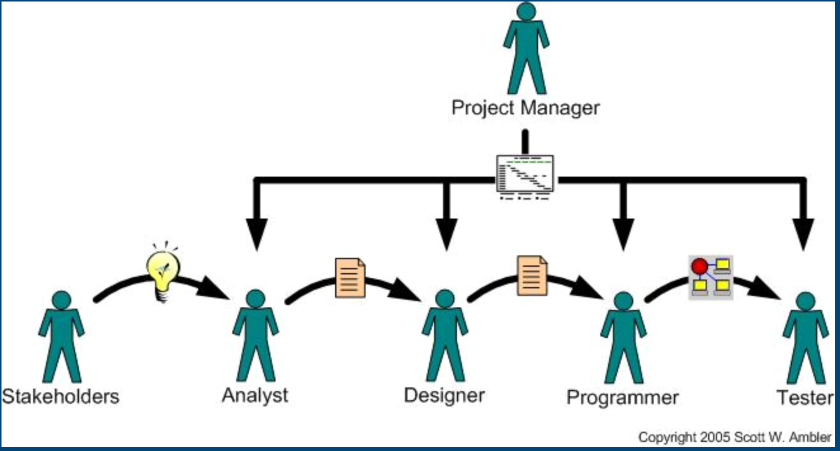


Software Engineering & Project

(COMPSCI 3006)

Lecture Outline

- General Information
- What is Software Engineering?
- The Robot Project

Teaching Team

- **Lecturer**
  - Amali Weerasinghe (course coordinator) Sarah Bernard (amali.weerasinghe@adelaide.edu.au)
  - David Milanese (david.milanese@adelaide.edu.au)

- **Tutors**
  - Sarah Bernard (sarah.bernard@student.adelaide.edu.au)
  - Shifa Haidry (shifa.haidry@student.adelaide.edu.au)
  - Gabriel Maicas (gabriel.maicas@adelaide.edu.au)
  - Sarah Bernard (sarah.bernard@student.adelaide.edu.au)
  - Tao Zhang (tao.zhamg01@student.adelaide.edu.au)

CSforums

- All information is provided on the canvas including:
  - Course outline
  - Schedule, lecture slides and readings
  - Deliverables and deadlines
  - Project specification and other supporting documents
  - Announcements and discussions
  - Other useful information

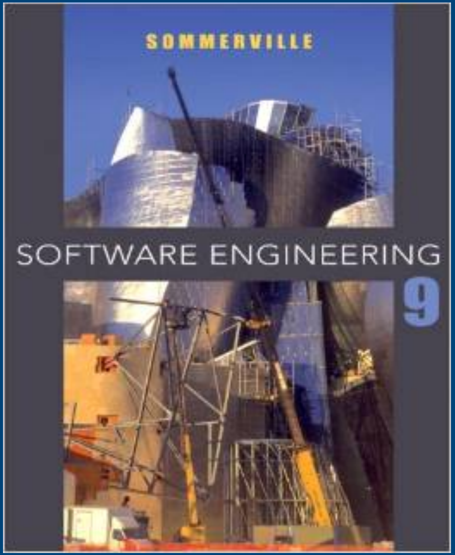
- It is YOUR responsibility to subscribe Announcements in canvas
  - And check it regularly – we will not chase you down.

The textbook and references

Text book

- Software Engineering 9th Ed.

  I. Sommerville, Addison-Wesley



Reference books

- Software Engineering: A practitioners approach, 5th Ed., R. Pressman, McGraw-Hill, 2001.
- Object-Oriented and Classical Software Engineering, 5th Ed., S. Schach, McGraw-Hill, 2002.
- Software Engineering Principles and Practice, 2nd Ed., H. VanVliet, Wiley, 2000.
- A Discipline for Software Engineering, W.S. Humphrey, Addison-Wesley, 1995.
- Managing Technical People, W.S. Humphrey, Addison-Wesley, 1997.
- Introduction to the Team Software Process, W.S. Humphrey, Addison-Wesley, 2000.

Teaching Arrangements

- Lectures
  - Monday 1-2pm Barr Smith South, 3029, Flentje Lecture Theatre
  - Wednesday, 10am-11am Johnson, G29, Rennie Lecture Theatre
  - Friday,11am-12pm, Physics, 103, Kerr Grant Lecture Theatre
  - Run from Week 1 to 8

Teaching Arrangements

- Involved in a major project
  - Performed in a group of 6-7
  - Control of a simple robot (LEGO Mindstorms EV3)
  - Not only code-base, but several main documents need to be produced
  - Deadlines and specifications will be listed on the course web site

- Weekly meetings
  - 25-minutes with teaching staff, from Week 3
  - At least another half-hour internal meeting without staff.

