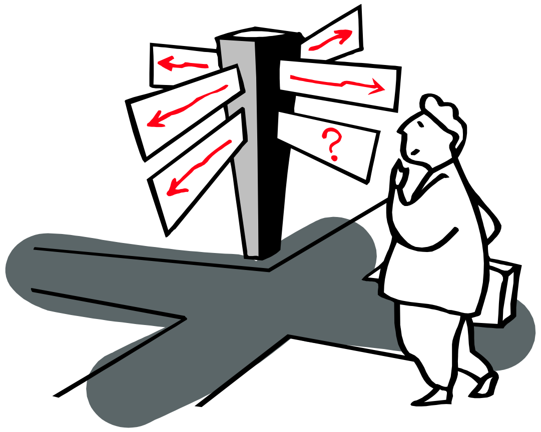
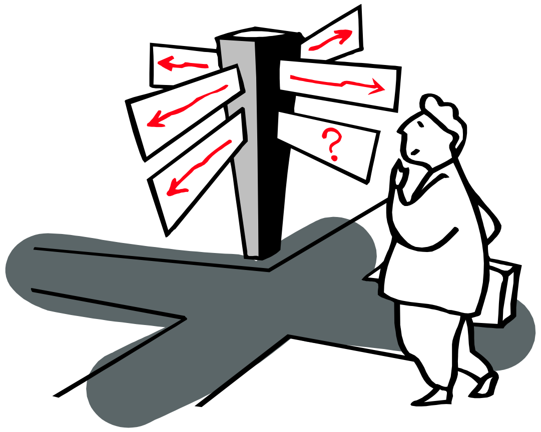
Universiteit restaurant: Kwaliteit of Niet?

CHAPTER 9 – Software Quality

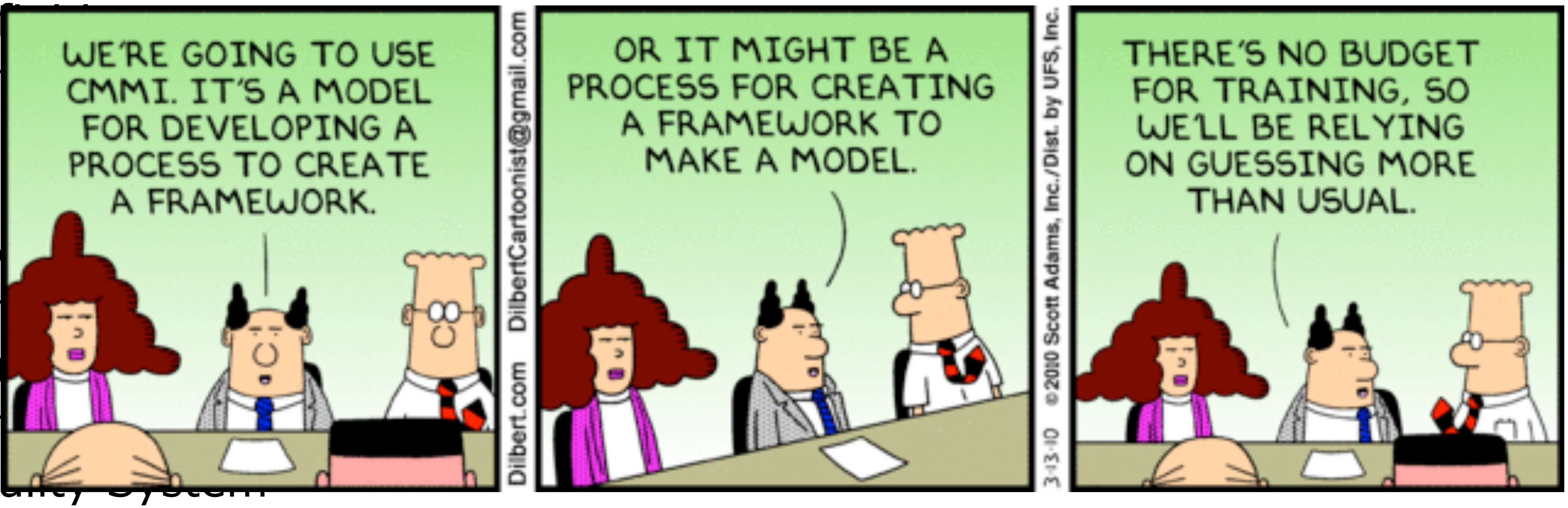


- Introduction
  + When, Why and What?
  + Product & Process Attributes
  + Internal & External Attributes
- Typical Quality Attributes
  + Overview
  + Definitions
- Quality Control
  + Quality Control Assumption
  + Quality Plan
  + Reviews & Inspections
- Quality in Scrum
  + Continuous Improvement
- Quality Standards
  + Quality System
  + ISO 9000, CMM, CMMI
- Conclusion

