Concordia University
Department of Computer Science
and Software Engineering

Software Process
SOEN 341/4 --- Winter 2009 --- Section S

DEVELOPMENT ENVIRONMENT AND RESOURCES

Team members information	
Name	**SID**
Marc-André Moreau	9347100
Mathieu Dumais-Savard	6095275
Julia Lemire	9402969
Phuong-Anh-Vu Lai	5644399
Sébastien Parent-Charette	9178821
Andreas Eminidis	9377603
Corey Clayton	9349200
Eric Chan	9365079

TABLE OF CONTENTS

LIST OF FIGURES

No table of figures entries found.

1. INTRODUCTION

This document examines the resources used in the completion of this project. These resources can be split into two categories: human and technological. The human resources include everything from the availabilities to the skill sets of the team members. The technological resources include any software, languages, libraries and tools that may be used to implement the application.

2. SERVER SETUP

2.1 Assembla.com Team Collaboration Space

Assembla.com is a website that offers an integrated solution for online team collaboration. While various paid plans are available, a free public plan exists that provides enough resources for the scope of this project.

2.1.1 Team Collaboration

In order to manage a project, the team leader can create a new team collaboration space and invite each member to join it. This requires that each member of the team create an account on assembla.com. To simplify things, we have used our Concordia netnames as our account names. Through team collaboration space, members can easily reach the rest of the team through a messaging system and a chat function. There is also a place where one can view a history of the latest events.

2.1.2 Git Distributed Version Control System

One of the main reasons for using assembla.com's service is because they provide free version control system hosting. Version control can easily be added to the team collaboration space by the administrator with a one-click installation. Assembla.com provides multiple version control systems such as subversion, mercurial and git. While subversion is a popular version control system, the new trend is to move to git. Git is a more recent and efficient system originally created by Linus Torvalds. When compared to subversion, git is a faster and more efficient system, especially in regards to branching.

2.2.3 Webhook

Another nice feature that made assembla.com a good choice for hosting our git repository is webhooks. The webhook is a simple HTTP POST that is sent automatically to notify of an event. As we wanted to host our web application on a separate server, we are using it to trigger an update of the application on the remote server whenever someone commits changes on the git repository. Without webhooks, this would have to be done by constantly polling the git server for updates, which is a quite inefficient way of doing things.

2.2 OpenSolaris Server

OpenSolaris is a free open source operating system based on Sun's Solaris. It is one of the variants of UNIX based on SVR4. As Sun mostly ships Solaris with their selection of high performance servers of all types, OpenSolaris is a good choice for making our web server. For this project, a dedicated server with a P4 3.0GHz and 1 GB of RAM is used to host our web application in development.

2.2.1 Web Stack (AMP)

OpenSolaris' web stack, called the "AMP" stack, which stands for Apache, MySQL and PHP, provides an easy way to set up the various programs needed to host our web application. This is perfect for us since this is exactly what we need.

2.2.2 Development Web Stack (AMP-dev)

A web development stack is also available, providing a full set of tools to develop web applications. While it is not really an option for team members to use, since nobody uses OpenSolaris as their desktop OS, it comes with documentation and instructions on how to use the tools it installs. This is great as it provided a lot of ideas on what alternatives to use, such as the NetBeans IDE.

2.2.3 Hostname

The server is connected to the internet with a residential Videotron connection. Since Videotron uses the usual HTTP port 80, we are using the alternative port 8080. Also, since Videotron does not provide a static IP, we registered a free hostname from dyndns.org. It can be automatically updated to point to the current IP address of the server using a client program. Our website can be accessed at http://team7.ath.cx:8080/

2.2.4 Individual Test Space

Since we are eight people working on the same website, a separate space is provided to each member so they can do their own testing without breaking the work of other members. Each member has an account on the server with a username corresponding to their Concordia netname. Apache has been configured to enable the user directory features, which gives each member a web folder accessible at http://team7.ath.cx:8080/~username/. Each user's home folder contains a public_html folder where they can put their files for their personal web space.

