Faculty of ECMS / School of Computer Science

Software Engineering & Project Quality Management


Sales team presenting the solution in Powerpoint
Excited Customer
Egineering team knowing the solution is not technically possible

Outline

- Software Quality: An Overview
- Software Quality Management
- Quality Assurance and Standards
- Quality Planning
- Quality Control Techniques

Scope of Quality Management

For large, complex systems, particularly for safety-critical system:

- **Quality management** is particularly important
- Rigorous approach needs to be taken
- **Quality documentation** is required, recording progress and supporting *continuity* of development as the development team changes

For smaller systems:

- Quality management needs less documentation
- Focus on establishing a **quality culture**

Case Study: Ariane 5

https://www.youtube.com/watch?v=PK_yguLapgA

Case Study: Ariane 5

- A **software** component was **reused** in a new environment
  - Previous generations → Ariane 5
  - **Conditions** of functioning were **significantly different** from the initial requirements of the software component
  - **Requirements** had **not** been **revised**
  - Component was never properly **tested** in the new environment
    - Flaws in development and implementation could not be detected

https://hownot2code.com/2016/09/02/a-space-error-370-million-for-an-integer-overflow/

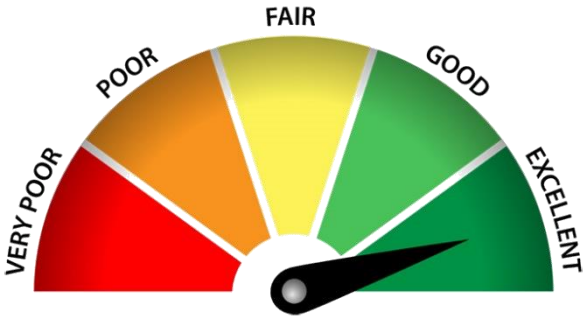
Case Study: Ariane 5

- The system identified and detected an error
  - The **specification** of the error-handling mechanism was **inappropriate** and caused the **final destruction**

→ Lack of **quality management** when transferring code from previous generations to Ariane 5

https://hownot2code.com/2016/09/02/a-space-error-370-million-for-an-integer-overflow/

Software Quality



- Quality
  - Product should **meet its specification**
- This definition is **problematic** for software systems:
  - There is a **tension** between customer **quality requirements** (efficiency, reliability, etc.) and developer quality requirements (maintainability, reusability, etc.)
  - Some quality requirements (e.g., maintainability) are **difficult to specify** in an unambiguous way
  - Software specifications are usually **incomplete** and often **evolving**
- The focus may be "**fitness for purpose**" rather than blindly trying to achieve specification conformance

Software Fitness for Purpose

- Questions to ask about your software system:
  - Have **programming** and **documentation standards** been followed in the development process?
    → http://google.github.io/styleguide/
  - Has the software properly **tested**? → unit/acceptance/integration testing
  - Is the software sufficiently **dependable** to be put into use? → trust through testing
  - Is the **performance** of the software acceptable for normal use? → benchmarks/profiling
  - Is the software **usable**? → user studies
  - Is the software **well-structured** and understandable?
    → software design pattern / anti-patterns / code smells
- Good quality software answers **"Yes"** to the above

