

CSC 5524 : Software quality, metrics, tests, processes

J Paul Gibson, D311

`paul.gibson@telecom-sudparis.eu`

<http://www-public.telecom-sudparis.eu/~gibson/Teaching/CSC5524/>

Software Quality - Tips Techniques & Tools

[http://www-public.telecom-sudparis.eu/~gibson/Teaching/CSC5524/
TipsTechniquesTools.pdf](http://www-public.telecom-sudparis.eu/~gibson/Teaching/CSC5524/TipsTechniquesTools.pdf)

These are the following TTTs that all quality software engineers know about, and use as a matter of habit:

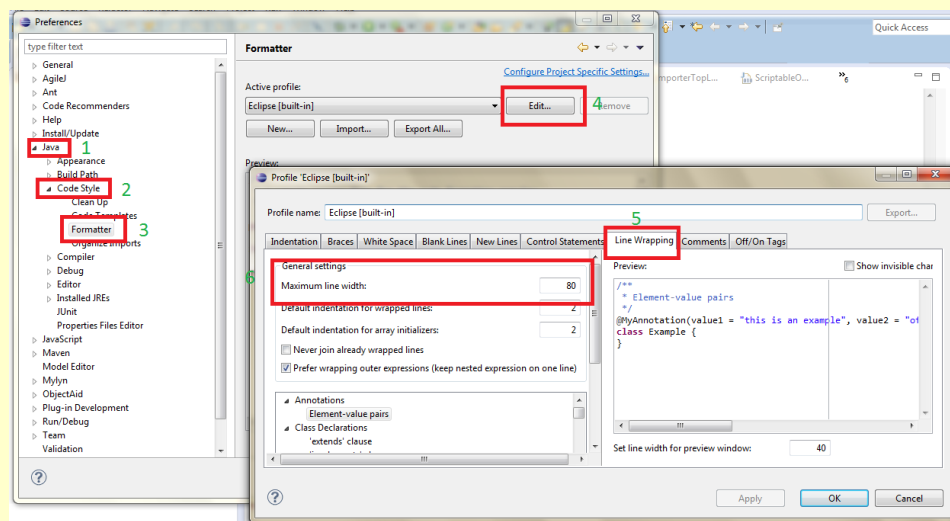
- Code style formatting
- Literate Programming
- Code Documentation Generators
- Code defect detection (bug finding)
- Design defect detection (bad smells), based on OO metrics
- Automated unit testing
- Code Coverage
- Version control
- Automated build: make, ant
- Continuous integration/delivery/testing
- Personal Software Process
- Software Process Improvement (SPI) & Capability Maturity Model (CMM)

Code style formatting: indenting, whitespace

All modern IDEs include tools for automatically formatting code to a predefined template/style/configuration

Many permit the definition of a library of these templates

TODO: Can you find and use this functionality in Eclipse (for Java)?



READING : *The elements of programming style*, Kernighan, Brian W., and Phillip James Plauger, 1979

Literate Programming

READING: Donald Knuth. "Literate Programming" (1984)

"Let us change our traditional attitude to the construction of programs: Instead of imagining that our main task is to instruct a computer what to do, let us concentrate rather on explaining to human beings what we want a computer to do."

Useful Link: http://vasc.ri.cmu.edu/old_help/Programming/Literate/literate.html

Code Documentation Generators

All modern IDEs include tools support for automating documentation generation

Language dependent, e.g. JavaDoc

<http://www.oracle.com/technetwork/java/javase/documentation/index-137483.html>

<http://doclet.com>

Language independent, e.g. doxygen

<http://www.stack.nl/~dimitri/doxygen/index.html>

Code defect detection (bug finding)

Static analysis of code

Example, **findbugs**

<http://findbugs.sourceforge.net>



Design defect detection (bad smells), based on OO metrics

Example: JDeodorant is an Eclipse plugin that identifies bad designs and proposes fixes (refactorings)



<http://users.encs.concordia.ca/~nikolaos/jdeodorant/>

Automated unit testing

Nearly all modern programming languages have library/tool support for automated unit testing

For Java - JUnit



<http://junit.org/junit4/>

For C - CUnit

<http://cunit.sourceforge.net>

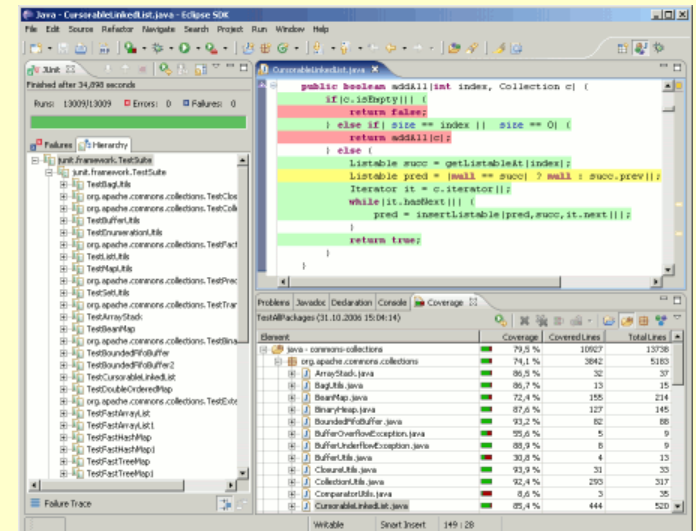
Code coverage

When running tests, you need to know what percentage of code executed during the test runs.

