Professional Issues II: Software Development

Markus Roggenbach

January 2020

You will learn

- How the module is assessed.
- Which literature you can use.
- Where the module syllabus sits within computer science.

A. Introduction

Topics

- Qualities of professional code
  - Well documented
  - Well readable
  - Bug free
  - Tested
- Tools for professional SW design
  - Code documentation – javadoc
  - Style-Guide checking – checkstyle
  - Debugging – eclipse-debugger
  - Testing – junit

When & where

Lectures:

Monday 2pm & Thursday, 3pm, Great Hall GH043

Lab-classes: one of

- Monday 10am, CF 104
- Monday 11am, CF 104
- Monday 12noon CF 104

Labs

Two people work together on one PC (pair-programming).

Please look for your lab partner now –
though you can switch partners during term.

First labclass: Monday, 3.2.20

Assessment

- $\sim$ 8 lab-classes, each lab-class for $\sim$ 2.5 % (20%)
  You will be working in groups of 2
- 2 courseworks, each coursework 10% (20%)
- Written examination (60%)
  - Problem sheets will 'cover' the exam questions
  - Revision lectures (at the end of the course)

Background Reading

Buy (?):

- I Summerville: *Software Engineering*. 8th Edition, Prentice Hall, 2007.
- P C Jorgensen: *Software Testing: A Craftman's Approach*. 4th Edition, CRC Press, 2014.

Look at

- Tool documentations.

B. Off-task Media Use

What is it?

While engaging in academic activities

- students frequently interact with a variety of digital media
- these interactions are mostly off-task (unrelated to their academic work).
- off-task media use changes how students approach learning.

van der Schuur et al. reviewed 43 studies:
negative correlation between media use while studying or attending lectures and academic performance.

Quotes from focus groups

Norm among students: *"Sometimes I feel really bad actually, that I'm on my phone in class, and I'll stop and then I look around me, everyone else is on their phone".*

Need to stay connected / up to date *"Most of the time, I open it. I mean, it's like sitting right there, looking at me, I need to see what's happening."*

Awareness *"It's not like we don't know that we are doing the wrong thing. We're aware of the costs, but, at that point in time, that immediate satisfaction factor is just too high."*

Control over technology *"I have to put it away, otherwise I'll check it every two seconds – I have to put it somewhere else."*

My offer

Have the first 2 rows in the room device fee zone.

Further reading:
Douglas Parry and Daniel Le Roux: Off-task Media Use in Lectures: Towards a Theory of Determinants, Springer 2018. [available on blackboard]

C. Why such a module?

Observation 1: Software development is "regulated"

. . . , at least, when you want a certificate!

The necessity of unit testing

ISO 9000-3 recognizes that several types of testing may be necessary to adequately exercise a product, such as **unit**, *integration, system, and acceptance* **testing**.

from `http://www.mhhe.com/engcs/compsci/pressman/information/olc/ISO9000.html`

ISO 9001

- One of the standards in the ISO 9000 family.
- Internationally recognized standard for the quality management of businesses.

`ISO = International Organization for Standardization.`

Observation 2: Software is 'buggy'

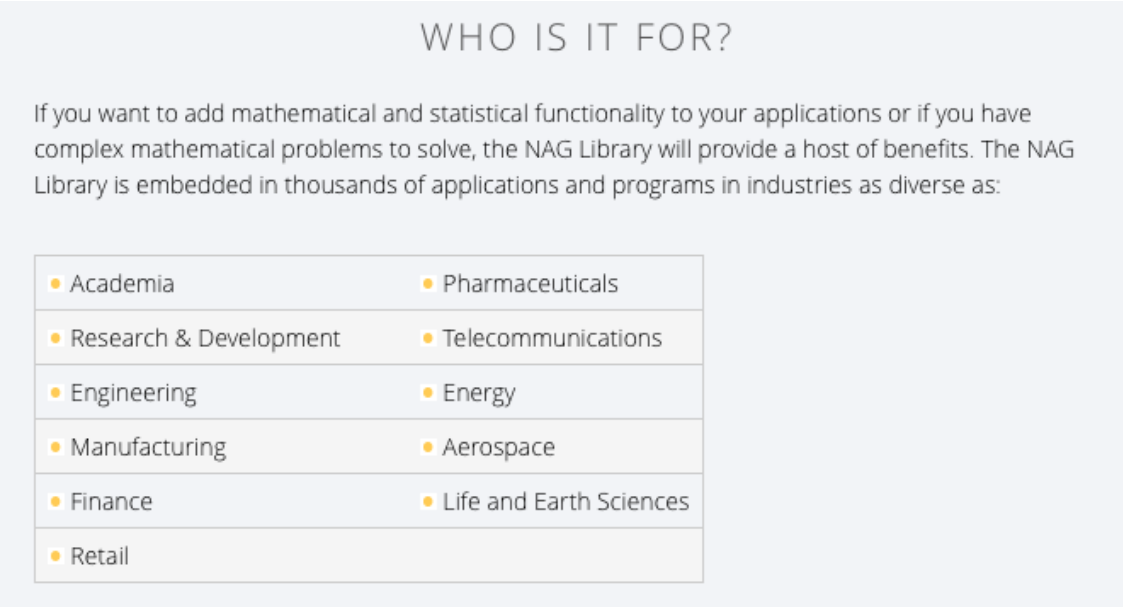
TOP 5 Software Failures of 2018–2019 https://blog.checkio.org/%EF%B8%8F-top-5-software-failures-of-2018-2019-5-is-pretty-alarming-2a5400b01658

- Facebook's apps outage
- CPUs flaw
- Crashed lunar lander
- British Airways glitch
- Self-driving killer car

Observation 3: Software is long in use

NAG Fortran Library

largest commercially available
collection of numerical algorithms
for Fortran today – can also be called, e.g., from Java



WHO IS IT FOR?