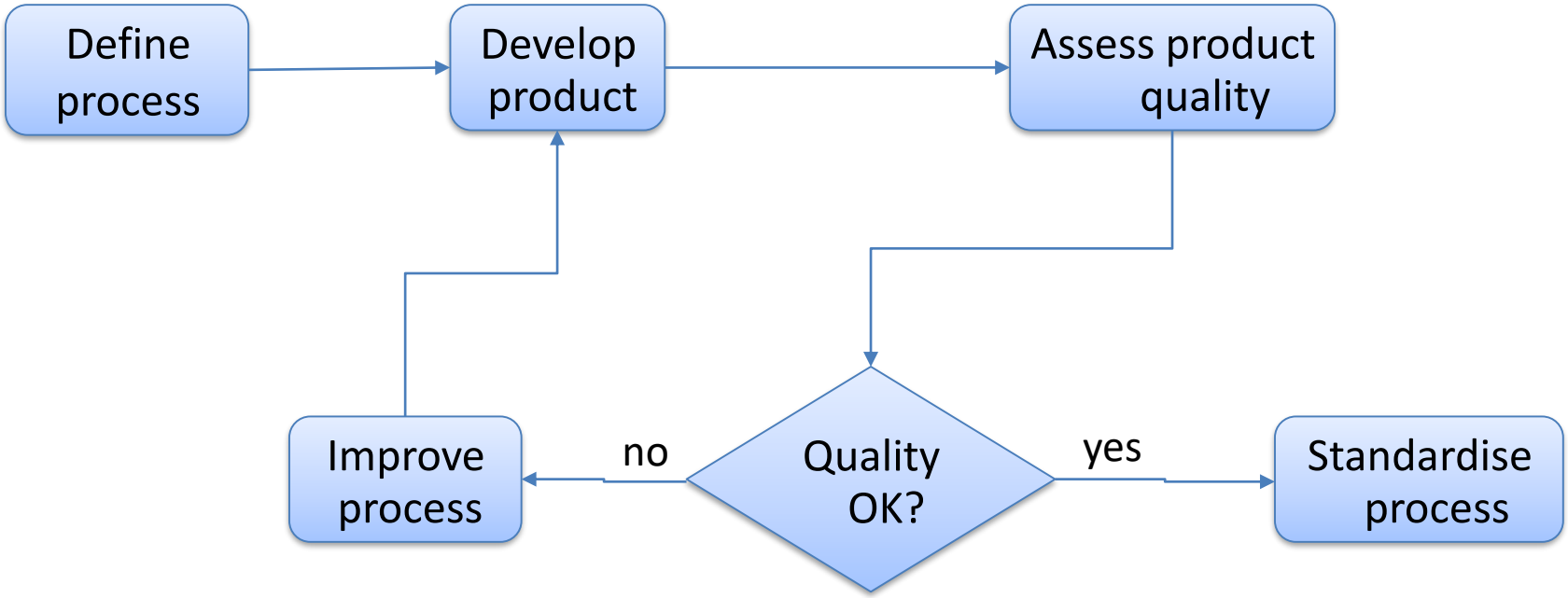
Software Quality Attributes

Safety	Understandability	Portability
Security	Testability	Usability
Reliability	Adaptability	Reusability
Resilience	Modularity	Efficiency
Robustness	Complexity	Learnability
Readability	Scalability	

- For **each software project**, the requirements of these attributes might be **different**!
  - There will always will be **conflicts** between attributes

Process and Product Quality

- The quality of a developed product is influenced by the quality of the **production process**



Quality Management Activities

- Quality **assurance**
  - Establish organisational procedures and **standards** for quality
  - See, e.g., the ISO 9000 standards
- Quality **planning**
  - Select applicable procedures and standards for a particular project and **modify** these as required
- Quality **control**
  - **Ensure** that procedures and standards are followed by the software development team

- Quality management should be adequately **separated from project management** to ensure independence

