EBU5304 Software Engineering 2018-19

Introduction

EBU5304: Software Engineering

Lecturers:

- Group E-commerce and Group Telecom class 1-5
  - Dr Ling Ma(Course organiser):
    - Teaching Weeks 1+2
      ling.ma@qmul.ac.uk
  - Dr Matthew Huntbach:
    - Teaching Weeks 3+4
      matthew.huntbach@qmul.ac.uk
- Group IoT and Group Telecom class 6-10
  - Dr Gokop Goteng
    - Teaching Weeks 1-4
      g.l.goteng@qmul.ac.uk

QMPlus and Email

- **Course website**:
  - QMPlus → **https://qmplus.qmul.ac.uk**
  - Course Area: EBU5304 – Software Engineering – 2018/19
  - Check it regularly, as we could put there information related to e.g. *extra practice exercises*.
- **Email**:
  - You are expected to check your email regularly!
  - Use your QMUL email or BUPT email ONLY

Emails to the lecturers from other accounts are ignored.

Questions and feedback

- Message board:
  - Use the “Message Board“ forum activity in this course area.
  - For all general questions related to the module
  - Check existing discussions before you post a new question.
  - Notification is sent to QMUL email ONLY

Message board is the primary way of communication for this module

- Feedback:

  Give lecturers feedback immediately during or after the lecture. Do not wait until SSLC.

  Module reps!!!! Please!

Lecture arrangement

Four days Lectures and exercises

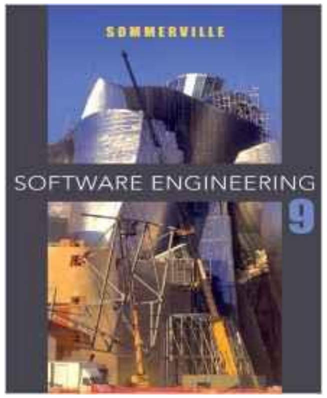
- Lectures: some notes on QM+ are blank
- Exercises: 15-20 minutes practice time every day.
  - Various activities
  - Could be anytime in the lecture
  - Solutions of the exercises will NOT be provided on QM+; ensure you participate to the activities to get them.

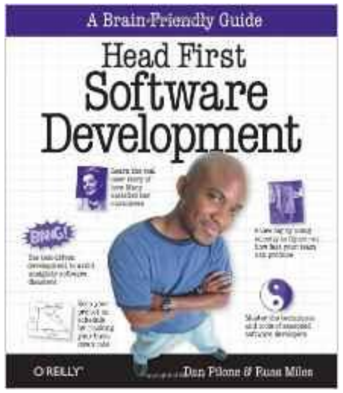
1 day workshop (split class): You only need to attend the 1 hour session.

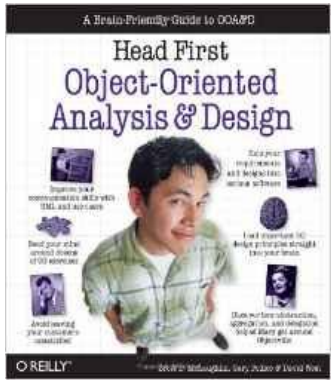
- Interactive workshop
- Demo and Feedback on your group coursework
- Arrangement will be announced in the week

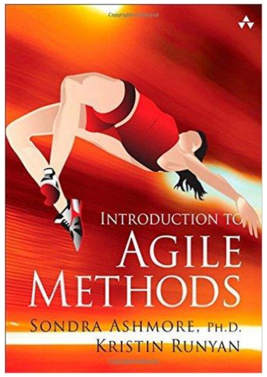
Recommended Textbooks

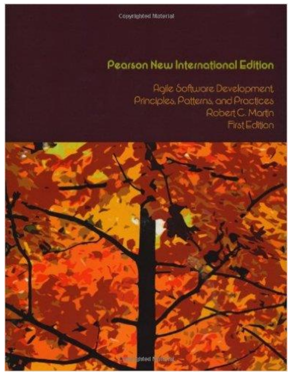
See the "Module Information" topic of the course website.