3. DEVELOPER CLIENT SIDE SETUP

3.1 Git

3.1.1 Installation (Windows)

A windows port of git is available as the msysgit project, available at http://code.google.com/p/msysgit/. The installation is straightforward – just follow the installation wizard and a shortcut should be added on the desktop. The program gives a full bash shell with git installed.

3.1.1 Installation (Linux / Mac OS X)

On Linux, the distribution's package manager is able to install git. On Mac OS X, an installer is available at http://code.google.com/p/git-osx-installer/.

3.1.2 Initial Configuration

Log in to your assembla.com account, click on the Source/Git tab. Click on "More Instructions" under "clone/push URL". There are two URLs that can be used for git, one is for public read-only access, and the second one is for read/write access. As we want read/write access, we need to use git@git.assembla.com:team7.git. The first thing that you have to do is to set up ssh with a public key that you need to give to assembla.com so that they can identify you. If you already have a key, it should normally be placed in ~/.ssh/id_rsa.pub. If you do not have a key, you can generate a new one with "ssh-keygen –t rsa". The ida_rsa.pub file is just a text file containing your public key. Either output it to your terminal to copy it with "cat ~/.ssh/id_rsa.pub" or open it with a text editor. In assembla, click on "My Start Page" on top right hand corner and then click the "Profile" page. Scroll down the profile page, and there should be a space to paste your ssh public key. Paste it there and save your changes. It is important that this step is correctly done otherwise you will not be able to modify anything on the git repository.

Once the ssh keys are correctly set up, open a terminal (or in Windows, start the git bash shell) and then enter the command "git clone git@git.assembla.com:team7.git". The result should resemble Figure 3.1.2.1.

```
marcan_m@linux:~/$ git clone git@git.assembla.com:team7.git
Initialized empty Git repository in /home/marcan_m/team7/.git/
remote: Counting objects: 49, done.
remote: Compressing objects: 100% (44/44), done.
Receiving objects: 100% (49/49), 78.10 KiB | 10 KiB/s, done.
remote: Total 49 (delta 5), reused 0 (delta 0)
Resolving deltas: 100% (5/5), done.
```

Figure 3.1.2.1 – Result of proper git clone.

If you see something like this, you have successfully fetched a copy of the git repository with full history of the source code. Now switch to the newly created directory using "cd team7". Once you are inside the directory, you can start configuring settings specific to your own copy of the git repository. "git config -l" lists the current configuration. Figure 3.1.2.2 shows an example of this.

```
marcan_m@linux:~/team7$ git config -l
core.repositoryformatversion=0
core.filemode=true
core.bare=false
core.logallrefupdates=true
remote.origin.fetch=+refs/heads/*:refs/remotes/origin/*
remote.origin.url=git@git.assembla.com:team7.git
branch.master.remote=origin
branch.master.merge=refs/heads/master
```

Figure 3.1.2.2 – Example of a list of git configurations.

We need to configure two settings that do not appear in the current list of settings: "user.name" and "user.email". The instructions page on assembla.com should give instructions similar to this. Figure 3.1.2.3 shows how this is done.

```
git config user.name "marcan_m"
git config user.email "marcandre.moreau@gmail.com"
```

Figure 3.1.2.3 – Instructions for configuring the user.name and the user.email settings.

It is important that these two settings match the username and email address listed in your assembla.com account, otherwise you will fail to submit any changes to the remote git repository. It is not necessary to use the –global flag, since we only need these settings to be set for this git repository. Next, verify that the new settings have been accepted. See Figure 3.1.2.4 for an example.

```
marcan_m@linux:~/team7$ git config -l
core.repositoryformatversion=0
core.filemode=true
core.bare=false
core.logallrefupdates=true
remote.origin.fetch=+refs/heads/*:refs/remotes/origin/*
remote.origin.url=git@git.assembla.com:team7.git
branch.master.remote=origin
branch.master.merge=refs/heads/master
user.name=marcan_m
user.email=marcandre.moreau@gmail.com
```

Figure 3.1.2.4 – An example of the instruction for calling the configured settings and a list of the current ones.

You should now be ready to start making changes.