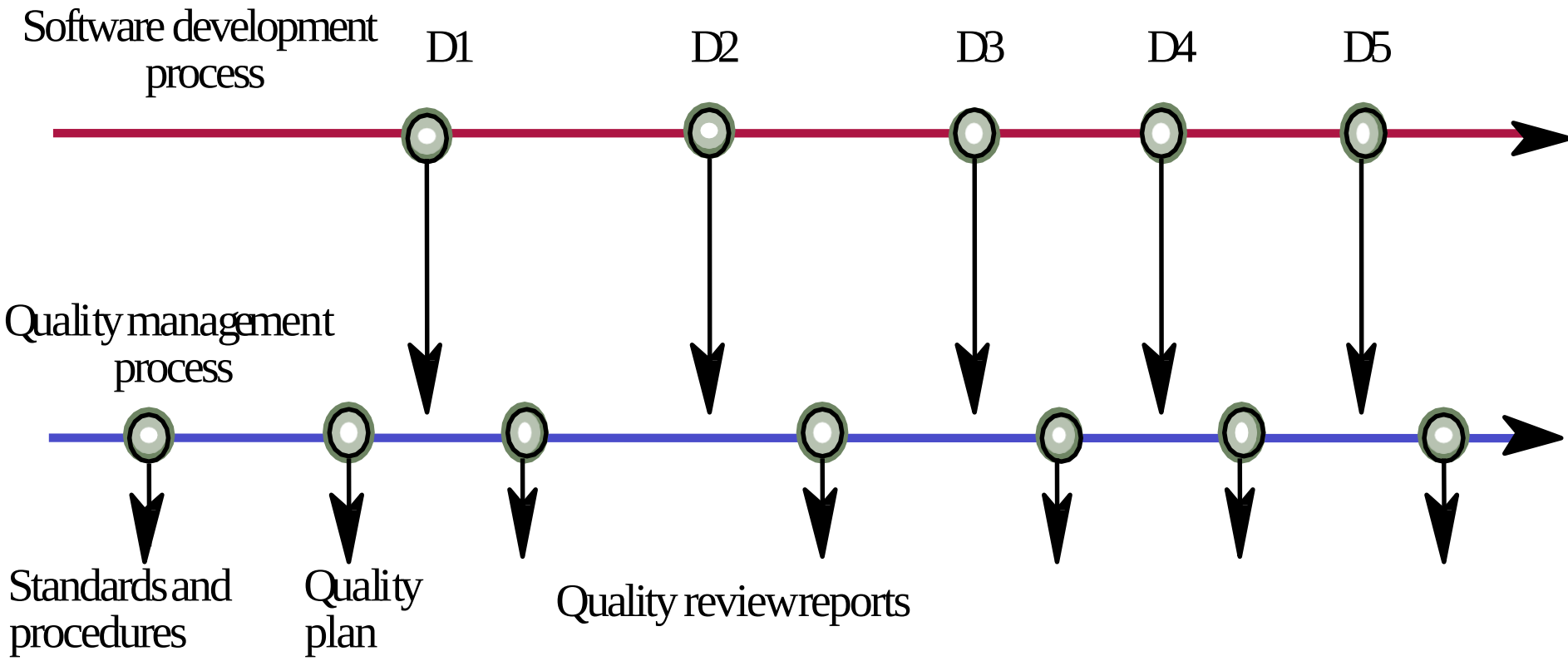
Goodhart's Law

- *"Any observed statistical regularity will tend to collapse once pressure is placed upon it for control purposes."*
  Charles Goodhart, British economist, 1975

- *"When a measure becomes a target, it ceases to be a good measure."*
  Marilyn Strathern, British anthropologist, 1997