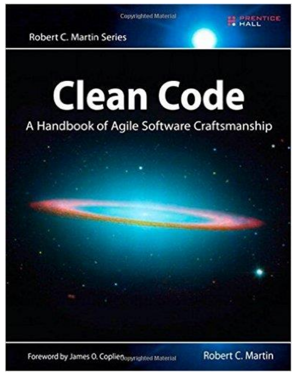


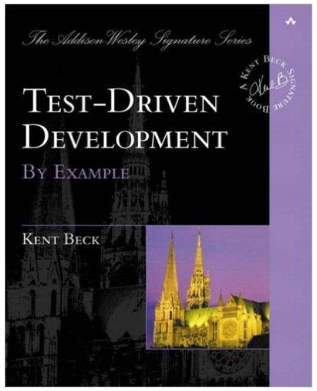



A Brain-Friendly Guide to OOA&D
Head First Object-Oriented Analysis & Design
O'REILLY




Pearson New International Edition
Agile Software Development Principles, Patterns, and Practices
Robert C. Martin
First Edition


Robert C. Martin Series
PRENTICE HALL
Clean Code
A Handbook of Agile Software Craftsmanship
Foreword by James O. Coplien
Robert C. Martin



Course Aims and Objectives

- The course provides:
  - An introduction to **modern** software development techniques necessary to produce high quality software and to manage the production of this software.
  - additional practice in program development
- The course aims to give each participant:
  - an idea of the necessity of good *software engineering practice* when developing *complex* software systems
  - knowledge of suitable software engineering *techniques* [in particular → practice in applying these techniques]
  - experience of *working in teams* to develop a product to a specification within strict deadlines [in particular → experience of *time-keeping*, which provides valuable experience for the final-year project]

Topics

- **Week 1**
  - Introduction to module & Software Engineering
  - Software Processes and Agile Overview
  - Requirements
- **Week 2**
  - Analysis and Design
  - Implementation and Testing
  - Project Management
- **Week 3**
  - Project Management (continues)
  - Design Principles
- **Week 4**
  - Design Patterns
  - Software Craftsmanship
  - Open Source software
  - Software Development Tools

Assessments

- **35% Coursework, made up of**:
  - Individual lab exercise: 5%
  - Group-based project: 30%
- **65% Final Examination**
  - Closed book exam, all compulsory questions.
  - **Duration**: 2 hours

Labs (assessed, 5%)

- Lab 1 – Java revision: programming exercises
- Lab 2-7 – Case study and develop software
- Lab 8 – Assessment (demonstration and file submission)

Lab Experiments: Executed individually in the computing lab, under the supervision of lab demonstrators, during a 2 hours timeslot.

- The two hours allocated for each lab will not be sufficient – you will have to spend more time each week working on your exercise.
- Individual assessment.

Group project (assessed, 30%)

- Group-based project
  - Groups (6 students) will be allocated and Coursework handout will be released in week 1.
  - No changes to the groups will be allowed.
  - Each group must appoint a leader (*project manager*).
  - You must meet all of your group members each week.
  - Submissions details will be published on QM+ in due course
- Assessment:
  - Demonstration 1(teaching week 2) 30%
  - Demonstration 2 (teaching week 3) 20%
  - Demonstration 3 (teaching week 4) 20%
  - Final submission (code and report) 30%

Plagiarism is strictly forbidden!

- What is it?
  - The reproduction of ideas, words or statements of another person without appropriate acknowledgement.
  - Examples:
    - A student knowingly permits another to turn in his/her work.
    - Presenting someone else's work as your own, without giving credit.
- All students must complete their own work <u>and</u> are expected to behave with integrity at **all** times.
- Plagiarism is strictly forbidden; *there are severe penalties* when detected!
- More information about this at the student handbook.

Any questions?

EBU5304 – Software Engineering

Getting started with Software Engineering

- Topics:
  - Overview of software
  - Introduce software engineering and its needs
  - The importance of software engineering

Queen Mary
University of London

Questions

Software = Programs ?

Software engineering = Programming ?