3.1.3 Contributing Changes

So you have worked on the project and you are ready to contribute your changes back to the git repository. The first thing you must do is add the files you have modified to the set of modifications you want to submit. Adding a file is as simple as typing “git add <filename>”. Removing a file is also simple, with “git rm <filename>”. If you want to move a file without losing its history in the repository, you can use “git mv <source> <destination>”. These are the most basic commands you need to know. Once you’ve added all of the items you wanted to submit, you can use “git status” to get a summary of what is going to be added and what is going to be ignored. Figure 3.1.3.1 shows an example where two new files have been created since the last version, but only one of the will be submitted:

```
marcan_m@linux:~/team7$ git status
# On branch master
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#         new file:   a.txt
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#         b.txt
```

Figure 3.1.3.1 – An example of a git status summary.

Once you are ready, you can use commit these changes. Figure 3.1.3.2 shows an example of the instruction for this.

Marcan_m@linux:~/team7$ git commit –a –m “comment describing the changes”
Figure 3.1.3.2 – An example of the instruction for a git commit including a comment.

A git commit creates a new version and commits the changes into that new version. The “-a” commits all of the added changes, and the “–m” is used to include a comment describing what those changes are. You can use git commit as many times as you want without affecting the work of other people, since the new version is only local to your machine at this stage.

After using git commit, you need to “push” your changes to the remote git repository so that the other developers can get it. Figure 3.1.3.3 shows the command for this.

marcan_m@linux:~/team7$ git push origin master
Figure 3.1.3.3 – The instruction for pushing changes into the git repository.

The “origin” command is the root of the remote git repository, and the “master” command links to the master branch. Because this is a small project, we will not be using the git superb branching feature. W e will only use a single branch, a.k.a. the master branch.

3.1.4 Synchronizing Work

If you attempt a “git push” and it fails, chances are it is because someone else has pushed a new version that is more recent than the one your work was based on. In this case, use “git pull” to fetch the latest version from the git repository. In the case that you had made changes that have not yet been contributed, git will automatically merge your changes with the latest version. After a successful merge, you should be ready to attempt to push your changes again. To see what the changes were, you can use “git log” or use a graphical tool like gitk. The log of the latest changes can also be browsed from assembla.com.

3.2 NetBeans IDE

3.2.1 Installation

On Linux, NetBeans can be installed by the distribution’s package manager. On Mac OS X and Windows, installers are available from netbeans.org. You need to install the PHP NetBeans bundle. If you choose to install the full NetBeans bundle, you will also need to install the JDK (Java Development Kit).

3.2.2 Plugins

In NetBeans, click on “tools->plugins” to get a list of available plugins, or to get updates. If you installed NetBeans without PHP, you can install the PHP plugin from there. A Javascript debugger plugin is available, which may be helpful.

3.1.3 Initial Configuration

Create a new project by going to “File->New Project” and selecting “PHP/PHP Application with Existing Sources”. Click “next”, which brings you to a dialog asking you for the source files location. Browse for the “www” folder of your local git repository. Call the project “Team7”, and then click “next”. For “Run As”, select “Remote Web Site (FTP, SFTP)”. Change the project url to http://team7.ath.cx:8080/~username/, where you replace “username” with your own, and a trailing slash at the end. For “remote connection”, click “manage”. Create a new connection of type SFTP. The hostname is team7.ath.cx, with your own username and password for ssh. Set the initial directory to /export/home/username/public_html where “username” is again your own username. Save the new connection by clicking “OK” and then select it. Empty the “Upload Directory” field and set “Upload Files” to “On Run”. Click “Finish” and you are done for that part. You can now code PHP in NetBeans. Clicking the green play button will automatically deploy your PHP code in your personal web folder and open a browser for you.

3.1.4 Remote PHP Debugger

In order to be able to use the remote PHP debugger from NetBeans, you need to open an ssh tunnel for port 9000 from the server to your own machine. This has to be done because the debugger attempts to connect to the debugger client locally on port 9000. By redirecting port 9000 to your own machine instead, the debugger will be able to connect to the PHP debugger client embedded into NetBeans. When you wish to use that feature, open a terminal and open an ssh tunnel as shown in Figure 3.1.4.1.

```
ssh -R 9000:localhost:9000 username@team7.ath.cx
```

Figure 3.1.4.1 – Command for opening an ssh tunnel in a linux terminal.