Assessment

- Three assessment components:
  - Two-hour Exam: **50%** of final mark
  - Group Project: **40%**
  - Individual Assessment Components **10%**
- Minimum Performance:
  - To avoid your grade being capped at **44F** under the school's minimum performance rule, you must achieve **at least 40%** in the exam and 50% overall for the course.
  - **Sign off** all project deliverables submitted by your group,
  - Sit the examination.
  - Submit what we actually ask for **in the form we ask for** it.
    - E.g. documents produced with LaTex etc., submissions via web submission

Assessment – Group Project

- Project mark is broken down as follows:
  - Project documentation: **57%**
    - Project Management Discussion
    - Software Requirements Specification (SRS)
    - Software Project Management Plan (SPMP)
    - Software Design Document (SDD)
    - Testing Report
    - User Manual
  - Other Group work: **10%**
    - Milestones1 & 2
  - Final Presentation and Demonstration: **33%**
  - For details, please visit the **course outline**

The best performing group will be awarded a prize!

Assessment - Individual

- Individual Assessment:
  - Presentations
    - Each student is expected to give one individual presentation during the course.
    - 5-6mins
    - This needs to be submitted as a YouTube video
    - For each presentation
      - Review a project deliverable (SRS)
      - Presentation
      - Self-reflection report

Individual Contributions

- Will be tracked via GitHub repository
- Group time sheets
  - A template will be provided
- Your project mark will depend on your contribution to the project

© 2004, The University of Adelaide

Workload

- Remember: This is the subject that employers regard most highly.
- You should expect to spend (at least) 120 hours on this course during the term
- This means the practical work should take you about 96 hours.
- These are ***productive* hours**:
  - if you are pretty poor at design, coding, debugging etc. – expect to spend more hours…

- **This is a fast paced course. You have to be fully committed from Week 1.**

Lecture Outline

- General Information
- **What is Software Engineering?**
- The Robot Project

What is Software Engineering?

- Not just programming!
  - Covers *all* aspects of the systematic construction of software with a specific purpose

    “multi–person construction of multi–version software”

    David Parnas
- Large Software Projects
  - Usually a collection of programs used over a long period
  - 25,000 to 10,000,000+ lines of code
  - Development teams of 10–200 people
  - Maintained by many generations of staff
- Never completely understood by one person

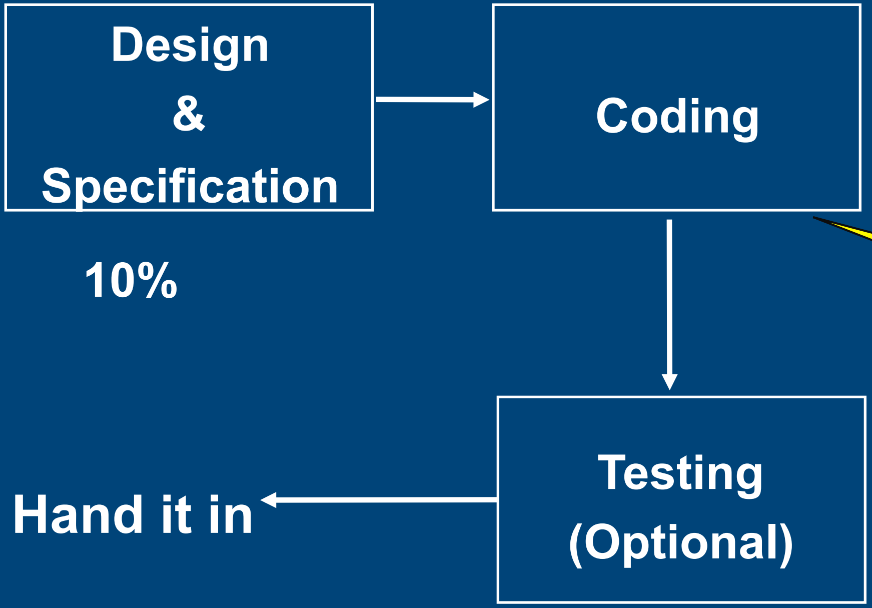
Problems for Software Engineers

- Dealing with large systems
- Software complexity
- Project complexity
  - Interactions with clients
  - Costing and management
  - Meeting deadlines
- Organisational problems
- Challenges of current SE practice:
  - Lack of proper understanding of software engineering
  - Lack of good metrics and measurement tools
  - Lack of good understanding of the standards
  - Lack of good tools to guide SE practices

Why Software Engineering?