Software

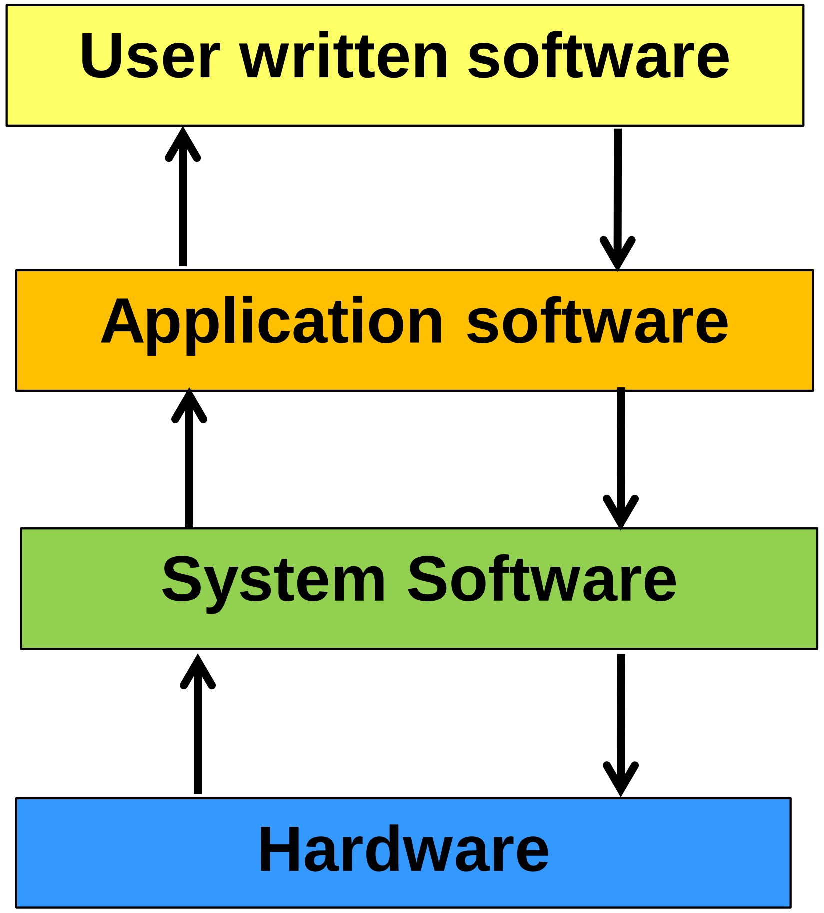
- Computer programs
- Data
- Configuration files
- Documentation
  - System documentation
  - User documentation/ manuals
  - Requirements
  - Design models
- Digital media

Computer software, or simply software, is a part of a computer system that consists of data or computer instructions, in contrast to the physical hardware from which the system is built.

In computer science and software engineering, computer software is all information processed by computer systems, programs and data. Computer software includes computer programs, libraries and related non-executable data, such as online documentation or digital media.
https://en.wikipedia.org/wiki/Software

Software types

- System software examples
  - Operating systems
  - Device drivers
  - Utilities
- Application software
  - Perform a specific task
  - Word processing
  - Image processing
- User-written software
  - Users' specific needs



Application Software types

- Stand-alone applications
- Web applications
- Phone App
- Embedded control
- Entertainment
- Modeling and simulation
- Data collection
- …

Generic vs Custom

- Generic Software
  - Developed for a general market
  - To be sold to a range of different customers
  - Example: Microsoft Office, Photoshop
  - Owned and controlled by the development organisation
- Custom software
  - Developed for a particular customer, according to their specific needs
  - Examples: software used in banks, airlines, embedded systems
  - Owned and controlled by the customer organisation

Generic vs Custom

- Trend (blur between generic and custom):
  – More and more, software companies are starting with a generic system and customising it to the needs of a particular customer.
- Examples:
  – University:
    - Moodle – generic
    - QMPLUS – for QMUL
  – Insurance company
  – Ticket booking

