Université de Paris

Génie Logiciel
Introduction to Software Engineering

Sylvain Lobry

17/09/2021

Resources: www.sylvainlobry.com/GenieLogiciel

Intro to Software Engineering

Notes on the labs

- Direct vs indirect consequences
- Man-hours
- Objectives of the lab
- Please connect to moodle and check that your group is correct
- If enough small exams (at beginning of the lab) -> 1 joker

Intro to Software Engineering

Wooclap

https://www.wooclap.com/L3GL2

Intro to Software Engineering

What is a Software Engineering?

- *IEEE Standard Glossary of Software Engineering Terminology*: “The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software”
- Arrêté ministeriel de 1983: “l'ensemble des activités de conception et de mise en œuvre des produits et des procédures tendant à rationaliser la production du logiciel et son suivi”

Intro to Software Engineering

What is a Software Engineering?

- Term coined by Margaret Hamilton: “*I fought to bring the software legitimacy so that it—and those building it—would be given its due respect and thus I began to use the term ‘software engineering’ to distinguish it from hardware and other kinds of engineering, yet treat each type of engineering as part of the overall systems engineering process*”.

- Historically, started in the October 1968 NATO conference on Software Engineering (Garmisch, Germany), chaired by Friedrich L. Bauer, Louis Bolliet and H. J. Helms

Intro to Software Engineering

What is a Software Engineering?

- Historically, started in the October 1968 NATO conference on Software Engineering (Garmisch, Germany), chaired by Friedrich L. Bauer, Louis Bolliet and H. J. Helms

- Domain created by a group of scientists as a reaction to 2 issues:
  - Most softwares are not reliable
  - Hard to finish in time a software that meets requirements.

- Prerequisite: programming

Intro to Software Engineering

Software Engineering Body of Knowledge (SWEBOK)

- Published by IEEE computer society (initiated in 1998, last version: 2013)
- International standard (ISO/IEC TR 19759:2005)
- Guide defining the body of knowledge of a Software Engineer
- 15 areas that every software engineer should know
- Was proposed as a formal requirements for the profession of software engineer
- Guide to SWEBOK downloadable for free on IEEE's website

Intro to Software Engineering

Software Engineering Body of Knowledge (SWEBOK)

- Software requirements
- Software design
- Software construction
- Software testing
- Software maintenance
- Software configuration management
- Software engineering management
- Software engineering process
- Software engineering models and methods
- Software quality
- Software engineering professional practice
- Software engineering economics
- Computing foundations
- Mathematical foundations
- Engineering foundations

Intro to Software Engineering

Back to wooclap

https://www.wooclap.com/L3GL2

Intro to Software Engineering

Software Engineering Body of Knowledge (SWEBOK)

- Software requirements
- Software design
- Software construction
- Software testing
- Software maintenance
- Software configuration management
- Software engineering management
- Software engineering process
- Software engineering models and methods
- Software quality
- Software engineering professional practice
- Software engineering economics
- Computing foundations
- Mathematical foundations
- Engineering foundations

Intro to Software Engineering

Software Engineering Body of Knowledge (SWEBOK)

- Software requirements
- Software design
- Software construction
- Software testing
- Software maintenance
- Software configuration management
- Software engineering management
- Software engineering process
- Software engineering models and methods
- Software quality
- Software engineering professional practice
- Software engineering economics
- Computing foundations
- Mathematical foundations
- Engineering foundations

Intro to Software Engineering

Software requirements

- Different types of requirements
- Analysis of the requirements
- Quantification, evaluation & validation of requirements

Intro to Software Engineering

Software design

- Architecture of the software
- User interface

Intro to Software Engineering

Software construction

- THE CODE
- Choice of a language
- Planning for reusability, changes
- Unit testing
- Profiling

Intro to Software Engineering

Software testing

- Different levels of testing
- Different types of tests
- Measurements of tests

Intro to Software Engineering

Software maintenance

- Cost of maintenance
- Different types of maintenance (technical, manangement)
- Techniques for maintenance

Intro to Software Engineering

Software configuration management

- Planing of the software configuration
- Control
- Auditing

Intro to Software Engineering

Software engineering management

- Planning
- Reviewing
- Closure

Intro to Software Engineering

Software engineering process

- Defintion of a process
- Software life cycles
- Measurement

Intro to Software Engineering

Software engineering models and methods

- Structure to software engineering
- Models as abstractions of software components
- Methods as an organization for a systematic approach

Intro to Software Engineering

Software quality

- Definition of a quality
- Management of quality
- Cost of quality

Intro to Software Engineering

Software engineering professional practice

- Legal issues
- Documentation
- Working in groups
- Communication

Intro to Software Engineering

Software engineering economics

- Basics of economy and finance
- Life cycles
- Economic analysis

Intro to Software Engineering

The foundations

- computing
- mathematical
- engineering

Intro to Software Engineering

Software Engineering Body of Knowledge (SWEBOK)

- Software requirements
- Software design
- Software construction
- Software testing
- Software maintenance
- Software configuration management
- Software engineering management
- Software engineering process
- Software engineering models and methods
- Software quality
- Software engineering professional practice
- Software engineering economics
- Computing foundations
- Mathematical foundations
- Engineering foundations