Leave the terminal window open for the entire time you need to use the remote PHP debugger.

4. PROGRAMMING LANGUAGES AND LIBRARIES

3.1 Languages

4.1.1 PHP

PHP is a server-side scripting language, which will be used to interface the database with the client-side application as well as process client-side requests with database information. We choose PHP to code the logic tier of the application because of several factors. It is the language that the largest number of people in our team felt comfortable coding with. It is well documented, largely used in the wild and easily deployable on any Apache instance. The code written using PHP is generally platform and environmentally neutral so it can be ported to any other environment quite easily without costly compilation.

4.1.2 HTML

HTML, which is short for Hypertext Markup Language, is a logical representation of a web interface and will used to organize the client-side aspect of the software. HTML was adopted for the front-end system, in order to comply with some of the platform constraints we met with in the problem statement defining our project. While many different technologies could display information in a web browser, only HTML provide compatibility across the largest amount of platforms and vendors. It is a well-defined, standardized markup language that will make the deployment of our application easier on any HTML 4.01 compliant browser, such as Internet Explorer 6.0+, Safari 2.0+, Chrome 2.0+ or Firefox 2.0+.

4.1.3 CSS

CSS, which is short for Cascading Style Sheets, is a system which maps visual styles and themes onto the web interface. Using CSS to style the page, an HTML document need not contain the visual aspect of the content, leaving it only the content and the structure in the HTML document. CSS can be disabled to permit a visually impaired person to access the page with little outside assistance.

4.1.4 Javascript (a dialect of ECMAScript)

Javascript is a client-side scripting language, which will be used to interact with the user on the web interface and send requests to the server and present responses from the server. In order to ease development by having a front-end with a constant and easily maintainable state, we choose JavaScript. Manipulating the object displayed in the browser's window will enable us to minimize the load placed on our logic tier thus complying with requirements of

speed and load we have to deal with. Javascript is the standard for interacting with the DOM (domain object model) that the browser exposes, in order to enable a programmer to manipulate objects parsed in the HTML code in various ways. This feature is standard across all major browsers aforementioned and is enabled on more than 95% of the web user's [1].

4.2 Libraries

4.2.1 jQuery

jQuery is a fast and concise JavaScript Library that simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development. jQuery is designed to change the way that you write JavaScript [2]. It is lightweight and offers a developer a lot of options that will help implement a complex UI project like ours. This open source project is maintained by the jQuery foundation, which is an NPO professionally run by a group of software engineers. The most prominent figure is John Resig, founder of jQuery, currently employed at Mozilla Corporation. Companies like Microsoft, Nokia, Google, Dell, Wordpress actively support jQuery. At the time of writing, 26.95% of all web sites used jQuery. Of all the sites that used Javascript as a web technology, more than 37% used jQuery which makes it the most popular technology used on the web today [3]. It is even more popular than Adobe Flash [4]. Of course popularity is of no interest without a community of devoted developers that provide bug fixes and plug-ins written for jQuery. jQuery is being served by google CDN (content delivery network) which ensure an up-to-date version is always running in our production environment [5].

4.2.2 jQueryUI

While jQuery provides an interface between the browser DOM and Javascript, jQueryUI provides a set of rich graphical user interface element that is highly reusable and customizable. We will particularly be interested in jQueryUI's ability to manage color themes, tabs, modal dialog, icons and buttons in a visually consistent manner without any added code in our project. jQueryUI is being served by google CDN (content delivery network), which ensure an up-tp-date version is always running in our production environment.

5. OTHER SOFTWARE

The following section covers the software used by the development team in order to plan and produce the application.

5.1 Operating Systems

5.1.1 Microsoft Windows

Microsoft Windows is the most widely used desktop operating system in the world and is one of the target platforms for the client-side aspect of the software being produced. It will be used by the team members mostly for its Office software.

5.1.2 Apple Mac OS X

Apple Mac OS X is a UNIX compliant operating system, which will be used by certain members of the development team to produce parts of the application. It will also be used to test the client-side aspect of the application for compatibility. Its use in the project is strictly a matter of personal preference of some members of the development team.

