Week 1: Software Engineering

Reading: Pressman Chapters 1 (Software Engineering); Pressman Chapter 2 (Process Models)

1. Develop your own answers to Pressman's five questions about software.
   1) Why does it take so long to get software finished?
   2) Why are software development costs high?
   3) Why can't we find all the errors before software is delivered?
   4) Why do we spend so much time and effort to maintain existing software?
   5) Why is it so difficult to measure progress as software is developed and maintained?

Solution: 1 because SW is complicated and the scope tends to grow during development
2 because SW is written by people and it is very flexible
3 because SW is so complex
4 because SW is so complex, and because it is expensive it has to last a long time
5 because SW is so flexible, every project is different; because SW developer productivity varies dramatically

More suggestions from Pressman
It takes software so long to be finished, for the following reasons
a) Facilities are not available on line.
b) Development tools do not work as expected.
c) Customer insists on the new requirements, requiring redesign and rework.
d) Product depends on the government regulations that change unexpectedly.
e) Strict requirements for compatibility with existing system require more testing, design, and implementation than expected.
f) Requirements to operate under multiple operating systems take longer to satisfy than expected.
g) Software project risk management takes more time then expected.
h) Dependency on a technology that is still under development lengthens the schedule.

Development costs are high:
a) Unacceptably low quality requires more testing, design and implementation work to correct than expected.
b) Development of the wrong software functions requires redesign and implementation.
c) Development of the wrong user interface results in redesign and implementation.
d) Development of extra software functions that are not required extends the schedule.

We can't find errors before we give the software to our customer for the following reasons:
a) Product depends on government regulation, which changes unexpectedly.
b) Product depends on draft technical standards, which change unexpectedly.
c) New development personnel sometimes are added late in the project.
d) Conflicts within teams sometimes results in poor communication and hence poor design
e) Sabotage by project management results in efficient scheduling and ineffective planning.
f) Sometimes the furnished components are poor quality resulting in extra testing, design and integration work and in extra customer-relationship management.

We continue to have difficulty in measuring progress as software is developed since:
a) Sometimes the purpose of the project is not clear.
b) There are a lot of business risks involved.
c) If the product built is not fitted well.
d) We need to review our work continuously.
e) A time check has to be maintained.
f) Project team has to be thorough and organized throughout the process.

2. A common problem during communication occurs when you encounter two stakeholders who have conflicting ideas about what the software should be. That is, they have mutually conflicting requirements. Develop a process pattern that addresses this problem and suggest an effective approach to it.

SOLUTION: A process pattern identifies the action is needed and the work tasks to be performed. It can be specified formally (as below) to document the actions so it can be reused and refined in other projects. For conflicting requirements, first consider the priority each stakeholder places on this requirements and others. Look for a win-win compromise. May need a meeting to brainstorm a compromise.

Pressman solution:
Pattern Name. Conflicting Stakeholder Requirements
Intent. This pattern describes an approach for resolving conflicts between stakeholders during the communication framework activity.
Type. Stage pattern
Initial context. (1) Stakeholders have been identified; (2) Stakeholders and software engineers have established a collaborative communication; (3) overriding software problem to be solved by the software teams has been established; (4) initial understanding of project scope, basic business requirements and project constraints has been developed.
Problem. Stakeholders request mutually conflicting features for the software product under development.
Solution. All stakeholders asked to prioritize all known system requirements, with resolution being to keep the stakeholder requirements with highest priorities and/or the most votes.
Resulting Context. A prioritized list of requirements approved by the stakeholders is established to guide the software team in the creation of an initial product prototype.
Related Patterns. Collaborative-guideline definition, Scope-isolation, Requirements gathering, Constraint Description, Requirements unclear
Known Uses/Examples. Communication is mandatory throughout the software project.

3. Cockburn argues "All organisations have a methodology - it is simply how they do business." and "Your methodology is everything you regularly do to get your software out . . . the conventions your group agrees to."

   Consider the 12 items in Cockburn's Elements of a Methodology figure (slide 8 lecture 1):

   > Process, Milestones, Values
   > Activities, Techniques, Tools
   > Teams, Roles, Skills
   > Quality, Products, Standards

   Choose a software project that you have worked on, select a few of the given methodology concepts and describe your methodology. Identify any lessons learned from that project and how you might improve it for future projects.

SOLUTION: To discuss in groups. Discussion points included:

Team and skills: identified project leader and audited skills of each of the members. One big team project broke into 2 subgroups and had a project manager to coordinate their work.

Standards and products: often determined by the university project eg sprints and deliverables for CITS3200 team project.

Milestones: One group noted late changes to requirements in a team project as a problem. Need a way to manage unreasonable expectations. Another problem is over-confidence expecting some jobs can be completed in a short time when they cannot.

4. S and J Robertson propose the "brown cow" model as a way of capturing multiple viewpoints, and capturing the "essence" of a problem. Click on the links to read How Now Brown Cow (pdf) and/or watch the (video).

   Consider the problem of providing material for learners and teachers in a university course Identify several different view points for this problem (stakeholders) and describe each one briefly. In groups, each student should role play one of these view points. Then, brainstorm specific items for each of the quadrants: what now; how how; what future and how future. For example, put in the how future quadrant, and then work backwards to discover the what now or what future quadrant requirements. For example, a how-future requirement could be “Access to learning materials should be password protected for enrolled students only”. Identify some “what” requirements for this problem. Concentrate on distinguishing between the what (essence) and how (solution), and on recognising different viewpoints.

Solution: Discuss in class. Two important take aways from this exercise: 1) separate the actual requirements from solution. and not to jump to a solution before you have understood the problem.

2) to recognise that there are (many) different viewpoints

Example of what now: “VC wants to protect the Intellectual Property of the University” Example of what future: “Students want to browse units to select a course”