Intro to Software Engineering

"No silver bullet"

- Many tools, ideas, practices have been developed for software engineering
- *No Silver Bullet – Essence and Accident in Software Engineering* by Fred Brooks, 1987:
- Decomposes complexity of a software project between:
  - Accidental complexity, introduced because of development choices
  - Essential complexity, inherent to the problem
- Argues that accidental complexity has substantially decreased and that essential complexity harder to reduce
- No single technology can reduce by an order of magnitude essential complexity

Université de Paris

Génie Logiciel
Software quality

Sylvain Lobry

17/09/2021

Resources: www.sylvainlobry.com/GenieLogiciel

Definition of software quality

- ISO 9000:2015 standard
  - **“degree to which a set of inherent characteristics of an object fulfils requirements”**
  - Requirements can be implicit or explicit

- Side note: ISO = International Organization for Standardization

Software Quality

Examples of desirable properties

- **Fiability**: the software is valid (obtains desired outcome) and robust (able to cope with errors during the execution or erroneous input)
- **Safety**: when software is safety-critical, additional requirements
- **Performances**: expression of requirements in execution or processing speed and in terms of resources (e.g. memory)
- **Compatibility**: the software can be used with other softwares (either defined or meeting a specific format)
- **Portability**: the software can be used in various environments (e.g. OS)
- **Ergonomics**: the software can be easily (or naturally) used by the end-user
- Many more in few slides

Software Quality

I want it all!

- Quality comes at a cost. Non qualitative software as well -> Notion of "Cost of Software Quality"
- Often "level of quality in a software product can be inferred from the cost of activities related to dealing with the consequences of poor quality"
- 4 types of cost of quality:
  - Prevention (training, infrastructure, tools)
  - Appraisal (review, testing)
  - Internal failure (before delivery of the product)
  - External failure (after delivery of the product)

Software Quality

Software Quality Assurance (SQA)

- Set of activities to ensure that the software is of suitable quality (at every stage of the developement)
- In other words, SQA will check the conformity of the product to standards
- Often, people from SQA are independent from the project
- NOT testing, e.g.:
  - Checking that 90% of the code is unit tested -> SQA
  - Writing the unit tests -> testing

Software Quality

Standardization bodies

- International System Organization (**ISO**)
- Institute of Electrical and Electronic Engineers (**IEEE**)
- Association Française de Normalisation (**AFNOR**)
- American National Standard Institute (**ANSI**)

Software Quality

Quality standards – ISO 25000 series

- *Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE)*
- Published from 2005
- Framework for the evaluation of software product quality
- 5 parts:
  - 2500x: Quality Management
  - 2501x: Quality Model
  - 2502x: Quality Measurements
  - 2503x: Quality Requirements
  - 2504x: Quality Evaluation

Software Quality

Quality standards – ISO 25010

- Follows ISO 9126
- 8 **product quality characteristics**, each with *sub characteristics*:
  - **Functional suitability**: how well the software provides functions satisfying explicit and implicit needs. *Functional Completeness, Functional Correctness, Functional Appropriateness*
  - **Reliability**: under specific conditions, what functionalities? *Maturity, Availability, Recoverability, Fault tolerance*
  - **Performance Efficiency**: performances vs resources. *Time behaviour, Resource utilization, Capacity*
  - **Usability**: Effort needed for use and assessment of such use by users. *Appropriateness Recognizability, Learnability, Operability, User error protection, UI aesthetics, Accessibility*
  - **Security**: How well the system protects user and data from vulnerabilities. *Confidentiality, Integrity, Non-repudiation, Accountability, Authenticity*
  - **Compatibility**: Degree to which a product, system or component can exchange information with other products, systems or components, and/or perform its required functions while sharing the same hardware or software environment. *Co-existence, Interoperability*
  - **Maintainability**: to which extent the software can be modified to improve it, correct it or adapt it to changes in environment, and in requirements. *Modularity, Reusability, Analysability, Modifiability, Testability*
  - **Portability**: Can the software be transferred from one environment to another? *Adaptability, Installability, Replaceability*

Software Quality

Formal software inspection

- Definition from NASA-STD-8739.9 : “[…] Technical evaluation process during which a product is examined with the purpose of finding and removing defects and discrepancies as early as possible in the software life cycle“
- In general, formal inspection has the following characteristics:
  - Control is made by technically competent persons
  - Product’s author is actively involved
  - Involves the following persons:
    - Moderator
    - Reader
    - Recorder
    - Author

Software Quality

Formal software inspection

- 5 stage process:

1. Planification and presentation by the author
2. Preparation (inspectors study the product, potentially with control list(s))
3. Inspection
4. Rework
5. Follow-up

Software Quality

Control lists

- List of (technical) elements to be verified during the actual inspection
- For instance, one list per programming language
- Errors found should belong to the control list and identified
- Needs to be concise

Intro to Software Engineering

Conclusion

- To be a good software engineer: coding is only a part of the job
  - One part out of 15
  - Can be seen as a pre-requesite
- Objective: meet quality criteria