methodology

This project is ideally suited for a modular design upon which different technologies are handled by individual modules. Additionally the technologies used are arranged in a hierarchical manner, which lends itself to a top-down or bottom-up design. We have a very clear understanding of the functions required of the system and feel a bottom-up design will prove itself effective and efficient. An evolutionary model will best support the level by level design in which we will be adding functionality of individual modules and then testing them. Section 8 goes into further detail about the architecture of the software.

8.2 Deliverables

The documentation that will be created for this project consists of four major documents. This document is the first and provides a solid base for the entire project design. The second major document is the functional specification document. That document will set the details for the implementation that will be used. After the design has been developed to a point where all functions are available, usability testing will be done. A usability report will follow that will summarize the results and offer suggestions and solutions to any problems. The final document is the as-built report that will cover the successful design of the project. The schedule for these documents is outlined in Table 10.3 of section 10.3. The standards for all delivered documents is available on the project web-site at http://www.cet.nau.edu/~dw2/terrauser/documents.html.

Throughout the implementation of the project, prototypes and proof of concepts will be provided to our client for review and feedback. In Table 8.1, the major milestones for the project are listed. For each of the milestones, a prototype of the functionality and/or integration will be made available to our client.

Milestone	Objective
Database Setup	Database created with structure required for user management.
Database Interface	Module implemented that handles all required database transactions to the web-interface.
Session Management	Reliable and secure user tracking system.
User Information Management	Module implemented that controls access/modification of user information through database interface.
Web-Interface Integration	Encapsulation of available functions in dynamically generated web pages.
Security Integration	Make internal transactions secure via encryption.
System Integration	Encapsulation of all desired functions in one package. Successful installation and run on desired server.

Table 8.1: Milestones

The architecture of the software will be fairly simple. Modules will be created at each level of technology used. The modules will act as an internal interface between each technology. The web-page modules use the database modules as an interface between the web pages and the database. This will be an effective and efficient way to provide easy interaction between the web-interface and the database.

A three-tier architecture can provide flexibility, reusability, and scalability when used for a distributed client/server design. It is a common architecture used for many Internet applications.

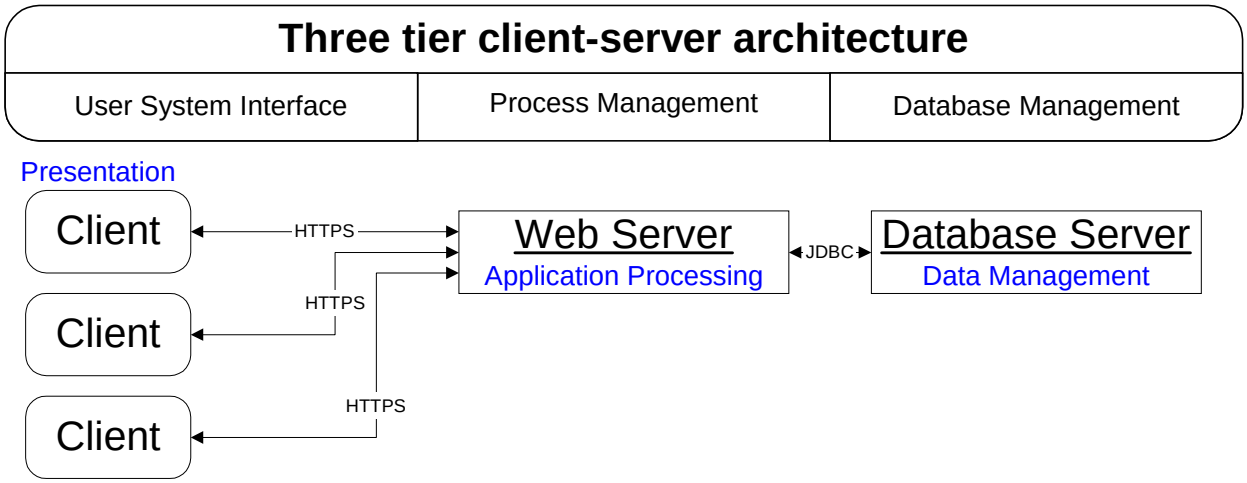

Figure 9.1: Architecture

By the use of a three-tier architecture it will allow information transfer between the web server and the database server to be optimized. The three-tier architecture allows for scalability. The architecture provides a framework for sub-system control and communication.

