Organisational Aspects of Software Development

Pedro Contreras

Department of Computer Science
Royal Holloway, University of London

January 29, 2008
Creating software is a complex task.

- Organising tasks
- Managing people
- Maintaining code
- Controlling the development process
- Ensuring quality
- Estimating economical cost
- etc.
We present some useful tools together with a few tips that can help when creating software.

This presentation is separated into two main sections.

- **Development process and collaborative work** deals with tools and techniques to improve communication among Virtual Organisations.
- **Software Engineering Tool** deals with tools and standards to make coding faster and understandable for other programmers.
Virtual Organisation (or community) comprises a set of independent individuals that share resources and skills to achieve its mission / goal, but that is not limited to alliance for profit enterprises. Interaction among members of the virtual organization is mainly done through computer networks.

For example a group of people sharing common interest can organise themselves by creating a virtual community through a web portal.
From the software engineering point of view, a virtual organisation will tackle problems related with the software development process.
Collaborative Work and Tools

- Wiki
- Internet Forum
- Blogs
- News
- e-mail (e-mail lists)
- Chat & instant messaging programs (e.g. MSN)
- Others: telephone, meeting, video conference.
Wiki

From Wikipedia, the free encyclopedia.

For other uses of this term, see Wiki (disambiguation).

A wiki ([wɪkiː] <wo-kee> or [wɪkɪ] <wick-ee> (according to Ward Cunningham)) is a type of website that allows users to easily add and edit content and is especially suited for collaborative writing.

The term wiki also sometimes refers to the collaborative software itself (wiki engine) that facilitates the operation of such a website (see wiki software).

In essence, wiki is a simplification of the process of creating HTML pages combined with a system that records each individual change that occurs over time, so that at any time, a page can be reverted to any of its previous states. A wiki system may also provide various tools that allow the user community to easily monitor the constantly changing state of the wiki and discuss the issues that emerge in trying to achieve a general consensus about the wiki content.

Some wikis allow completely unrestricted access so that people are able to contribute to the site without necessarily having to undergo a process of 'registration' as had usually been required by various other types of interactive websites such as Internet forums or chat sites.

The WikiWikiWeb is named after the "Wiki Wiki" line of Chance RT-52 buses in Honolulu International Airport. The name is based on the Hawaiian term wiki, meaning "quick", "fast", or "to hasten" (Hawaiian dictionary). Sometimes wikiwiki (or Wikipi) is used instead of wiki (Hawaiian dictionary).

Wiki is sometimes interpreted as the backronym for "What I know is", which describes the knowledge contribution, storage and exchange function.

### Contents

1. Key characteristics
   1.1 Pages and editing
      1.1.1 Standard
   1.2 Linking and creating pages
   1.3 Searching
   1.4 User actions on a single wiki

Organisational Aspects of Software Development
**Forum example**

![WS-Talk forum screenshot](http://www.akra.de/ws-talk/fudforum/381458639/index.png)

**Organisational Aspects of Software Development**

January 29, 2008
Coding standards & Java

In the same way that any language has its own writing standards and protocols, coding in Java has some basic guidelines. This has the purpose of making the code more readable and understandable for other developers. The following section describes some simple principles when writing code in Java.
Software Engineering Tools

- Adhere to the style of the original
- Adhere to the principle of least astonishment
- Do it right at first
- Document any deviation
Java coding conventions

- Formatting
- Naming
- Documentation
- Packaging code
Java coding conventions

- **Formatting:** This includes things such as indentation for block statements, breaking up long lines, and use of white spaces instead of “hard tabs” - what looks perfectly formatted in one environment, can look as complete chaos in another-

- **Naming:** Usually Java Software Development Kit convention from Sun Microsystems are used, which includes some of the following: how to name classes, variables, methods, and constants, and when to capitalise.
**Documentation:** Write documentation for those who will be using the software as well as the people that will maintain it. Documenting Java code for other programmers can be done by means of using comments and using Java documentation package.