Most modern IDEs provide coverage tools/plugins

For Eclipse, with Java, a good example is Eclemma

<http://eclemma.org>



Version control

You should know the basics of

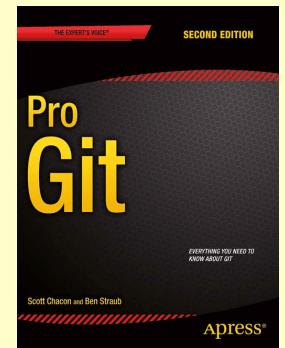
Subversion -svn -

<https://subversion.apache.org>

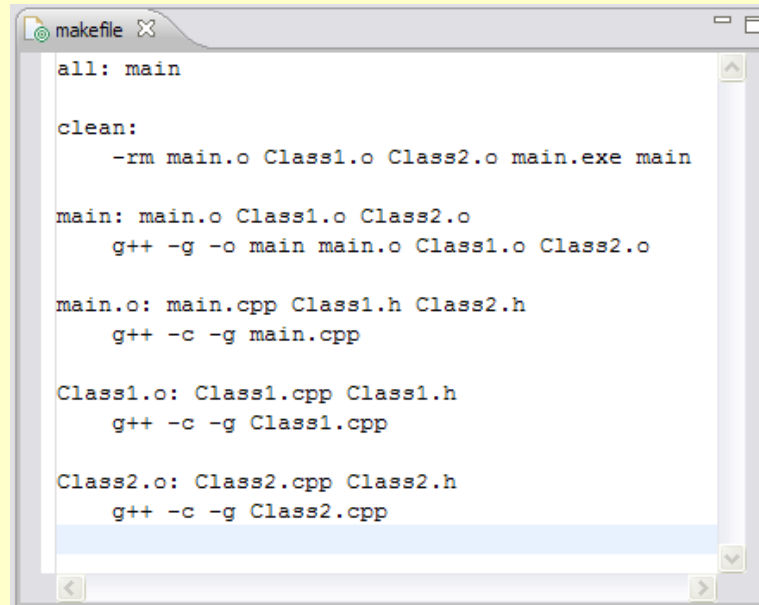


Git

<https://git-scm.com/>



Automated build: make, ant



```
all: main

clean:
    -rm main.o Class1.o Class2.o main.exe main

main: main.o Class1.o Class2.o
    g++ -g -o main main.o Class1.o Class2.o

main.o: main.cpp Class1.h Class2.h
    g++ -c -g main.cpp

Class1.o: Class1.cpp Class1.h
    g++ -c -g Class1.cpp

Class2.o: Class2.cpp Class2.h
    g++ -c -g Class2.cpp
```

<http://www.gnu.org/software/make/manual/make.html>



Another neat tool

Continuous integration/delivery/testing

Maven

<https://maven.apache.org>



Sonar

<http://www.sonarqube.org>



Jenkins

<https://jenkins.io>

Hudson

<http://hudson-ci.org>



Make build easy

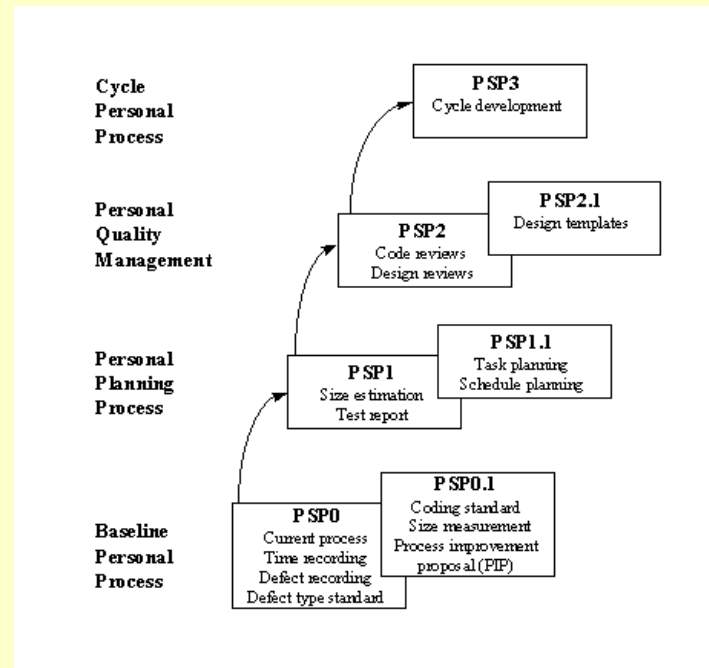
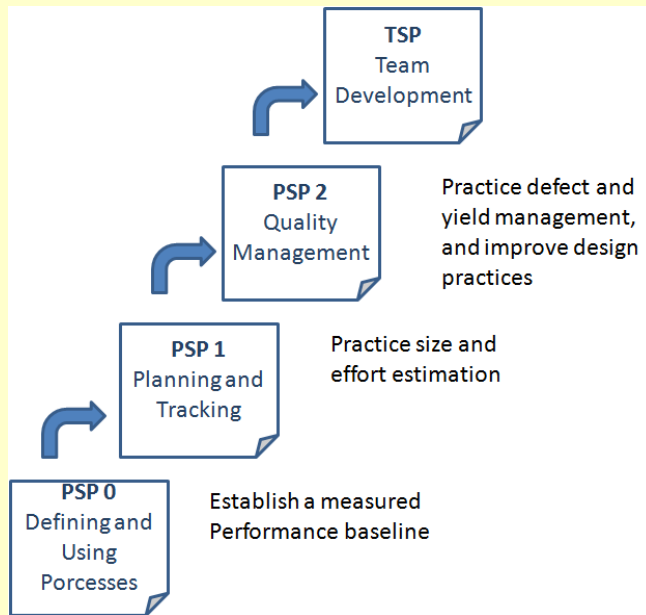
Make build a uniform process