The following pages provide various diagrams of the software architecture. Since there are two interfaces to the software, one for administrators and on for regular users, the figures reflect the two different modes of access. State diagrams showing the nearly linear operation of the software is provided in Figure 9.2 and Figure 9.3. The object model diagram in Figure 9.4 shows the different modules. In Figure 9.5 and Figure 9.6, the data flow diagrams show the transfer of data between various functions of the software.

State Diagrams

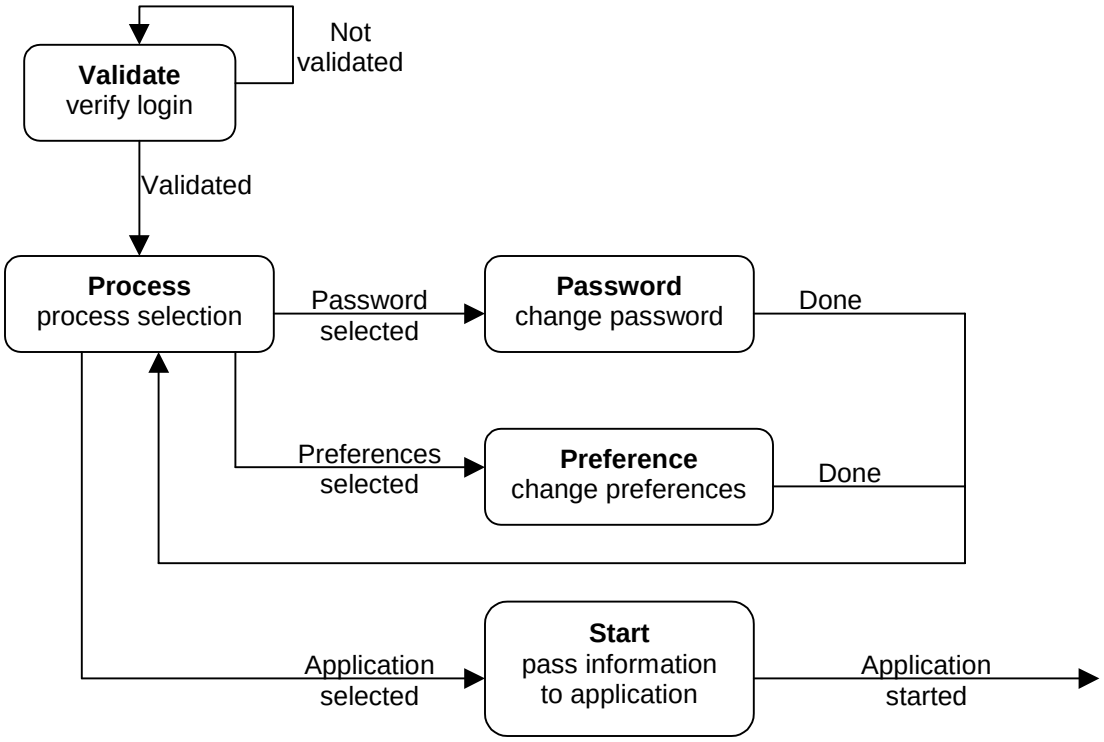

Figure 9.2 State Diagram Administrator Access

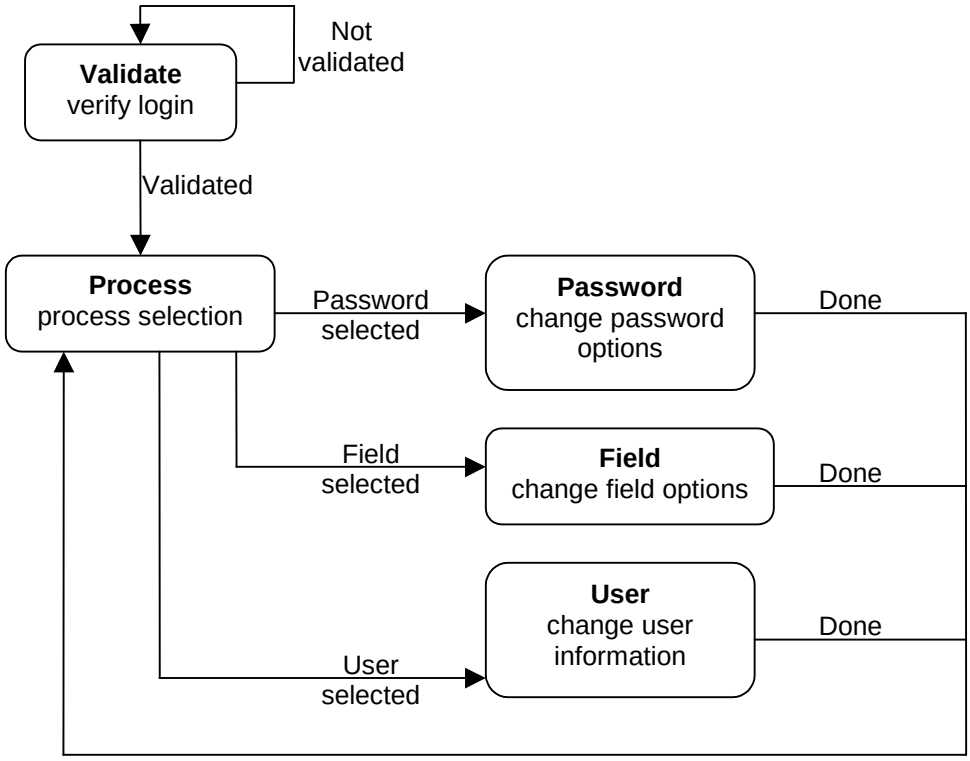

Figure 9.3 State Diagram User Access

Object Diagrams

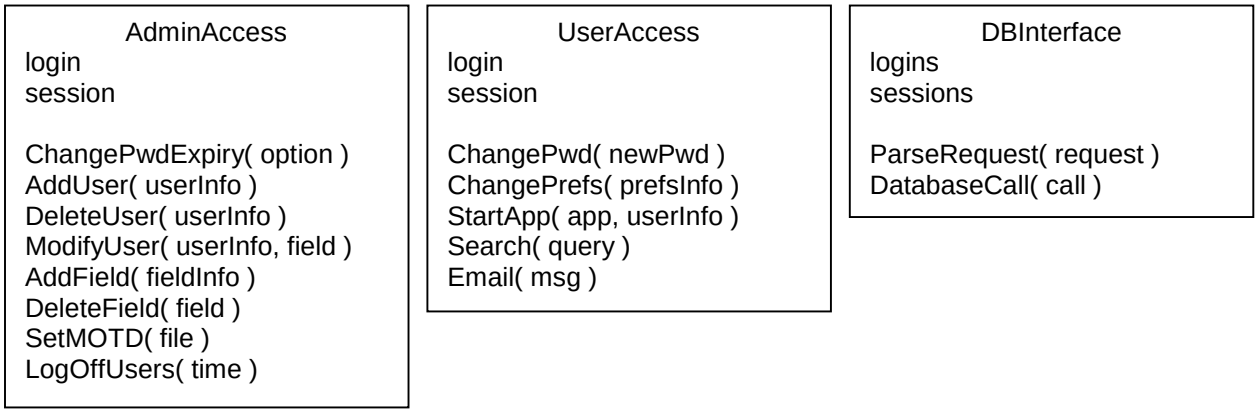

Figure 9.4 Object Models