CHAPTER 9 – Quality Control



- Introduction
  + When, Why and What?
  + Product & Process Attributes
  + Internal & External Attributes
- Typical Quality Attributes
  + Overview
  + Def
- Quali
  + Qu
  + Qu
  + Rev
- Quali
  + Co
- Quali
  + Quality System
  + ISO 9000, CMM, CMMI
- Conclusion


WE'RE GOING TO USE CMMI. IT'S A MODEL FOR DEVELOPING A PROCESS TO CREATE A FRAMEWORK.
OR IT MIGHT BE A PROCESS FOR CREATING A FRAMEWORK TO MAKE A MODEL.
THERE'S NO BUDGET FOR TRAINING, SO WE'LL BE RELYING ON GUESSING MORE THAN USUAL.
Dilbert.com DilbertCartoonist@gmail.com
3-13-10 ©2010 Scott Adams, Inc./Dist. by UFS, Inc.

© dilbert.com

Literature

+ [Ghez02] In particular, chapter “Software: Its Nature and Qualities”
+ [Pres00] In particular, chapter “Software Quality Assurance”
+ [Somm05] In particular, chapters “Quality Management” & “Process Improvement”

- Web-Resources
  + ISO [http://www.iso.org/]
    - ISO 9000 family
      > [https://www.iso.org/iso-9001-quality-management.html]
  + CMM (Capability Maturity Model)
    - Paulk, Mark C.; Weber, Charles V; Curtis, Bill; Chrissis, Mary Beth (February 1993). "Capability Maturity Model for Software (Version 1.1)" (PDF). Technical Report. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University. CMU/SEI-93-TR-024 ESC-TR-93-177.
  + CMMI® (Capability Maturity Model Integration)
    - [https://cmmiinstitute.com]

Famous Quality Incidents

Tacoma Narrows Bridge

Mature engineering disciplines learn from their mistakes

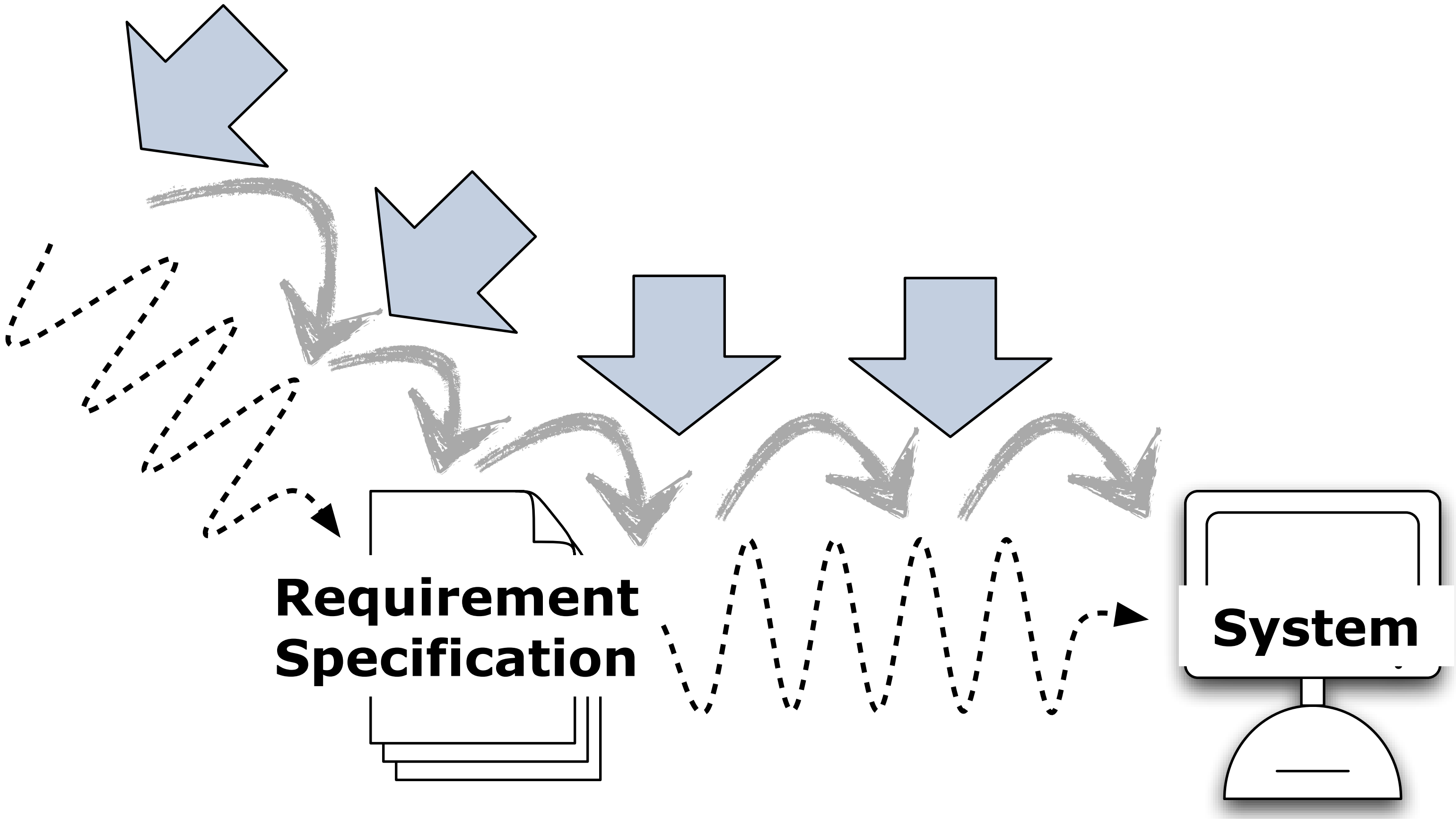
Denver International Airport Baggage Handling System