https://en.wikipedia.org/wiki/Goodhart%27s_law

Quality Management and Software Development

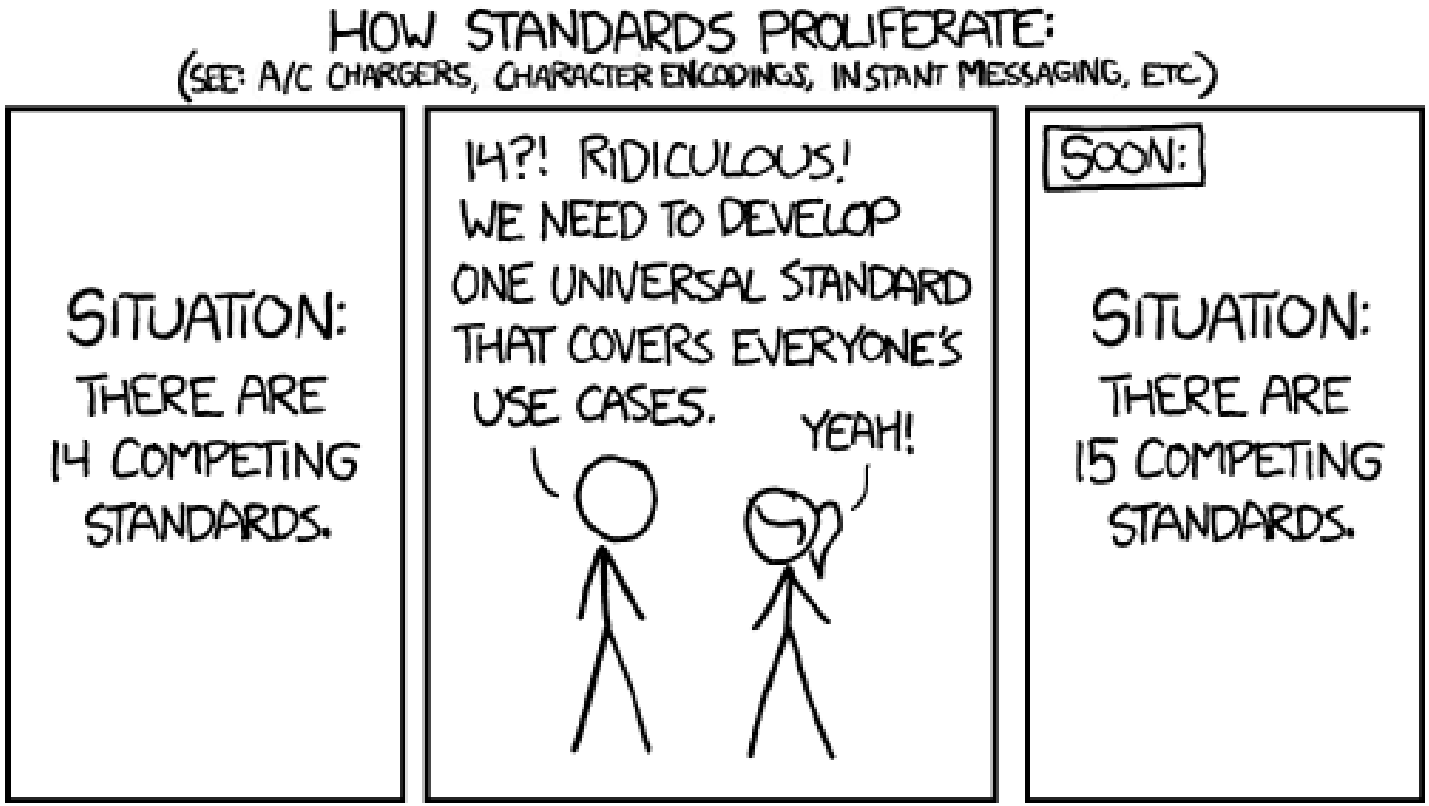


Quality Assurance and Standards

Standards are the key to effective quality management

- **Classification 1:**
  - International
    - ISO 9000, ISO 9001, IEEE, DoD, …
  - National
  - **Organizational**
- **Classification 2:**
  - **Product** standards
    - Apply to the software being developed, e.g. coding style, documentation standards, etc.
  - **Process** standards
    - Define how the software process should be enacted

Side Note: Standards



https://imgs.xkcd.com/comics/standards.png

Side Note: Standards

"Uncle" Bob Martin - "The Future of Programming"
https://youtu.be/ecIWPzGEbFc?t=430

Product and Process Standards

Product standards	**Process standards**
Design review **form**	Design review **process guidelines**
Requirements document structure	Submission of documents to CM
Method header format	Version release process
Java programming style	Project plan approval process
Project plan format	Change control process
Change request form	Test recording process