Data Flow Diagrams

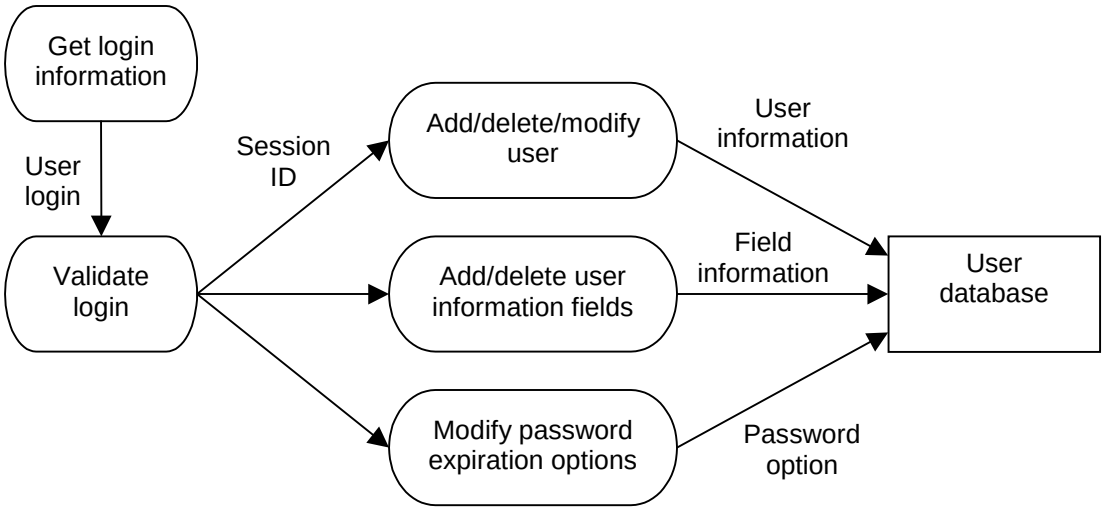

Figure 9.5 Administrator Access Data Flow Diagram

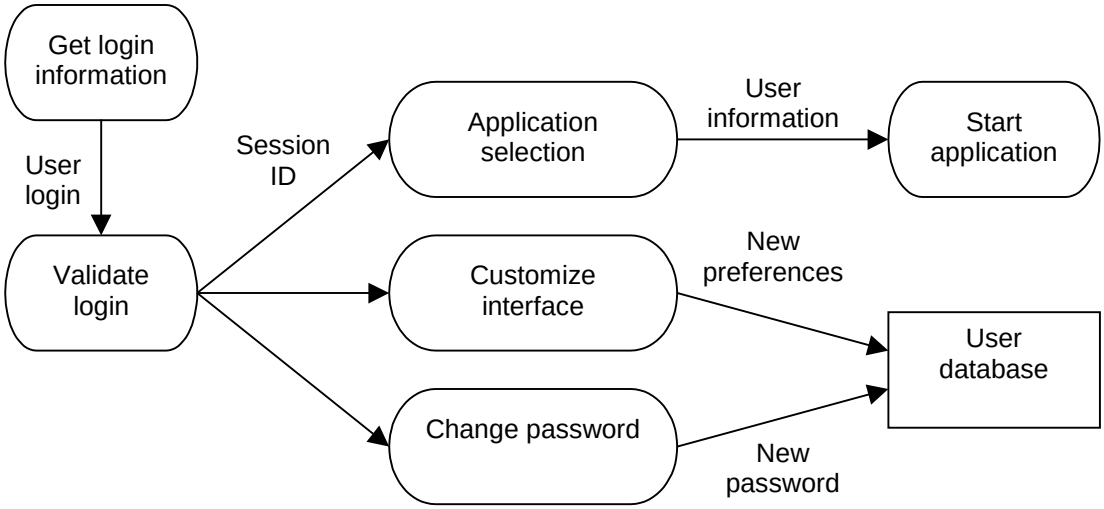

Figure 9.6 User Access Data Flow Diagram

10.1 Resources

Tools

See section 4.4.2 for technical constraints.

Hardware/Software Environment

Our sponsor has specified the environment our software will run in. The hardware the software is intended to run on is any computer running Linux with Apache, Tomcat, Java, and MySQL installed and running. See section 4.4.2 for technical constraints. This environment is already available for use by our team. Additionally, we have our own server with the required software for use.

Information available on the Internet about the resources is shown in Table 10.1.

Resource	Address
SuSE Linux	http://www.suse.com/index_us.html
Apache	http://www.apache.org/
Java	http://java.sun.com/
MySQL	http://www.mysql.org/ http://www.mysql.com/
Section 508	http://www.section508.gov/
Tomcat	http://jakarta.apache.org/tomcat/
USGS TerraWeb	http://terraweb.wr.usgs.gov/

Table 10.1: Resources

10.2 Budget

Estimated budget is shown in Table 10.2 below.

Items	Budget
Document Resource (books, etc)	$150
Printing / Copying	$100
Communication	Free (local telephone)
Travel	Negligible (local)
Hardware	Free (already available)
Software	Free (open source)
Network Communication	Free (NAU & USGS network)

Table 10.2: Estimated Budget

The standard capstone budget is around $500. Our team has come in under budget because of location and the nature of the project. Main costs are incurred from printing and books.