Ariane 5

SENTINEL
UNITED STATES DEPARTMENT OF JUSTICE
GLOBAL INFORMATION SHARING
TO PROTECT AND DEFEND THE NATION
FEDERAL BUREAU OF INVESTIGATION
DEPARTMENT OF JUSTICE
FEDERAL BUREAU OF INVESTIGATION

FBI Sentinel Project

When Quality Control?



quality in the final system ⇔ control quality of all intermediate steps.

However ... Quality control $\not\Rightarrow$ High-quality system

Quality control tries to *eliminate coincidence*
⇒ Quality control makes achieving quality *repeatable*

Why Quality Control?

Lives are at stake
(e.g., automatic pilot)

Huge amounts of money
are at stake
(e.g., Ariane V crash)

Software became Ubiquitous
Our society is vulnerable!

Corporate success or failure is at stake
(e.g., telephone billing,
VTM launching 2nd channel)

Your personal future is
at stake (e.g., Y2K
lawsuits)

Quality vs. Requirements

“Simplistic” Definition: Software Quality =

- Deliver
  + (a) what’s required
  + (b) on time
  + (c) within budget

> Cover quality in the “non-functional” requirements

Acoustics

Earthquake resistant

Quality ≠ Requirements

Consider requirements/implementation table

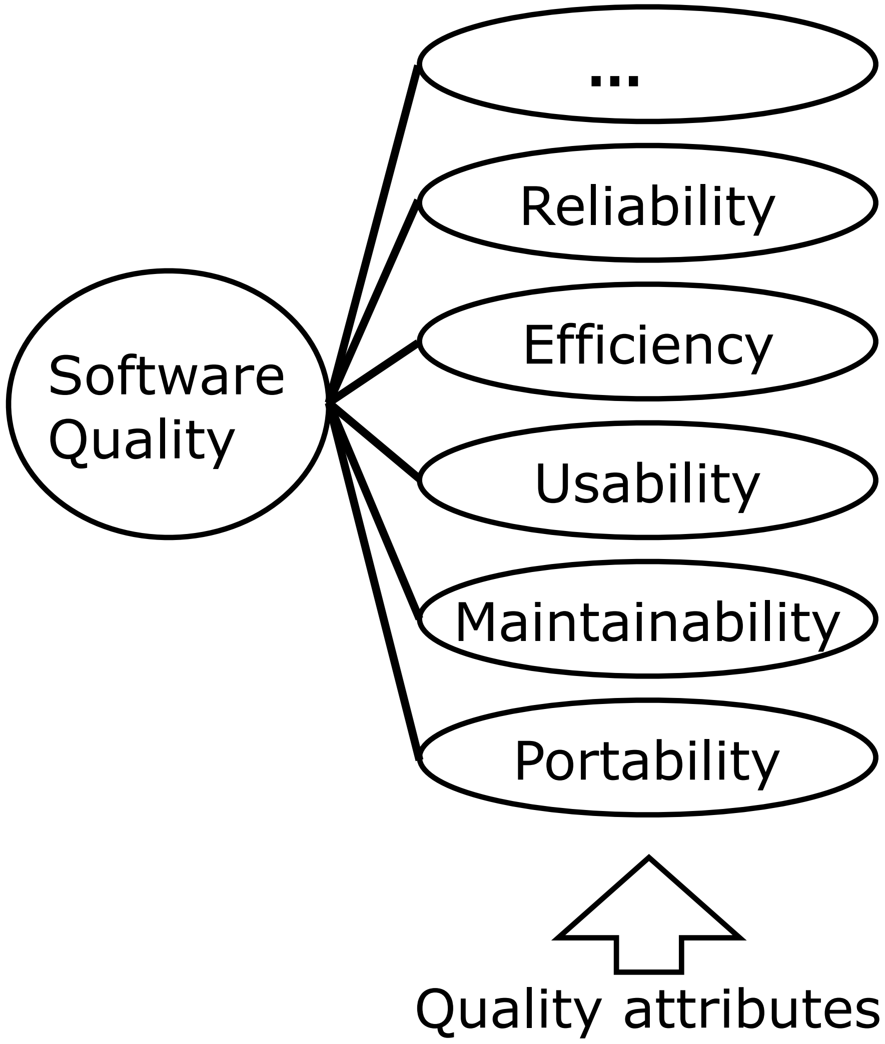
• Requirement	a car, quite cheap	a car, price unimportant
• Implementation		

