Introduction to SRPM

Software Requirements & Project Management
CITS3220
CITS3220 Organisation

- Unit Coordinator: Prof Mark Reynolds
- Lecturers: Mark Reynolds, Terry Woodings, Mohammad Behdad plus guests from industry
- Contact: CS reception Email: admin3220@csse
- Lectures: Tuesday 11-12 & Thu 3-4 GPB2LT
- Practical Class: Thu 4-5
- Consultation: Tuesday 12 noon
- See SCHEDULE for assessment deadlines
Overview of Lecture 1

1. Problems addressed in SRPM concern
   - Requirements
   - Project Management
   - Validation

2. Characteristics of
   - Software
   - Software Engineering
Requirements Engineering
Requirements (to date)

“design and implement a flight booking system using Java RMI. It is assumed that the Company is on the server side, with all the different system classes and functionalities, and the GUI client and interfaces are on the client side. Consider the following use cases:

Login: This use case starts when the Travel Agent wants to login and use the different system functionalities. The system prompts the Travel Agent to complete their username and password in order to be able to use the system. The Travel Agent enters their username and password. The system validates the username and password, and grants the user access to the different airlines records, and the different system functionalities.”
[Ref: M.Bennamoun, CS304 project]

- Clear
- Concise
- Doesn’t change
- Feasible
Scientific American
Order Processing

- Problems: labour-intensive, slow, expensive, unreliable, unable to manage peak demands
- Solution 1: Replace Tab runs with automated master file processing and updating system
Problems with Solution 1

- Costs up
- Reliability and quality of service down
- More clerical staff required
- Employee morale down
- Employee turnover up

Why? Programming solution overlooked some key operational elements of the problem

To be continued in lecture 3
Project Management (to date)

“The deadline for the project is 5pm Tuesday 1st of June 2004. The demonstration will be held on Thursday 3rd of June from 9am-12pm in Room 2.07. Your should submit: Source code and class files for your system. The code has to be well documented. A how to use (or run) file.

You will be assessed on …”

[Ref: M.Bennamoun, CS304 project]

- **Fixed Deadline**
- **Fixed project time (plus or minus a few late nights)**
- **Feasible: done 100s of times before**
Denver Airport Baggage System

The New Denver Airport was to have a totally automated baggage-handling system: a large scale, client/server, real time distributed system with 35 kilometres of track, 4,000 high-speed, bag carrying, independent "telecars" mounted on tracks to carry and deliver baggage to 20 airlines.
Denver Airport Outcomes

- Delivered 16 months late
- US$3.2m over budget
- Delivered system was mostly manual
- Cost of US$m per day in lost revenues, interest and operating costs.
Software Crisis?

The term **software crisis** was coined at the first NATO SE Conference in 1968.

**Causes:**
- overall complexity of the software process
- immaturity of software engineering as a profession

**Effects:**
- Projects running over-budget.
- Projects running over-time.
- Software was of low quality.
- Software often did not meet requirements.
- Projects were unmanageable and code difficult to maintain.
CHAOX Report 1994

1994: of 8,380 projects in the government and private sectors in the USA:

- 31% of software projects are cancelled before they are completed.
- 53% of those that are completed cost an average of 189% of their original estimates.
- Of those 53%, only 42% have the original set of proposed features and functions.
- Only 9% of the projects were completed on time and on budget (16% counted as successes)
CHAOS Report 2004

- Project success rates have increased to 34% of all projects; more than 100% improvement from 1994.
- 15% of software projects are cancelled before they are completed.
- 51% are challenged projects (over time, over budget and/or lacking critical features).
- Of those 51%, most have cost overrun under 20% of budget.
- All projects (including failures) cost overrun 43% in 2004 vs 180% in 1994.
- Why? “The primary reason is the projects have gotten a lot smaller. Doing projects with iterative processing as opposed to the waterfall method, which called for all project requirements to be defined up front, is a major step forward.”

Standish Chairman Jim Johnson

Software Magazine: 2004-01-15
Experts disagree about the validity of CHAOS findings. Researchers who have queried Standish, asking for a description of their research process, a summary of their latest findings, and in general a scholarly discussion of the validity of the findings, have been rebuffed, calling the Standish methods into question.