10.3 Schedule

The timeline is a significant process for a top-quality product to be created in a given time. Our timeline must follow the college capstone curriculum. An overall timeline for the project is shown in Table 10.3.

We will use Visio software to create Gantt charts to keep track of our tasks for this project. The chart will be updated weekly. Gantt charts of the project for Fall 2001 and Spring 2002 are shown in the appendix.

Timeline	Objective	Document / presentation
1st week of November	Initial sponsor contact Team forming stage	Team Inventory Team Bylaws Team Website
2nd week of November	Problem definitions / statements	
Mid. November	Initial requirements/specification acquisition, requirements document	Requirement document Introduction presentation
	Project analysis	Feasibility study document Risk assessment document
End of November	Draft proposal	
Early December	Complete specification	Proposal document Proposal presentation
During winter break	Learn and master technology that need on the project	N/A
Mid. January	Development detailed architecture	Functional Specification document
End of January	Functional Specification accepted by the sponsor	
Feb., 2002	Implementation	
March, 2002	Testing / Integration	Usability Document
April 26, 2002	Capstone Project Conference	As-built report / presentation

Table 10.3: Project Timelines

The first phase contains the team forming an initial sponsor contact. There the team members discussed their skills for the project. Team standards were discussed and documented. The first meeting with the sponsor was held to introduce the team members to the client and to discuss the project in more detail.

Requirements capture and documentation, feasibility study, risk analysis, and proposal documentation were in phase two. The project was analyzed from different angles to produce the best solutions. High-level software architecture has been created. Hardware/software setup is also an ongoing process in phase 2.

Over the winter break, the team members will learn the skills needed for the project before phase 3. Spring semester starts with phase 3, and the functional specification will be completed. After the functional specification is accepted by the client, detailed design and implementation take place. A detailed list of milestones for this phase is in Table 8.1 in section 8.2.

In phase 4, the product will be tested and integrated. The requirements and specification will be reviewed to examine if the product satisfies the clients' needs. Usability will be also checked. As the last stage, the Capstone Project Conference will be held on April 26, 2002. An as-built report and presentation will be given.

We are confident that our timelines are sufficient for our project to develop a useful, high-quality product. We allowed time between the semesters to gain knowledge of technology that we will need on the project. Implementation should start as soon as the spring semester starts to allow enough time for testing and integration. Our timeline also follows the college capstone curriculum. The project will be completed for the Capstone Project Conference.

11. CONCLUSION

The information provided in this proposal serves as a solid framework for developing the web-based user management package for US Geological Survey in Flagstaff. This team has analyzed the problem from different views, such as feasibility, risks, resources, schedule, and budget, and developed a high-level architecture of the product. After a number of studies, we found that the technologies are available to accomplish the tasks required by the product with low risks. The project is economically viable, and our product will be a valuable solution for our clients to manage data and TerraWeb applications with security. Team TerraUser is confident that we will design and develop a product in a successful manner.

We look forward to sharing our solution to benefit the USGS and improve their business.