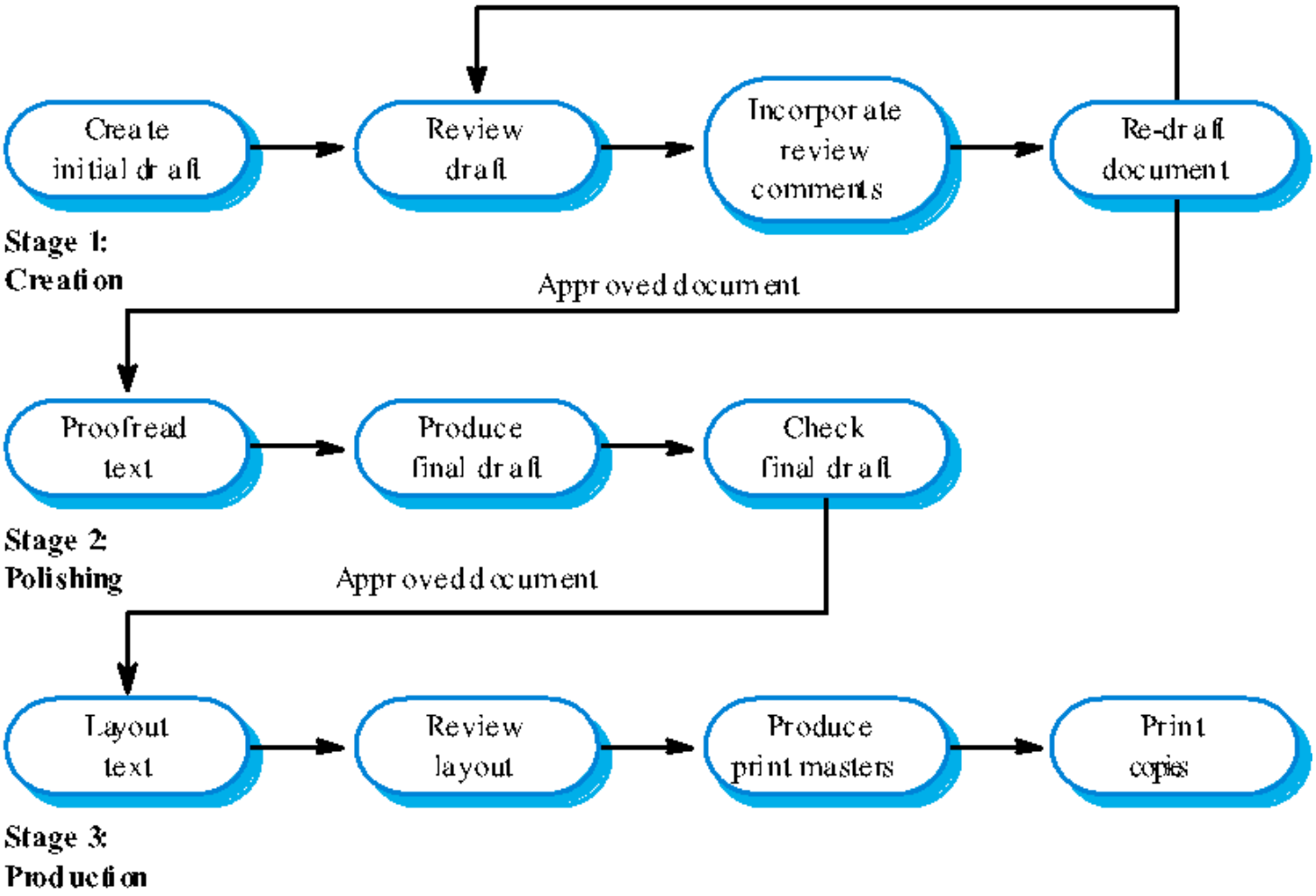
Importance of Standards

- **Encapsulation of best practice**
- Avoids repetition of **past mistakes**
- Help to define what quality means in a particular setting by providing a **framework**
- Standards provide **continuity:** New staff can understand the organisation by understanding the standards that are used