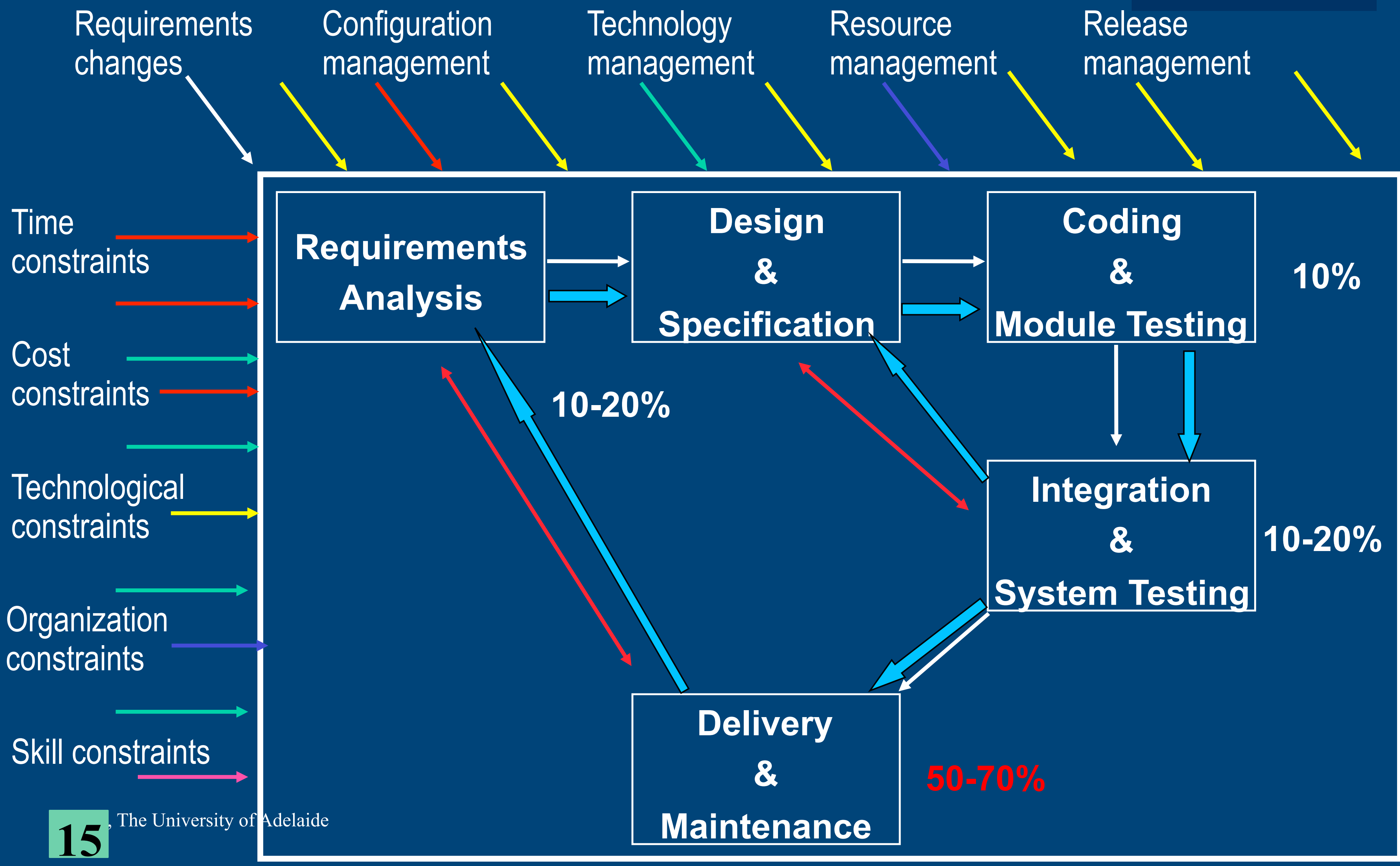
- Software is pervasive
  - Banks, shops, cars, institutions, home ...
- Software is by no means "perfect"
  - Malfunctions are common e.g.
    - spreadsheet program destroys database
    - computer controlled jail doors won't lock
  - and sometimes dangerous e.g.
    - Mercedes with "graceful-stop no-skid" computer controlled brake
      - 120m skid mark; 1 passenger dead
    - Airbus crash into a forest at air show
- Software projects are frequently out of control
  - US Navy engages GE for software for frigates in 1983
  - frigates built in 1986/7 but no software till 1988
  - GE Sacked, Nobel-Tech gets contract delivers in 18 months!