> Both adhere to their specifications ...
> ... but do they have the same quality?

- Covering quality in the “non-functional” requirements is too simplistic
  - + How to assess the quality of the requirements?
    - “Are we building the right product”
      vs. “Are we building the product right”
  - + Development team has (implicit) requirements too
    - Maintainability etc. are usually not specified

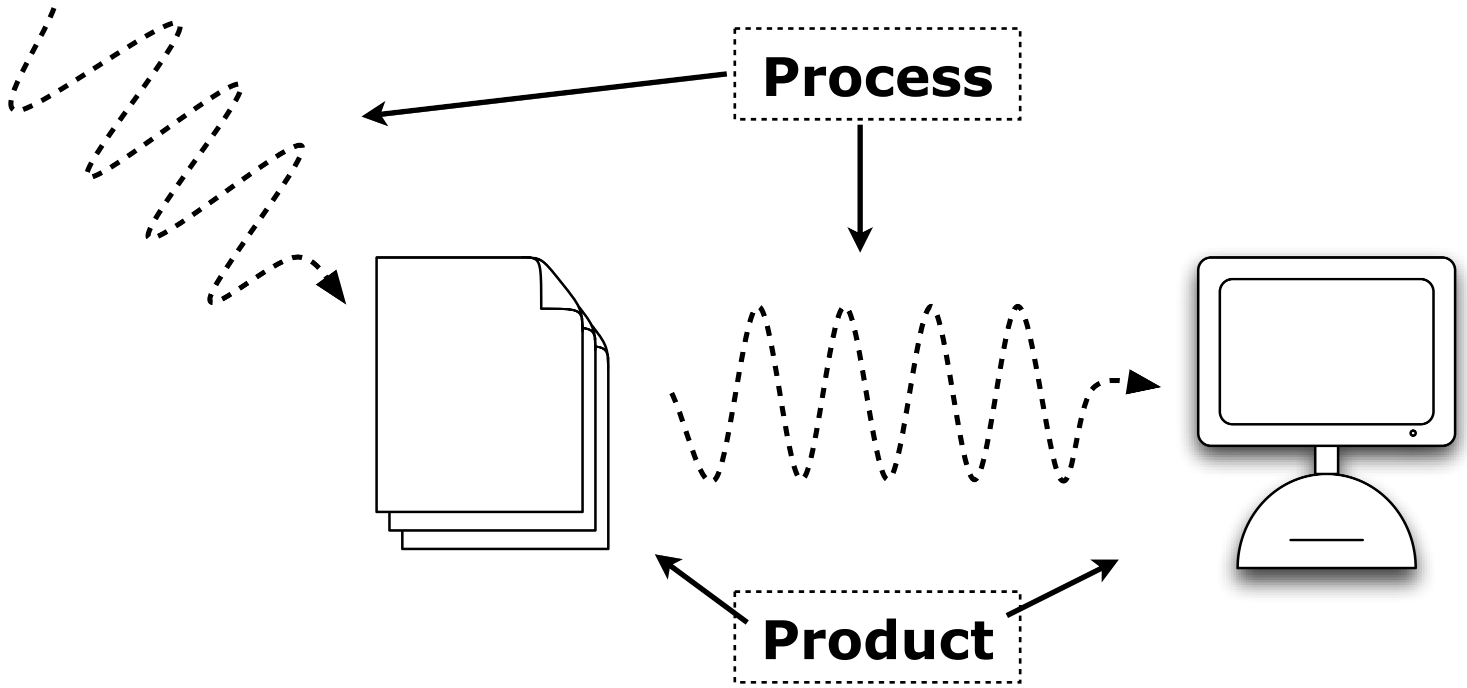
Hierarchical Quality Model

- To side step the quality vs. requirements discussion
  + Define quality via hierarchical quality model, i.e. set of quality attributes (a.k.a. quality factors, quality aspects, ...)
  + Choose quality attributes (and weights) depending on the project context
    > Nevertheless: variation of simplistic quality = requirements



- choose your own set of quality attributes
- may be further refined into subattributes, ...

