Software Tools

Software Architecture, Process, and Management

This lecture will review some of the software that is available to assist with project management of specifically software projects.

Tool Types

- Version control (e.g. Git)
- Build control (e.g. make)
- Debuggers (e.g. gdb)
- Profilers
- Unit/regression testing (e.g. JUnit)
- Bug/issue tracking (e.g. Trac)
- Documentation generation (e.g. JavaDoc)
- Project management (e.g. MS Project)
- Integrated suites (e.g. RUP)
- Others

Why Use Version Control?

- Three main reasons:
  1. Allow multiple developers to edit the same files concurrently
  2. Maintain a history of changes such that any previous version can be inspected
  3. Allow changes to be distributed to multiple concurrent versions
     - For example, development and release versions

Build Control

- Build control tools like `make` automate the process of generating an executable or compiled version of a program or other file or document from source files:

```
UNIX> make
    cc -c file1.c
    cc -c file2.c
    cc -o a.out file1.o file2.o
UNIX>
```

Debuggers

- Debuggers allow you to:
  - Step through code line by line
  - Set break points
  - Allow state to be examined or changed
- They are essential for C/C++, but extremely useful even for interpreted languages
- Many IDEs include integrated debuggers, but there are separate command-line debuggers:
  - gdb, jdb, pdb, etc.

  and many graphical ones:
  - ddd, Insight, etc

Profilers

- Many student code submissions contain evidence that there has been an attempt to optimise the code for better performance
- There is little evidence of any benchmarking
- Even less evidence of any profiling
- Benchmarking, can be as simple as timing how long it takes to run your test suite, without it, you cannot know that your optimisation has indeed been an improvement
- Profilers tell you where the time and memory are spent in your application
- If you optimise without a profiler you are **guessing** and there is a strong possibility that you are wasting your time
- **JVMMonitor** is an example of monitoring/profiling software

Unit/Regression Testing

- Setting up an **<u>automated</u>** suite is the second thing to do after setting up source code control
- It is not less important than source code control, it is just that your test suite should be stored in source control
- You should never need to run a test manually, it is just as quick to add a new test to the automatic test suite and have that run it
- The output of tests should be automatically checked
- Again, I see many student submissions with a few test input files, but no means of automatically running these test inputs and checking to see if the output is as expected

Testing

- More importantly tests make sure you are not introducing flaws into existing features that once worked
- Automatic testing means that you will run your tests periodically and usually before any commit to the source code control
- The earlier you find flaws, the easier and quicker they are to fix

Document Generation

- No one actually writes documentation consistently, so it always gets out of sync with code
  - similarly for comments
- Documentation generation software like JavaDoc (Java), NDoc (.NET), PHPDoc (PHP), and Doxygen (C++, C, Java, etc.) Sphinx (Python) automatically generates documentation from your source code and comments
- Most of it is guaranteed to be up to date, and if developers know their comments are going to be used as-is for the reference manual then they are less likely to consider it wasted effort to update their comments when the code changes

Project Management

- Keeping track of all the sub-tasks in a large project, such as:
    - who they are assigned to
    - in what order they have to be done
    - how long each will take
  is a pain (at best)
- Project management software like Microsoft Project helps with this
    - Main feature: automatic Gannt charts and PERT charts
- Basecamp is a bit more ambitious:
    - adding distributed time tracking
- Also many free programs, such as: Planner, GanttProject, KPlato, etc
- Even for individual projects Trello is well worth looking into