Provide quality information feedback

Support/enforce best practice for test/delivery

Wrap up all this on an intreated service/server

Personal Software Process

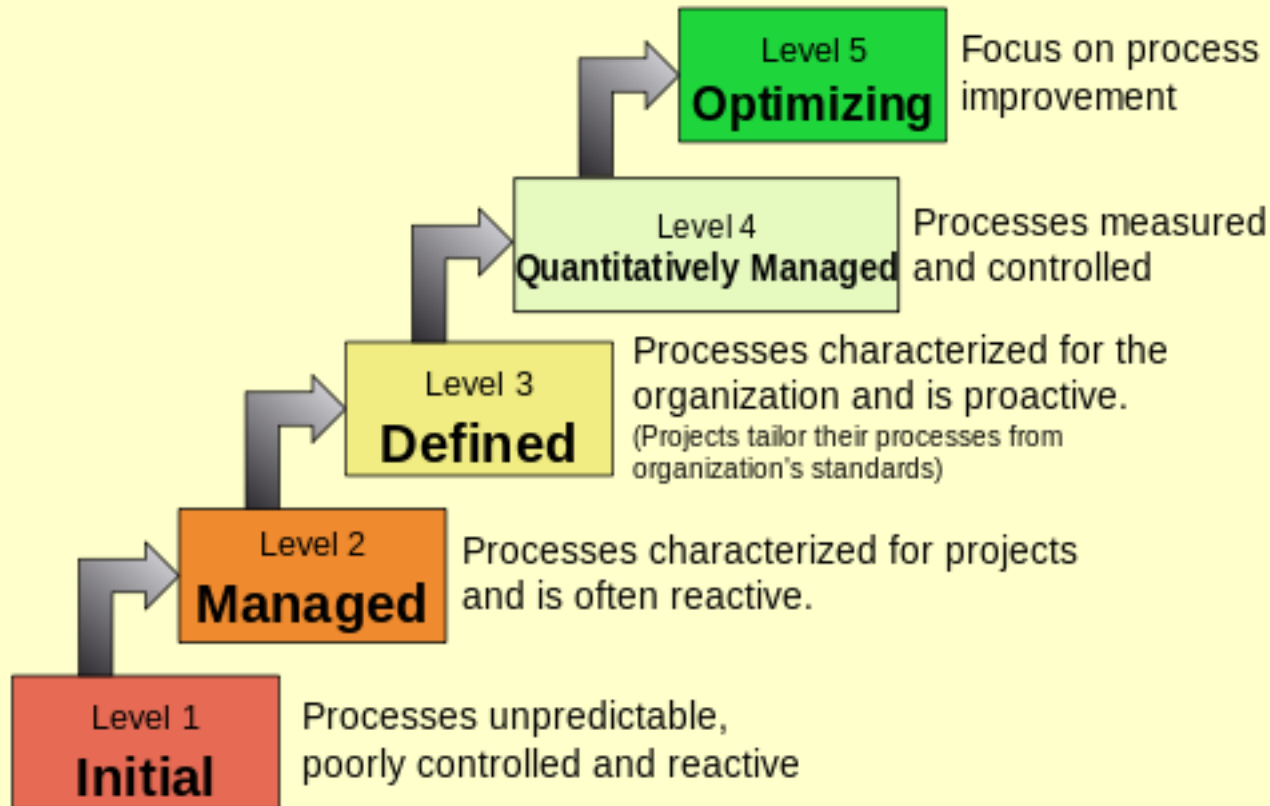


Using a defined and measured Personal Software Process, Watts S. Humphrey, 1996

The Personal Software Process: A Cautionary Case Study, Philip M. Johnson, Anne M. Disney, 1998

Software Process Improvement (SPI) & Capability Maturity Model (CMM)

Characteristics of the Maturity levels



<http://www.sei.cmu.edu/cmml/>