Product vs. Process Attribute



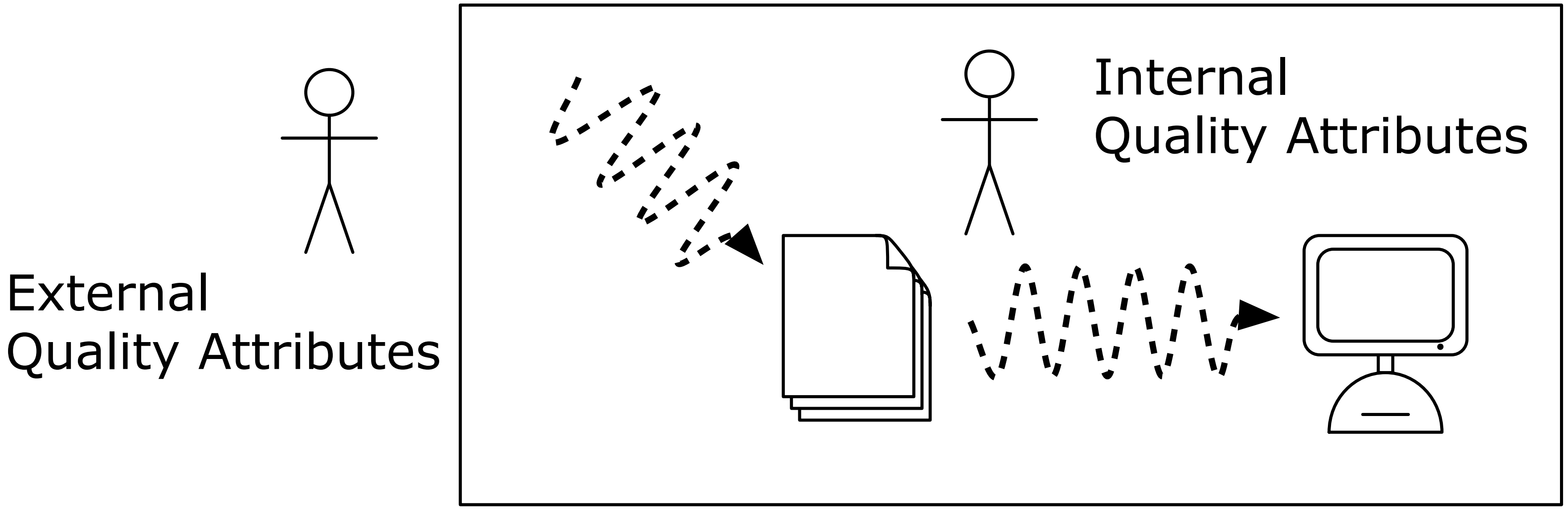
Quality attributes apply both to the product and the process.

- *product*: delivered to the customer
- *process*: produces the software product

Underlying assumption: a quality process leads to a quality product (cf. metaphor of manufacturing lines)

External vs. Internal Attributes

The distinction between the two is not as sharp as it seems!



Quality attributes can be external or internal.

- *External*: Derived from relation between environment and system/process.
  > To derive, the system or process must have *run to completion*.
- *Internal*: Derived immediately from the product or process description.
  > To derive, it is sufficient to have the *description*.

Underlying assumption: internal quality leads to external quality (cf. metaphor of manufacturing lines)

Quality Attributes Overview

See [Ghez02], section 2.2 Representative Qualities

		Product	Process	External	Internal
Product / External					
	Correctness, Reliability, Robustness	x		x	
	Efficiency	x	(x)	x	
	Usability	x	(x)	x	
	Maintainability	x		x	
	• Repairability	x		x	
	• Evolvability	x	(x)	x	
	• Portability	x		x	
Product / Internal					
	Verifiability	x		(x)	x
	Understandability	x		(x)	x
Process					
	Productivity		x	x	
	Timeliness		x	x	
	Visibility		x	(x)	x

Correctness, Reliability, Robustness

3 external product attributes

Correctness

- A system is correct if it behaves according to its specification
  - + An absolute property (i.e., a system cannot be “almost correct”)
  - + ... in theory and practice undecidable

Reliability

- The user may rely on the system behaving properly
- The probability that the system will operate as expected over a specified interval
  - + A relative property (a system has a mean time between failure of 3 weeks)

Robustness

- A system is robust if it behaves reasonably even in circumstances that were not specified
  - + A vague property (once you specify the abnormal circumstances they become part of the requirements)

(This slide is a copy from Chapter 5 — Testing)

Efficiency, Usability

2 external attributes, mainly product - sometimes also process

Efficiency (Performance)