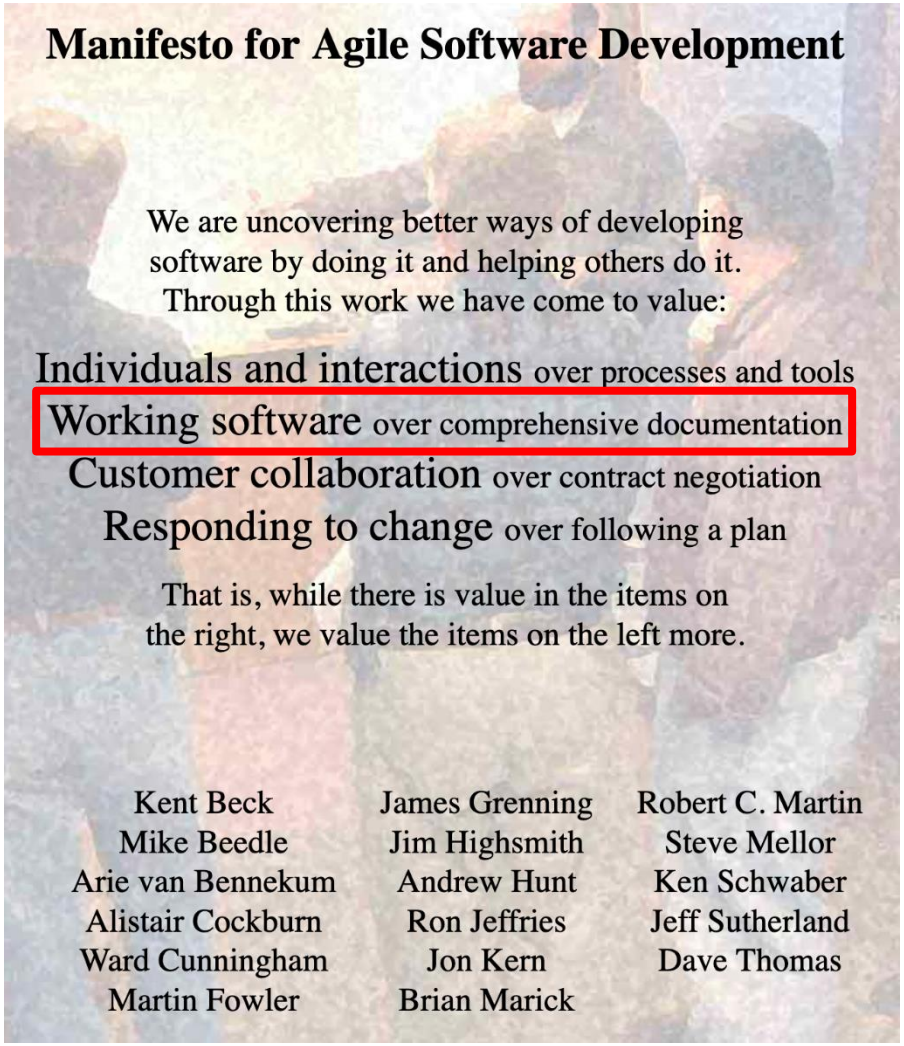
Documentation Standards

- **Documents** are the only **tangible** manifestation of the software
- **Standards** related to documentation:
  - **Process** standards
    - Concerned with how documents should be developed, validated and maintained
  - **Document** standards
    - Identification standards
    - Structure standards
    - Presentation standards (fonts, styles, use of logos, etc.)
    - Update standards
  - **Interchange** standards
    - Concerned with the **compatibility** of electronic documents

Documentation Process



Documentation Process



https://agilemanifesto.org/

Documentation Process – What not to do

- Writing tasks are only assigned to certain team members
- Individuals email their contributions to a "documentation manager"
  - Often a short time before deliverable is due
  - Often as a Word document → possible layout issues
- Documentation manager has to compile the document in a rush
  - Will often have to write bits not completed by team mates
- Minimal reviewing of the document is done

Coding Conventions

- Coding conventions are important for a number of reasons
  - 80% of the lifetime cost of software goes to **maintenance**
  - Very few software is maintained by the original author
  - Code conventions improve readability
    - New code can be understood more quickly
    - Easier to onboard developers
- Example: http://google.github.io/styleguide/

[Source: paraphrased from Sun coding convention]

Standards in Your Project

- We will **not** be expecting you to comply with any particular standard
- However, there are a number of **document templates** and guidelines (e.g. for SRS, SPMP, SDD)
  - These are informed by international (product) standards (IEEE)
  - You should use these as a **guide**
  - You may want to **adapt** them to your own needs

Quality Planning

- A **quality plan** sets out the **desired product qualities** and how these are **assessed** and defines the most significant quality attributes

- The plan should define the quality assessment **process**

- The plan should set out which **organisational standards** should be applied and, where necessary, define new standards to be used

Quality Planning

- Quality plan structure
  - Product introduction
  - Product plans
  - Process descriptions
  - Quality goals
  - Risks and risk management

- Quality plans should be **short succinct documents**
  - If they are too long, no one will read them