Problems…

THE PROJECT I INHERITED HAS WEAK CODE. I NEED TO REWRITE IT FROM SCRATCH.
WILL THERE EVER BE AN ENGINEER WHO SAYS, "THAT LAST GUY DID A GREAT JOB. LET'S KEEP ALL OF IT"?
I'M HOPING THE IDIOT YOU HIRE TO REPLACE ME SAYS THAT.
Dilbert.com DilbertCartoonist@gmail.com
8-12-14 ©2014 Scott Adams, Inc./Dist. by Universal Uclick

Good Software

Features of "Good" software?

```
This is an Intentionally BLANK slide. We will
discuss it in class and you can write down
notes.
```

Software Engineering

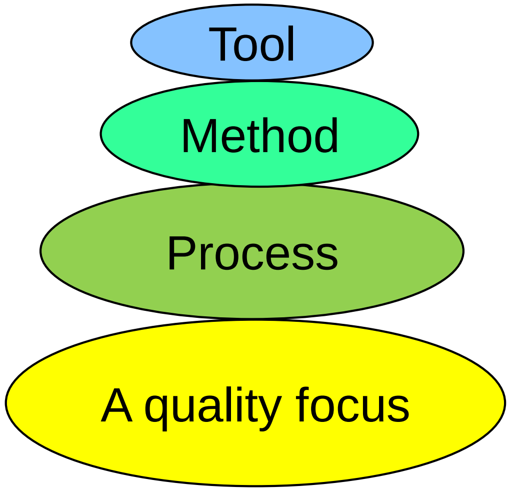
- An engineering discipline
  - Theories, methods, tools, constraints
- Concerned with all aspects for professional software production
- Goal: develop high-quality software
- Systematic and organised approach
- Use appropriate methods, tools and technologies
  - The problem to be solved
  - The development constraints
  - The resources available

Software Engineering

- "(SE) A systematic approach to the analysis, design, implementation and maintenance of software. It often involves the use of CASE tools. There are various models of the software life-cycle, and many methodologies for the different phases". [1]
- "Software is developed or engineered, not manufactured". [2]
- Process used to develop software; ultimate aim is to make software robust, easy to maintain etc.
- Process also allows software to be created by large teams, all working on the same plans, avoids software conflict etc.

[1] http://www.hyperdictionary.com/dictionary/software+engineering
[2] "Software Engineering", Pressman, 2000, McGraw Hill, pg.6