TerraUser Timeline for Fall 2001

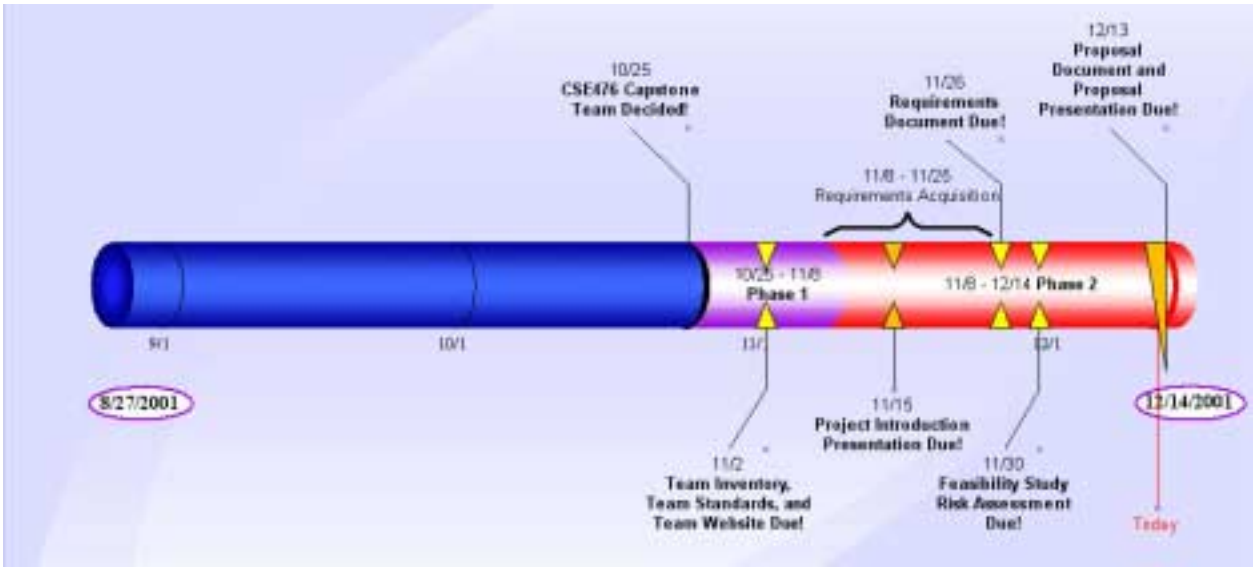

Figure 12.1: Fall Semester Timeline

Fall semester has been completed; dates on fall timeline are accurate.

TerraUser Timeline for Spring 2001

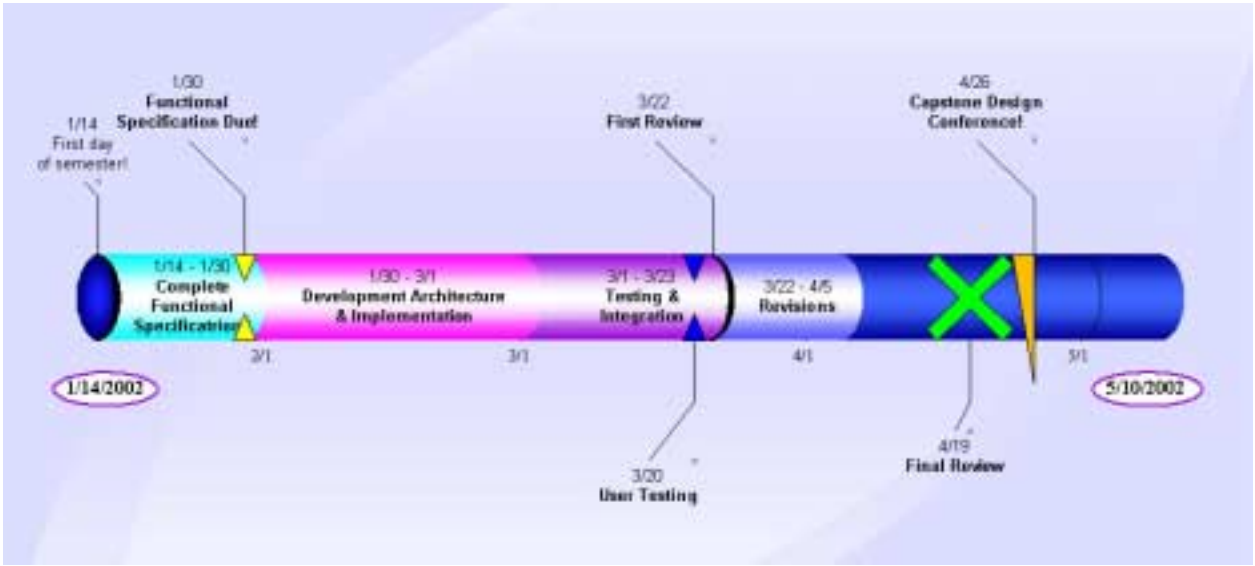

Figure 12.2: Spring Semester Timeline

Timeline for spring semester is a very rough estimate.

Document Conventions

Definitions, acronyms and abbreviations for a number of terms are provided because of formal and technical nature of this document.

Acceptance Test

A formal test conducted by the end user of a system, to determine if the system works according to specifications and should be accepted.