Communications of the ACM column, Robert Glass asked two questions of the Chaos Report: "Does it represent reality?" and "Is it supported by research findings?"
Verification & Validation
Verification & Validation (to date)

“During your five minutes demonstration you have to demonstrate some of the test cases below (as will be requested during the demonstration). You also have to show on your report that you have repeatedly and successfully tested ALL these cases with (whenever possible) a snapshot of your GUI. You may add a better mean to prove or illustrate your results on your report. Here are the cases you need to test.” etc etc. [Ref: M.Bennamoun, CS304 project]

- Then submit and forget 😊

- Limited testing time (up to fixed deadline)
- No maintenance cycle
- No users
The Cost of Software Failures

- Loss of mission
- Financial loss
- Loss of reputation
- Loss of life
NASA's Mars Global Surveyor Lost

- MGS spacecraft, missing since November 07, may have got lost because of faulty software.

- “The error could have caused the craft's solar array to move to the wrong spot in response to a NASA command on 2 November. The MGS then went into a "safe mode", possibly oriented in a way that left temperature-sensitive components in sunlight for too long, causing a battery to fail and so leaving the spacecraft without enough power to operate.”

- From issue 2587 of New Scientist magazine, 19 January 2007, page 4-5
Mars Climate Orbiter

1999 MCO lost just as it entered Mars orbit

cause: failed translation of Imperial units into Metric units in a segment of ground-based, navigation-related mission software

cause: systems engineering, project management, and communication problems

2005 Titan Probe?
Ariane 5

- On 4 June 1996 Ariane5 exploded 40sec after lift-off
- cause: a floating-point error in the Inertial Reference System, a reused module from Ariane4
Figure 5: A typical Therac-25 facility after the final CAP.
Figure 1: Upper turntable assembly.
Therac-25

- Between June 1985 and January 1987 the Therac-25 radiation therapy machine massively overdosed 6 people.
- Failure can be traced to race conditions in the (re-used) software BUT
- Many systems aspects are also responsible: lack of defensive design, unrealistic risk assessments etc.
Software Engineering
Software Engineering & Computer Science

Computer Scientist asks: what problems can be solved using software?

Real World Problems

Software Engineer asks: How do we build software that solves real world problems?

Software Solutions
Characteristics of Software Systems

- Software is amongst the most complex of man made systems
- Engineered software artefacts are invisible
- To remain useful, SW systems must change
- New software applications are driven by the fast pace of technological change
Is there a Software Crisis?

“Software engineering gets a really bad press… however, the problems with the [Denver airport] system were much more than software problems. They included problems of system acquisition, volatile requirements, management and hardware design… The system as a whole, and not just the software failed to operate correctly”

[Sommerville Systems Engineering notes, 1997]
“Socio-technical systems are systems of systems that include human and organisational systems as well as software and hardware systems. I am convinced that progress in building more dependable software relies on firstly, making that software a better fit for the organisation and, secondly, designing that software so that people in the organisation can recover from failures. I am therefore primarily interested in the interface between the system software and the operational and organisational processes that use that software.”

Prof Ian Sommerville 2006
http://www.cs.st-andrews.ac.uk/
Responsibility

“one of the major differences between people and computers: people can be given or assume responsibilities and computers can't. Many system failures are, at least, partly a consequence of responsibility failures. To reduce responsibility-related failures, we need to develop a deeper understanding of these failures, to understand how responsibilities interact in complex computer-based systems and to invent ways of making responsibilities explicit in models that can be used to inform system design.”

www.dirc.org.uk
Software Engineering (Defns)

- The discipline of **software engineering** is the systematic study of concepts, procedures and tools for developing software. [Sommerville]

- The activity of software engineering is to
  - **engineer** (verb) to build, construct, manage
  - **software** (noun) programmes not forming part of the machine
Software Engineering (Defns)

- “The establishment and use of sound engineering principles (methods) in order to obtain economically software that is reliable and works on real machines” [Bauer 1972]

- “SE is that form of engineering that applies the principles of computer science and mathematics to achieving cost-effective solutions to software problems” [CMU/SEI-90-TR-003]
Software Engineering (Defns)

“The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software” [IEEE 1990]

“SE is the application of science and mathematics by which the capabilities of computer equipment are made useful to man via computer programs, procedures and associated documentation” [Boehm 1981]
Systems Engineering

- Systems Engineering takes a broader view of the process of developing software; it is the activity of understanding, specifying, designing, integrating, testing and deploying socio-technical systems.

- That is, systems used by people to support some kind of business or operational process and in which some of the components are software controlled computers.
Software Engineering Disciplines

- Computer Science
- Engineering
- Mathematics
- Project Management
- Economics
- Psychology
- Domain Knowledge (many disciplines)