5.1.3 GNU/Linux

GNU/Linux is a UNIX compliant operating system, which will serve as the back-end of our application. It has been chosen for having a great architecture and a solid reputation as an operating system in the industry. It will be used in conjunction with the Apache server, the mySQL database management system and PHP.

5.2 Office Software

5.2.1 Microsoft Office

Microsoft Office is a set of software applications, which will be used in the planning and the documenting of the application during development. More specifically, the Microsoft Office Word, Microsoft Office Excel and Microsoft Office Visio applications will be used.

5.2.2 Dia

Dia is a software application, which will be used to construct the different diagrams required in documenting key aspects of the application.

5.2.3 MySQL

MySQL is a database management system, which will be used to store all of the persistent information used by the application. It one of the most used database management systems used in web applications and offers excellent performance with nominal overhead.

6. Human Resources

This section details the members of the development team, their roles, their technical experience and their average availability per week.

6.1 Eric Chan

Role: Team Leader, Programmer
Knowledge: - Web Development
- Database management
- Assembler, C, C++, Perl, Python, HTML/CSS/JavaScript
Experience: - Team projects
- Internships
Availability: 4 hours weekly

6.2 Sébastien Parent-Charrette

Role: Programmer
Skill: - Database development
- Low level system programming
- Assembler, C, C++, Java, Visual Basic, Perl, PHP
Experience: - Contract based software development
- Team projects
Availability: 4 hours weekly

6.3 Corey Clayton

Role: Programmer
Skill: - Low level system programming
- Mobile hardware system development
- Assembler, C, C++, Perl, Python
Experience: Team projects
Availability: 4 hours weekly

6.4 Mathieu Dumais-Savard

Role: Programmer
Skill: - Web Development
- Database Management
- HTML/CSS/JavaScript, Perl, PHP, Python
Experience: - Extensive number of web related personal projects.
- Business Intelligence
- Team projects
Availability: 4 hours weekly

6.5 Andreas Eminidis

Role: Programmer
Skill: - Database Development
- Detailed knowledge of security practices
- Assembler, C/C++, C#, Java, Perl, PHP, Python
Experience: - Work Experience
- Team projects
Availability: 4 hours weekly

6.6 Phuong-Anh-Vu Lai

Role: Programmer
Skill: - Database Programming
- 3D graphics programming
- C/C++, Java, Visual Basic, Perl, PHP, MATLAB
Experience: - Team projects
- Quality Assurance Testing for mobile device applications
Availability: 4 hours weekly

6.7 Julia Lemire

Role: Documentation Coordinator, Programmer
Skill: - Group management skills
- C++, C#, HTML/CSS/JavaScript (jQuery, Ext JS)
Experience: - Work Experience in web application maintenance
- Team projects
Availability: 4 hours weekly

6.8 Marc-André Moreau

Role: Systems Administrator, Programmer
Skill: - Detailed knowledge of networking protocols
- Detailed knowledge of security practices
- Knowledge of documentation process
- Assembler, C/C++, Java, Perl, PHP, Python
Experience: - Extensive number of personal projects
- Work Experience (incl. Software development process)
- Team projects
Availability: 4 hours weekly

REFERENCES

[1] "Browser Statistics." [Online]. Available: http://www.w3schools.com/browsers /browsers_stats.asp. [Accessed: Feb. 4, 2010].

[2] "jQuery: The Write Less, Do More, JavaScript Library." [Online]. Available: http://www.jquery.com/. [Accessed: Feb. 4, 2010].

[3] "JQuery Usage Statistics". [Online]. Available: http://trends.builtwith.com/javascript/ JQuery. [Accessed: Feb. 4, 2010].

[4] "JavaScript Usage Statistics". [Online]. Available: http://trends.builtwith.com/javascript. [Accessed: Feb. 4, 2010].

[5] "Developer's Guide – Google AJAX Libraries API". [Online]. Available: http://code.google.com/intl/xx-pirate/apis/ajaxlibs/documentation/. [Accessed: Feb. 4, 2010].