- Use of resources such as computing time, memory
  - + Affects user-friendliness and scalability
  - + Hardware technology changes fast!
    - - (Remember: First do it, then do it right, then do it fast)
- For process, resources are man-power, time and money
  - + relates to the “productivity” of a process

Usability (User Friendliness, Human Factors, Human Engineering)

- The degree to which the human users find the system (process) easy to use
  - + Depends a lot on the target audience (novices vs. experts)
  - + Often a system has various kinds of users (end-users, operators, installers)
  - + Typically expressed in “amount of time to learn the system”

Maintainability

external product attributes (evolvability also applies to process)

Maintainability

- How easy it is to change a system after its initial release
  - + software entropy
    - > maintainability gradually decreases over time

Often refined in ...

Repairability

- How much work is needed to correct a defect (= corrective maintenance)

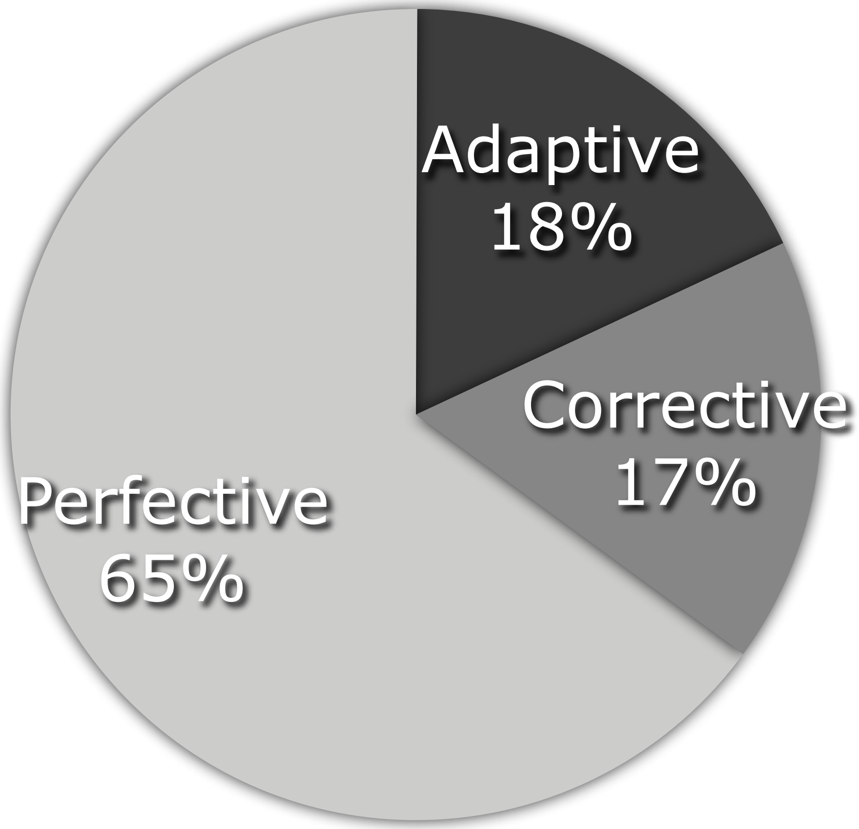
Adaptability (Evolvability)

- How much work is needed to adapt to changing requirements (= perfective maintenance)
  - > both system and process

Portability

- How much work is needed to port to new environment or platforms (= adaptive maintenance)

Maintenance costs



Verifiability, Understandability

internal (and external) product attribute

Verifiability

- How easy it is to verify whether desired attributes are there?
  + internally: e.g., verify requirements, code inspections
  + externally: e.g., testing, efficiency

Understandability

- How easy it is to understand the system
  + internally: contributes to maintainability
  + externally: contributes to usability

Productivity, Timeliness, Visibility

external process attribute (visibility also internal)