**Packaging code:** Java code organises classes in packages, making it easy to re-use code. For example, when creating a new package include only related classes, since when using the package it needs to be imported. Therefore if packages are not well organised the software becomes inefficient.
Interactive Development Environment (IDE) is an integrated system to assist in the software writing; usually such systems includes tools to help with code editing, graphical design, compiling and running programs, and debugging.
IDE NetBeans example

```cpp
string freqFilename(argv[2]);

bool extractFreq = false;
if (strcmp(argv[3], "true")==0) {
    extractFreq = true;
}

string emailDB(argv[4]);
string flatfileDB(argv[5]);
string stoplist(argv[6]);
string dictionary(argv[7]);

bool useDict = false;
if (strcmp(argv[8], "true")==0) {
    useDict = true;
}

ParseFiles pf(dr, freqFilename,
    extractFreq,
    emailDB,
    flatfileDB,
    stoplist,
    dictionary,
    useDict);

t.stop();
cout << "Running Time: " << t << endl;
return 0;
```
A version Control System is a centralised place where files can be stored, and accessed from any machine with Internet connection. Also it provides a way to store different versions of a document. Then if any version needs to be recovered it can be done easily.
Example: Concurrent Version System
Bugzilla is a free Defect Tracking Systems that allows individual or groups of developers to keep track of outstanding bugs in their software effectively. Bugzilla creates a web-based central repository to maintain a running list of reported defects and their status.
Some of the following tasks can be carried out with Bugzilla:

- Track bugs and code changes
- Communicate with team-mates
- Submit and review patches
- Manage quality assurance (QA)
Bugzilla engine example

Bugzilla Version 2.16.5

Organisational Aspects of Software Development
Apache Ant is an open source Java-based build tool

- Ant build files are platform independent
- Ant tracks files dependencies
- Ant Java-based tasks
Ant build files are platform independent

Ant resolve any platform dependencies such as Operating System (OS) commands (e.g. create directories) and how to format correctly the Java classpath.
Ant tracks files dependencies

javac compiler is only invoked when source files have been changed. Thus when compiling files just the changes are recompiled and not everything.
Ant Java-based tasks

Ant includes a wide range of tasks, that are very helpful for customising processes. For example Ant includes task for running JUnit tests. Also Ant can be extends by writing custom tasks.
Apache Ant Example

```xml
<project name="alwaysontop" default="make.jar" basedir="." >

<!-- IMPORTANT VARIABLE HERE -->
<property name="build.dir" value="C:/Projects_Local/DevDaily/AlwaysOnTopTest"/>
<path id="class.path">
  <fileset dir="lib">
    <include name="**/*.jar"/>
    <include name="**/*.zip"/>
  </fileset>
</path>
<target name="init">
  <property name="project.name" value="alwaysontop"/>
  <property name="jar" value="${build.dir}/jar/${project.name}.jar"/>
  <property name="mainclass" value="com.devdaily.alwaysontop.Main"/>
  <property name="sampleOutDir" value="${build.dir}/data"/>
  <property name="nameOutDir" value="data"/>
  <timestamp/>
</target>

<target name="create_classes_dir" depends="init">
  <mkdir dir="${build.dir}/classes-ant"/>
</target>

<!-- CLEAN TARGET -->
<target name="clean">
  <delete dir="${build.dir}/classes-ant"/>
</target>

<!-- COMPILE TARGET -->
<target name="compile" depends="clean,create_classes_dir">
  <javac srcdir="${build.dir}/classes-ant" source="1.4">
    <src path="src"/>
    <include name="**/*.java"/>
    <classpath refid="class.path"/>
  </javac>
  <copy todir="${build.dir}/classes-ant">
    <fileset dir="${build.dir}/src">
      <include name="**/*.gif"/>
      <include name="**/*.jpg"/>
      <include name="**/*.png"/>
    </fileset>
    <fileset dir="${build.dir}/">
      <include name="reports/**/*"/>
    </fileset>
  </copy>
</target>

</project>
```

Organisational Aspects of Software Development

January 29, 2008
There are a lot of different kinds of testing that can be performed on a software project. In some cases testing requires extensive feedback from the end users; other testing forms may require a dedicated Quality Assurance teams, or other extensive resources. Unit test, and more specifically JUnit, is a piece of code dedicated to exercise a very small, and specific functionality of the code to be tested.
SSH is a program for logging and executing commands into a remote machine. It provides secure encrypted communications between two non-trusted hosts over an insecure network. SSH can use different authentication methods such as RSA keys (algorithm for public-key encryption). Though remote log-in is the primary use of SSH, the protocol can be used as a general purpose cryptographic tunnel, capable of copying files, encrypting e-mail connections, and triggering remote execution of programs.
References