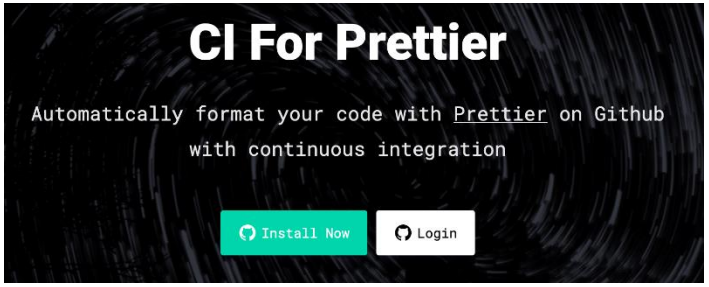
Quality Planning

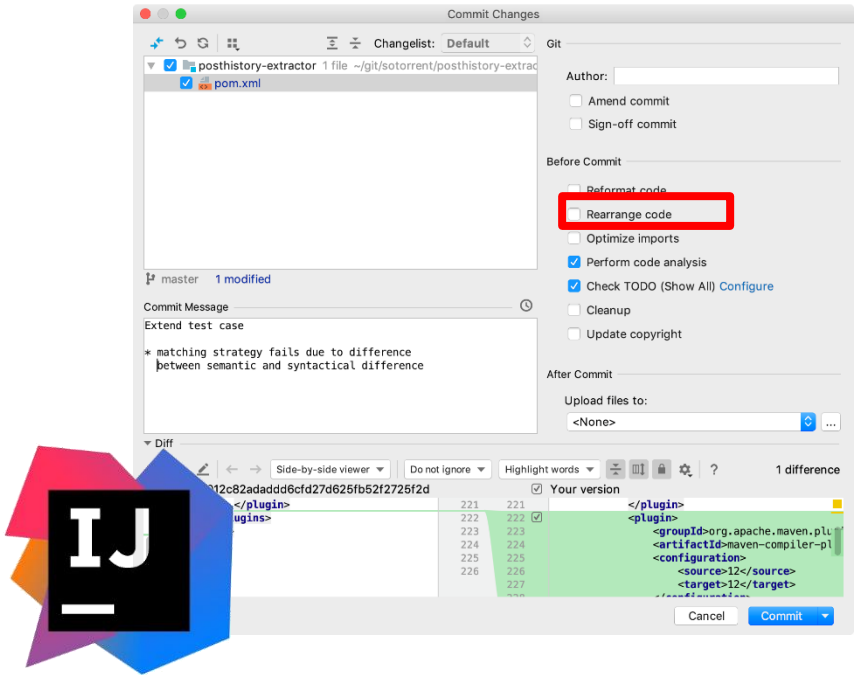
- Objectives to be achieve
  - cycle time,
  - cost,
  - resources etc,
- Operating practices in the process
- Responsibility allocation
- Specific documented standards and practices
- Testing and inspection practices and their stages
- A procedure for changes to a quality plan
- A method to check objectives

https://asq.org/quality-resources/quality-plans

Quality Control

- This involves **checking** the software development process to ensure that **procedures and standards are being followed**





- A widely used approach to quality control is **quality reviews**

Quality Reviews

- A **group of people carefully examine** part or all of a software system and its associated documentation, to find potential problems
  - Code, designs, specifications, test plans, standards, etc. can all be reviewed
  - Software or documents may be 'signed off' at a review which signifies that progress to the next development stage has been **approved** by management

- It is the **principal method** of validating the quality of a process or of a product

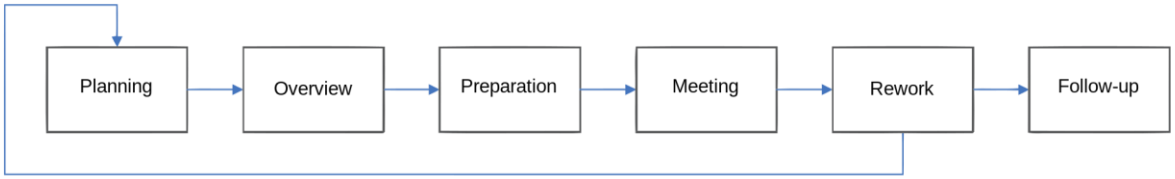
Review Results

- Comments made during the review should be classified
  - **No action**. No change to the software or documentation is required
  - **Refer for repair**. Responsible person (e.g., business analyst, designer, programmer) should correct an identified fault
  - **Reconsider overall design**. The problem identified in the review impacts other parts of the design. Some overall judgement must be made about the most cost-effective way of solving the problem (follow-up meeting with all stakeholders)
- Requirements and specification errors may have to be **escalated to the client**

Types of Reviews

- **Technical review**
  - Review for conformance to standards or achievement of project milestones
  - Led by team leader with **management** participation

- **Software inspection** ("Fagan inspection")
  - Peer review with formal process
  - Led by **independent moderator**
  - Systematic data collection
  - Focus on **fault detection** and description



Types of Reviews

- **Structured walkthrough**
  - **Less formal** than inspection
  - Usually led by **producer** (e.g., developer or business analyst)
  - No formal data collection
  - No participant preparation