Productivity

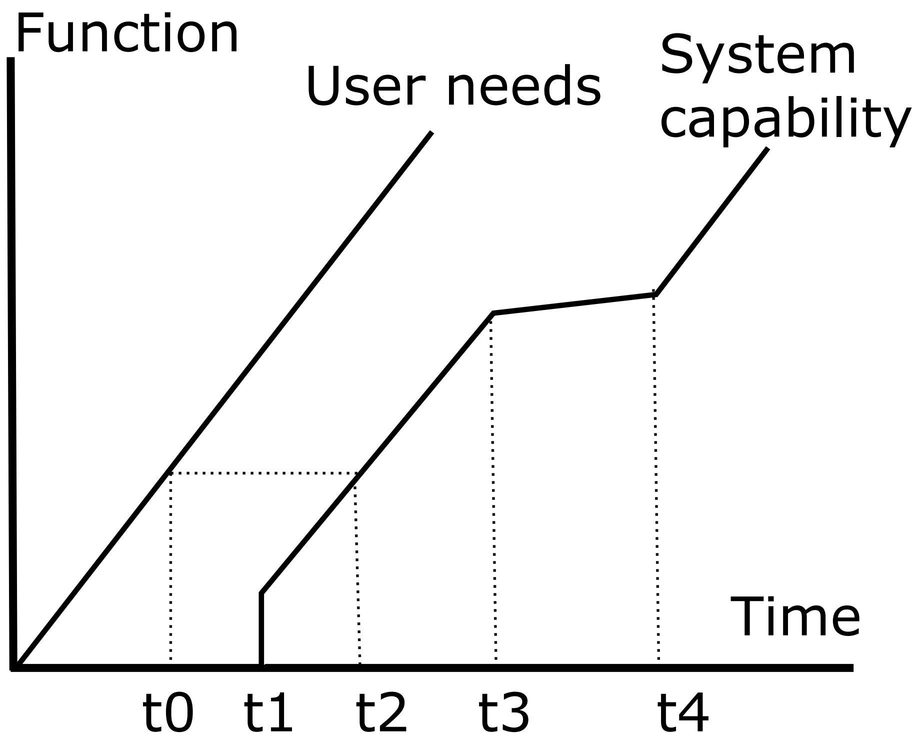
- Amount of product produced by a process for a given number of resources
  + productivity among individuals varies a lot
  + often: productivity (∑ individuals) < ∑ productivity (individuals)

Timeliness

- Ability to deliver the product on time
  + important for marketing ("short time to market")
  + often a reason to sacrifice other quality attributes
  + incremental development may provide an answer

Visibility (Transparency, Glasnost)

- Current process steps and project status is accessible
  + important for management; also deal with staff turn-over



Productivity, Timeliness, Visibility

What is meant with “short time to market”?

- Time to market = The time that is needed between requirements specification (feature, user story, use case) agreed upon and deploying said requirement in production
- Short time to market = we can release the requested features fast

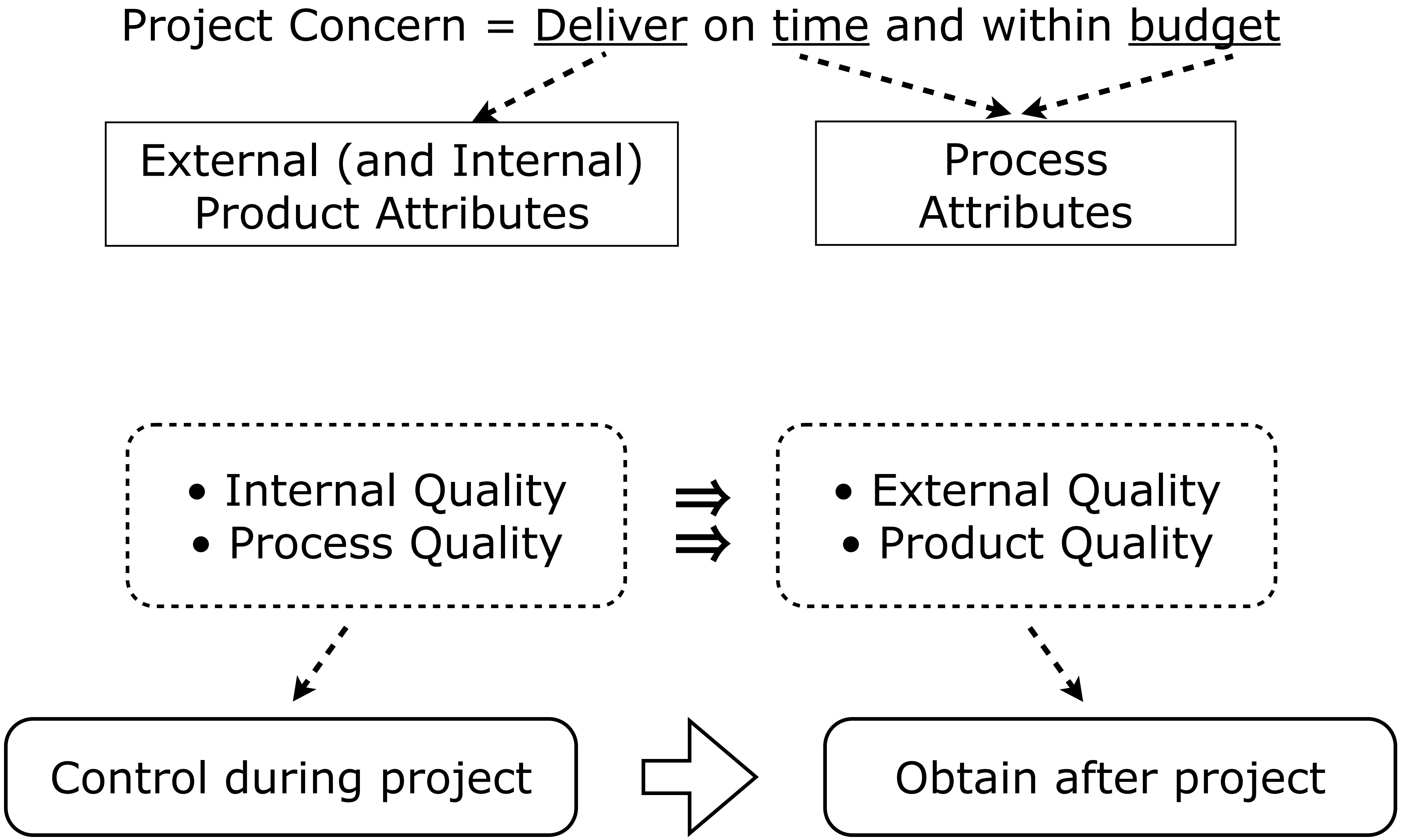
Can you name 3 related quality attributes and provide definitions for each of them?