Apache

The Apache Project is a collaborative software development effort aimed at creating a robust, commercial-grade, featureful, and freely available source code implementation of an HTTP (Web) server. The project is jointly managed by a group of volunteers located around the world, using the Internet and the Web to communicate, plan, and develop the server and its related documentation. These volunteers are known as the Apache Group.

Authentication

Verification of identity as a security measure. Passwords and digital signatures are forms of authentication.

CGI

Common Gateway Interface - A way of interfacing computer programs with HTTP or WWW servers, so that a server can offer interactive sites instead of just static text and images.

CGI script

Common Gateway Interface script - A program that is run on a Web server, in response to input from a browser. The CGI script is the link between the server and a program running on the system; for example, database CGI scripts are used with interactive forms.

CSS

Cascading Style Sheets - style sheet mechanism that has been specifically developed for Web page designers and users. Style sheets describe how documents are presented on screens, in print, and even in spoken voice. Style sheets allow the user to change the appearance of hundreds of Web pages by changing just one file. A style sheet is made up of rules that tell a browser how to present a document. Numerous properties may be defined for an element; each property is given a value.

HTML

Hypertext Markup Language - The language used to create World Wide Web pages, with hyperlinks and markup for text formatting (different heading styles, bold, italic, numbered lists, insertion of images, etc.).

HTTP

Hypertext Transfer Protocol - The protocol most often used to transfer information from World Wide Web servers to browsers, which is why Web addresses begin with http://, also called Hypertext Transport Protocol. It conventionally uses port 80.

Java

A simple, object-oriented, distributed, interpreted, robust, secure, architecture-neutral, portable, multithreaded, dynamic, buzzword-compliant, general-purpose programming language developed by Sun Microsystems in 1995(?). Java supports programming for the Internet in the form of platform-independent Java "applets".

Interface

A shared boundary where two or more systems meet; or the means by which communication is achieved at this boundary. An interface can be between hardware and hardware (such as sockets and plugs, or electrical signals), hardware and software, software and software, human and computer (such as a mouse or keyboard and display screen).

JavaScript

A cross-platform WWW scripting language from Netscape Communications, very popular because it is simple and easy to learn. It can be included in an HTML file by using the tag <script language="JavaScript">.

JSP

JavaServer Pages - A freely available specification for extending the Java Servlet API to generate dynamic web pages on a web server. Industry leaders wrote the JSP specification as part of the Java development program.

JDBC

Java Database Connectivity Pages – Part of the Java Development Kit that defines an application programming interface for Java for standard SQL access to databases from java programs.

Linux

An Open Source implementation of UNIX created by Linus Torvalds, which runs on many different hardware platforms including Intel, Sparc, PowerPC, and Alpha Processors. Hundreds of application programs have been written for Linux, some of these by the GNU project. Linux and Linux tools can be downloaded via the Internet or BBS for free, or purchased as part of a distribution on a CD-ROM.

MOTD

Message of the Day.

MySQL

MySQL is a true multi-user, multi-threaded SQL (Structured Query Language) database server. SQL is the most popular database language in the world. MySQL is a client/server implementation that consists of a server daemon mysqld and many different client programs and libraries. The main goals of MySQL are speed, robustness and ease of use.

PDF

The file extension for a Portable Document Format file.

Perl

Perl is a general-purpose programming language invented in 1987 by Larry Wall. With over one million users worldwide, it has become the language of choice for World Wide Web development, text processing, Internet services, mail filtering, graphical programming, systems administration, and every other task requiring portable and easily developed solutions.

RCS

Revision Control System - A version control system that automates the storing, retrieval, logging, identification, and merging of revisions. RCS is useful for text that is revised frequently, for example programs, documentation, graphics, papers, and form letters.

SCCS

Source Code Control System - A popular code management system for Unix systems.

SSL

Secure Sockets Layer - A protocol from Netscape Communications Corporation, which is designed to provide secure communications on the Internet.

TerraWeb

Title of website and web applications that have been developed by Deborah L. Soltesz. These applications and web pages are designed to support a group of USGS scientists and computer scientists that are working on terrestrial remote sensing (http://terraweb.wr.usgs.gov/).

USGS

United States Geological Survey.

User

An individual who uses a computer, program, network, or related service for work or entertainment; usually there is a distinction between a user and a programmer or other person who works with the computer on an expert or technical level.

Definitions from: http://www.computeruser.com/resources/dictionary/