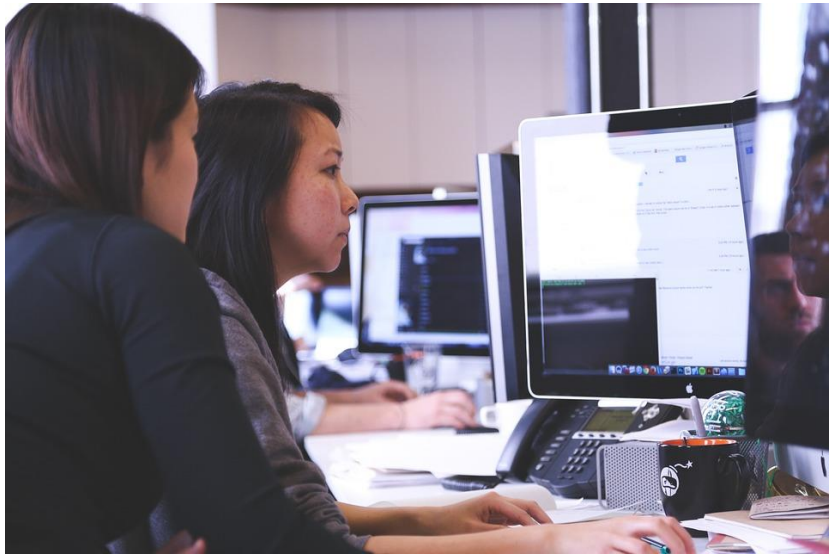
- **Audit**
  - **External review** of work product
  - Independently managed
  - Usually late in the process
  - Rare situation

Agile Approaches to Quality

- **Pair programming** (develop software artefacts as a team)
  - **Driver** writes the software
  - **Navigator** eyeballs the software looking for defects
  - Both contribute to the thought processes and problem solving
- Studies show a marked increase in quality
  - Especially in terms of defect rate
  - Strengthening the case for pair-programming (IEEE Software July/August 2000)
- **Cost** is **no where near double** that of single programmer

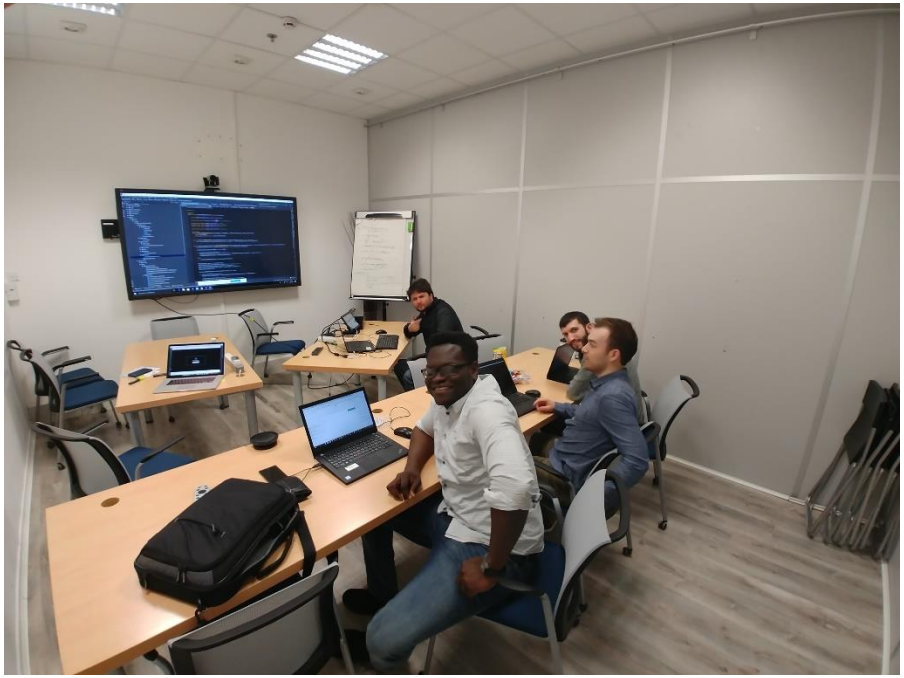
Agile Approaches to Quality

- **Test-driven development**
  - Test-first development
  - Incremental test development
  - User involvement in the test development and verification
  - Test automation and continuous integration
  - More on this in testing lecture



Agile Approaches to Quality

- **Mob programming**
  - the whole team works on the same thing, at the same time, in the same space, and at the same computer



https://en.wikipedia.org/wiki/Mob_programming

Relevance to your project

- In your project, think about:
  - What **software quality activities** have you planned and/or taken?
  - What **quality control** techniques have you used or plan to use in your project?
  - Which **techniques** do you want to use in your quality management process?
  - Standards/**principles** that you have followed / want to follow?

Key points revisited

- Software quality management is concerned with ensuring that **software meets its required standards**
- Quality assurance **procedures should be documented**
- Software **standards** are an encapsulation of **best practice**
- **Reviews** are the most widely used approach for assessing software quality
- All team members are responsible for software quality and documentation