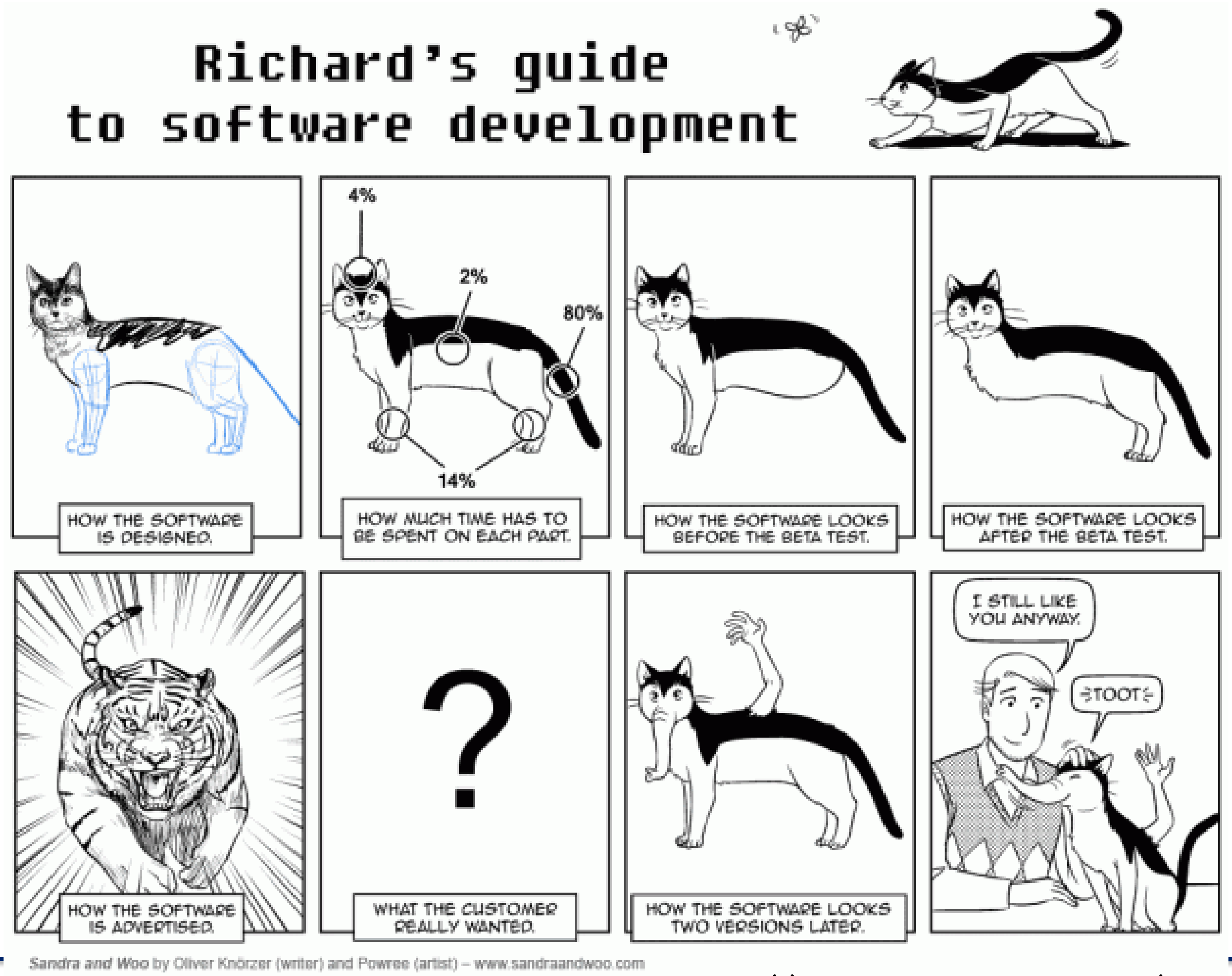
Software engineering layers



- Software engineering is a layered technology
  - Commitment to quality
  - Process: foundation layer
  - Methods: technical layer
  - Tools: support layer

Why do we need software engineering?

- Questions and issues:
  - Why does it take so long to get software finished?
  - Why are development costs so high?
  - Why can't we find all errors before we give the software to our customers?
  - Why do we continue to have difficulty in measuring progress as software is being developed?
  - Why do we create software that does not fulfil user requirements?


Richard's guide to software development
4%
2%
80%
14%
HOW THE SOFTWARE IS DESIGNED.
HOW MUCH TIME HAS TO BE SPENT ON EACH PART.
HOW THE SOFTWARE LOOKS BEFORE THE BETA TEST.
HOW THE SOFTWARE LOOKS AFTER THE BETA TEST.
HOW THE SOFTWARE IS ADVERTISED.
?
WHAT THE CUSTOMER REALLY WANTED.
HOW THE SOFTWARE LOOKS TWO VERSIONS LATER.
I STILL LIKE YOU ANYWAY.
TOOT
Sandra and Woo by Oliver Knörzer (writer) and Powree (artist) – www.sandraandwoo.com

http://www.jokeoverflow.com/

Software failures

- Large scale software development failure
- Small scale software development failure
- Read the examples on QMPLUS
  - National Air Traffic Services (NATS)
  - London Ambulance Service (LAS)
  - NHS IT System
  - Heathrow T5
- Why did these implementations (or systems) fail? How could this have been avoided?
- Any large scale software failure examples in China? (homework: find examples)

General issues that affect software

- Heterogeneity
  - distributed systems, across networks, different types of devices
- Business and social change
  - emerging economies develop, new technologies
- Security and trust
  - it is essential that we can trust that software
- Scale
  - Software has to be developed across a very wide range of scales

Summary

- Software
  - Types
  - Good software features
- Software engineering
- Software failure
- Issues

References and further reading

- Chapter 1 – “Software Engineering” textbook by Ian Sommerville
- “Software failure” examples – see course website