The Software Life Cycle “Student View”



Smart programmer is more important than anything else

The Software Life Cycle in Reality



Lecture Outline

- General Information
- What is Software Engineering?
- The Robot Project

The Project

- Specified in a hand-out (to be released on the course site)
- Specification is ***INTENTIONALLY*** incomplete
  - Requirements are highly uncertain, changeable at this stage which ***are very similar to the reality of many software projects in industry.***
    - It is solely your task, and responsibility to use requirements engineering techniques to get a good understanding of the requirements.
- Hardware:
  - Lego Mindstorms NXT robot
    - Kit contains an ‘Intelligent Brick’
    - Three servo motors
    - Light sensor and ultrasonic sensor
    - Sound sensor & two bump sensors.

  - Robots have been re-flashed to use *leJOS,* to allow programming in Java

Software Eng. Lab

- You will be given access to the software engineering lab: Ingkarni Wardli Room 4.62.
- **This lab is only for current Software Engineering and Project students.**
- Each group will have a numbered locker in which they can store their lego-kit or robot
- Six project meetings will run in parallel during some time slots (starting from week 3)
    - Two of the group meetings will be held here

The Project: Process

- Requirements phase:
  - What functionality is required
  - What hardware/software must be used
  - What can be achieved in the available time
  - **Specify** what will actually be attempted
  - Analyse the characteristics of the project based on the requirements
    - Analysis of Risk, Reliability, and Safety issues
- Design phase
  - Including interface design, architecture design, coding, code review, testing
- Planning
  - Develop a plan to address concerns issues, milestones, cost estimation, risk analysis, etc.
- Demonstrate a prototype to show your understanding of the system and reflection on your experience

The Project: Process

- Project work must follow a *process model*
  - Select a process model that works for you. (you might use more than one model)
  - We will discuss processes in a later lecture.
    - Very clear about what process model are you going to use in the project
      - Justify the selection
    - Use the practices, principles and engineering approaches or methodologies that we taught in the course
  - Organise your group
    - Groups must organise *themselves – You have to know your responsibilities in the group*
  - Plan your project at the beginning
    - Project management plan has to be written as early as possible
  - Coding is NOT the most important aspect!!
  - How you go about producing the software is just as important !!

The Project: Timetable

- The schedule of deliverables is put on the course Web site.
- You are responsible for planning your project
  - One of the deliverables is a *Software Project Management Plan in which we expect to see a plausible schedule...*
- There are compulsory deadlines we set
  - Milestones – designed to keep you “on track”
- There are compulsory deadlines that you must meet
  - You will be setting some group milestones early in the process and will be responsible for meeting those too
    - Failure to meet the milestones you set will incur a penalty
    - You will be able to re-negotiate them (although not the day before – see the schedule)

The Project: Meetings

- Starting from Week 3, each group will have a half-hour meeting with a lecturer.
- These are compulsory and ALL group members must attend.
- Absences will require the usual documentation: Medical certificate etc.
- All meetings must be accurately minuted. These minutes form part of the required documentation of the project.
- Some weeks will have mandatory agenda items which must be included (as specified on the schedule and by us from time to time)
- In these meetings, we will mainly act as your clients.
- Not the place to spring highly technical questions on us …

The Project: Tools

- You will need to become familiar with the tools very early on.
- All will be typical tools used in the industry, and some will be new to all of you. The required tools are discussed in **Section 5 of the Project Description.**
- Many of these are specified and **MUST** be used. Some tools include:
  - Development tools: Java, leJOS NXJ
  - You must use make and to ant to build your system
  - GitHub
  - Latex
  - Operation system: Unix , Windows, Mac
  - IDE (Integrated Development Environments)
  - Other Software Engineering Tools: Requirements, Planning, Testing, Configurations, etc.
- You will be given time to learn, however, time is very limited.
- Competence of using the tools will be assessed through documents, project presentation and demos

The Project: Documents

- All documentation for your project including the agendas and minutes must be produced using Latex.
- **Documents produced using any other software will not be accepted**.
- Latex is very easy to use but does require a small investment initially in learning it.
    - Resources will be supplied on the course website.
    - Start learning it early …

Groups

- To allow us to form the groups, we want enrolments stabilized asap.
  - **This means that you must enrol / (or unenrol) by ***5pm Wednesday, 26 July!**
  - **Allocating to group will be finalised by end of the week.**
  - **Anyone who wants to unenrol after group allocation needs a serious, convincing reason.**
- You will be assigned to a group of 6-7
  - To simulate what you expect in reality as much as we can
  - Often your team mates come from different backgrounds, both academic and culture
  - Developing skills on how to work with people