Massey University

159.251 - Software Design and Construction - 2017

Introduction

Welcome to 159.251 Software Design and Construction. This paper is a core component of the Software Engineering and Computer Science degrees. However, this paper is also useful for students majoring in Engineering (BE) and Information Technology (BInfSci).

Objectives

This is *not* a classical programming paper. The main focus of this paper is on **productivity**. This includes aspects such as:

1. be able to work in teams
2. be able to analyse faults and performance bottlenecks in programs
3. design programs that have a certain level of scalability and robustness
4. automate routine tasks
5. write software that is maintainable

While the tools and methods discussed in this paper are not directly related to any particular life cycle model, many have been developed and proposed in the context of *agile software engineering*. We will quickly recap some agile software engineering practices in the beginning of this paper. This paper also uses many ideas from an approach called *software craftsmanship*.

Assessment schedule for 159.251

10 weeks in Total

- **Assignment 1 -** worth **19%**
- **Assignment 2 -** worth **19%**
- **Weekly Tutorials:** 8-9 tasks, that together is worth **10%.**
- **Final exam:** time and place to be advised. Worth **52%**.

Weekly Tutorials

There are weekly tutorials / labs – starting from week 2. **Note that these tutorials are marked, and contribute 10% to the overall mark. It is therefore highly recommended to attend and submit these tutorials!**

Technology Choices

This is a rather technical paper, and requires a good understanding of programming. We will assume that you are familiar with Java or a similar language (such as C#), and most of the examples used in this paper will use Java and Java-related development tools. Many development tools used will be available as plugins for the Eclipse development environment. However, most concepts can be easily applied to other technologies, and for each tool and method we discuss there will often be one or many counterparts in other technology spaces (such as .NET with VisualStudio or Python).

We made this choice for the following reasons:

1. Java remains to be one of the most popular programming languages, and many of the organisation that hire our graduates use it
2. Java is conceptually very similar to some other mainstream programming languages, such as C++ and C#.
3. Java and many Java tools are free.
4. Java and Java development tools are platform independent, and run on all major computing platforms including MacOS, Windows and Linux.
5. Many of the tools we discuss (such as profiling and build tools), do not require Java but the Java runtime environment (the Java Virtual Machine JVM). This means that these tools can also be used with other languages that can compile into Java byte code such as Python (JPython), Scala and Ruby (JRuby).

We will also discuss OS scripting in this paper. Here we use Unix. We realise that many students use Windows PCs. However, it is rather easy to install unix command line support on Windows by using cygwin. Note that you can also run Linux distributions directly from within windows, for instance by using the Ubuntu LiveCD. If you use a MAC, you already have access to the unix command line.

Text Books

There is no prescribed text book for this paper. However, there are some excellent books directly related to the topics covered. In particular, we highly recommend the following books:

- Andrew Hunt and David Thomas: The Pragmatic Programmer: From Journeyman to Master. Addison-Wesley, Oct 1999.
- Robert Martin: Clean Code: A Handbook of Agile Software Craftsmanship. Prentice Hall 2009.
- Joshua Bloch: Effective Java Second Edition. Sun Micro 2008.

Topics

Introduction

1. The Problem with Software

Part one: Collaborative Software Engineering

1. The agile thinking – Agile Software Development
2. Version Control and Software Configuration Management
3. Continuous integration and Continuous Development
4. Software Metrics and measurements
5. Software Licensing

Part one: Effective Software Development and Software Quality

1. DRY and the Evils of Duplication
2. Separation of Concerns and Orthogonality
3. Organising Code
4. Testing
5. Process Automation and Diagnostics
6. Design Patterns, Antipatterns and Refactoring

Software

Installing Java

1. Download and install the Java Development Kit (JDK) from http://www.oracle.com/technetwork/java/javase/downloads/index.html for your platform.
2. The minimum version required for this paper is **1.8.**
3. To check the version of Java, open a command line window and enter **java -version**

Installing Eclipse

Go to http://www.eclipse.org/downloads/ to download **Eclipse IDE for Java Developers**. The minimum version we use is **Juno (4.2)**. Later versions (Luna, Mars, Neon, Oxygem) can also be used.

Creating an account in a Version Control System

As a part of this course, you are required to have an account in service that provide free source code hosting. Our preference is Bitbucket (https://bitbucket.org/product). Creating an account with Bitbucket is easy and straightforward (Click **Get Started** and then follow the instructions).

IMPORTANT: when you create new projects for this course, make sure that these projects are Private so that only you can have access to these projects.

Customising Eclipse

General procedure to install plugins:

1. Start the installation procedure: select the Help>Software Updates>Find and Install... menu item.
2. Select "Search for new features to install" option and click Next.
3. Click "New Remote Site ...", and enter URL
4. You have to accept the license at some stage, and should restart Eclipse after you have installed an update.

name	plugin url	purpose	remarks
PMD	https://dl.bintray.com/pmd/pmd-eclipse-plugin/updates/	source code audit	
FindBugs	http://findbugs.cs.umd.edu/eclipse/	source code audit	
Git	https://git-scm.com	*Git application for different OS a*	*Also install SourceTree (https://www.sourcetreeapp.com) - Git client for Windows and Mac.*
EclEmma	http://update.eclemma.org	test coverage tool	
Eclipse Metrics Plugin	http://metrics2.sourceforge.net/update	code metrics	
JDepend	http://mcs.une.edu.au/doc/jd	design metrics	

	epend/docs/JDepend.html		
JDeodorant	https://users.encs.concordia.ca/~nikolaos/jdeodorant/	Refactoring plugin for Eclipse	

Note: if you are behind a proxy, you have to configure the proxy settings in the Eclipse general preferences.

Accessing Examples

The examples used and referenced in this paper can be downloaded from the following GIT repository:

https://bitbucket.org/jensdietrich/oop-examples

Instructions how to do this can be found on this web site.