If you want to add mathematical and statistical functionality to your applications or if you have complex mathematical problems to solve, the NAG Library will provide a host of benefits. The NAG Library is embedded in thousands of applications and programs in industries as diverse as:

Academia	Pharmaceuticals
Research & Development	Telecommunications
Engineering	Energy
Manufacturing	Aerospace
Finance	Life and Earth Sciences
Retail	

- *The NAG project began in* **1970** *as a collaborative venture . . . between the Universities of Birmingham, Leeds, Manchester, Nottingham and Oxford, and the Atlas Computer Laboratory.*
- Latest release: Mark 26, October **2016**.
- NAG provides improvements to gridding algorithm for the Square Kilometre Array Radio Telescope – 10 November 2017.

`NAG = Numerical Algorithms Group`

from `http://www.nag.co.uk/numeric/FL/fldescription.asp`

Observation 4: maintenance cost dominates development cost

Year	software maintenance costs	Reference
2000	>90%	Erlikh (2000)
1993	75%	Eastwood (1993)
1990	>90 %	Moad (1990)
1990	60-70%	Huff (1990)
1988	60-70%	Port (1988)
1984	65-75%	McKee (1984)
1981	>50%	Lientz & Swanson (1981)
1979	67%	Zelkowitz et al. (1979)

Further data on maintenance

- costs: more than $70 billion in 2000 for the USA alone
- Diffent types of maintenance
  - Perfective maintenance – new functional or nonfunctional requirements
  - Corrective maintenance – fixing bugs
  - Adaptive maintenance – keeping up with changes in the environment

- Maintenance tasks:
  - 65% of maintenance: perfective – Lientz & Swanson (1981).
  - 75% of maintenance: adaptive or perfective maintenance – (Martin, 1983; Nosek & Palvia, 1990; van Vliet, 2000).
  - 50% of time spent in the process of understanding the code – (Fjeldstad & Hamlen, 1983; Standish, 1984).

- Legacy code amount to be maintained
  - 1990: estimated 120 billion lines of source code
  - 2000: about 250 billion lines of source code
  - Older languages are not dead. E.g. 70% or more of the still active business applications are written in COBOL.

Further reading: SoftwareMaintenanceCosts.pdf on Blackboard.

Observations lead to "Sustainable SE": Ease maintenance/reduce maintenance cost

- "well"-documented code $\leadsto$ javadoc
- "well"-written code $\leadsto$ checkstyle
- "well"-tested code $\leadsto$ junit

The current "software quality crisis"

- Late 1960's: "software crisis"
  - Characterized by: cost of software $>$ cost of hardware.
- Nowadays: "software quality crisis"
  - Characterized by: cost of verification and validation $>$ cost of programming

D. Software Engineering

Definition

systematic development of software products

Software product includes e.g.

- Requirement documents
- Design documents
- Program
- User manual
- System manual

Further reading

The term "software engineering" came into common usage as a result of the NATO Workshops on Software Engineering in 1968 and 1969. At that time the term was intentionally chosen as a provocation rather than as an indication of actual practice. During the intervening decade software engineering has evolved from a wish into a major subdiscipline of computer science and engineering. Although much remains to be done, a body of knowledge and a set of methodological guidelines are emerging which embody the application of traditional engineering values to the production and maintenance of software systems.

R Fairly: Educational Issues In Software Engineering. Proceedings of the 1978 annual ACM conference. ACM, 1978.

Software life-cycle

Described by various "models" such as

- Waterfall model
- V model
- Spiral model
- Extreme programming (XP)
- Scrum

ISO 9000 describes standards for formally organizing processes with documentation. $\rightsquigarrow$ Certification.

`ISO = International Organization for Standardization.`

Scope of the course: "Programming in the small"

- "in-the-small" – within one module
  Focus: how does this module work?
- "in-the-large" – using several modules
  Focus: how do the modules work together?

F DeRemer, H Kron: *Programming-in-the large versus programming-in-the-small*. Proceedings of the international conference on Reliable software, pp 114–121 ACM, 1975.

What you have learned today

Definitions

- Software Engineering
- Software Product
- Software Lifecyle
- Software Maintenance
  - perfective maintenance
  - corrective maintenance
  - adaptive maintenance

Examples discussed include

- NATO Workshops on Software Engineering in 1968 and 1969.
- ISO 9001 – standard for the quality management of businesses.

You should be able to explain by example

- what is perfective maintenance?
- what is corrective maintenance?
- what is adaptive maintenance?