- Productivity
  + Amount of product produced by a process for a given number of resources

- Timeliness
  + Ability to deliver the product on time

- Visibility (Transparency, Glasnost)
  + Current process steps and project status is accessible

Quality Control Assumption



Otherwise, quality is mere coincidence!

Quality Plan

Project Concern = Deliver on time and within budget

Project Plan
schedule: plan time budget: plan money quality plan: plan quality

A quality plan should:

- set out *desired product qualities* and how these are assessed
  + define the *most significant* quality attributes
  (cf. Quality Attributes Overview)
- set out which *organizational standards* should be applied
  + typically by means of *check-lists* and *standards*
- define the quality *assessment* process
  + typically done via *quality reviews* after internal release

Types of Quality Reviews

Review type	Principal purpose
Formal Technical Reviews (a.k.a. design or program inspections)	Driven by *checklist* • detect detailed errors in any product • mismatches between requirements and product • check whether standards have been followed.
Progress reviews	Driven by *budgets, plans and schedules* • check whether project runs according to plan • requires precise milestones • both a process and a product review

- Reviews should be recorded and records maintained
  + Software or documents may be “signed off” at a review
  + Progress to the next development stage is thereby approved

Review Meetings and Minutes

- (See [Pres00])

Review meetings should:

- typically involve 3-5 people
- require a maximum of 2 hours advance *preparation*
  + reviewers use checklists to evaluate products
- last less than 2 hours

The review minutes should summarize:

- 1. What was reviewed
- 2. Who reviewed it?
- 3. What were the findings and conclusions?
- 4. Decision
  + *Accepted* without modification
  + *Provisionally accepted*, subject to corrections (no follow-up review)
  + *Rejected*, subject to corrections and follow-up review

Review Guidelines

- 1. Review the product, not the producer
  - > Quality is a team responsibility
- 2. Set an agenda and maintain it
- 3. Limit debate and rebuttal
  - > Consensus is not required
- 4. Identify problem areas, but don’t attempt to solve every problem noted
- 5. Take written notes
  - > Blackboard or electronic white-boards for group awareness
- 6. Limit the number of participants and insist upon advance preparation
- 7. Develop a checklist for each product that is likely to be reviewed
- 8. Allocate resources and time schedule for reviews
  - > Including time for the modifications after the review
- 9. Conduct meaningful training for all reviewers
- 10. Review your early reviews
  - > Customise the review process by learning from your early attempts

Sample Review Checklists (i)

Project Plan

- 1. Is software scope unambiguously defined and bounded?
- 2. Are resources adequate for scope?
- 3. Have risks in all important categories been defined?
- 4. Are tasks properly defined and sequenced?
- 5. Is the basis for cost estimation reasonable?
- 6. Have historical productivity and quality data been used?
- 7. Is the schedule consistent?
- ...

Requirements Specification

- 1. Is information domain analysis complete, consistent and accurate?
- 2. Does the data model properly reflect data objects, attributes and relationships?
- 3. Are all requirements traceable to system level?
- 4. Has prototyping been conducted for the user/customer?
- 5. Are requirements consistent with schedule, resources and budget?
- ...

Sample Review Checklists (ii)

Design

- 1. Has modularity been achieved?
- 2. Are interfaces defined for modules and external system elements?
- 3. Are the data structures consistent with the information domain?
- 4. Are the data structures consistent with the requirements?
- 5. Has maintainability been considered?
- …

Code

- 1. Does the code reflect the design documentation?
- 2. Has proper use of language conventions been made?
- 3. Have coding standards been observed?
- 4. Are there incorrect or ambiguous comments?
- …

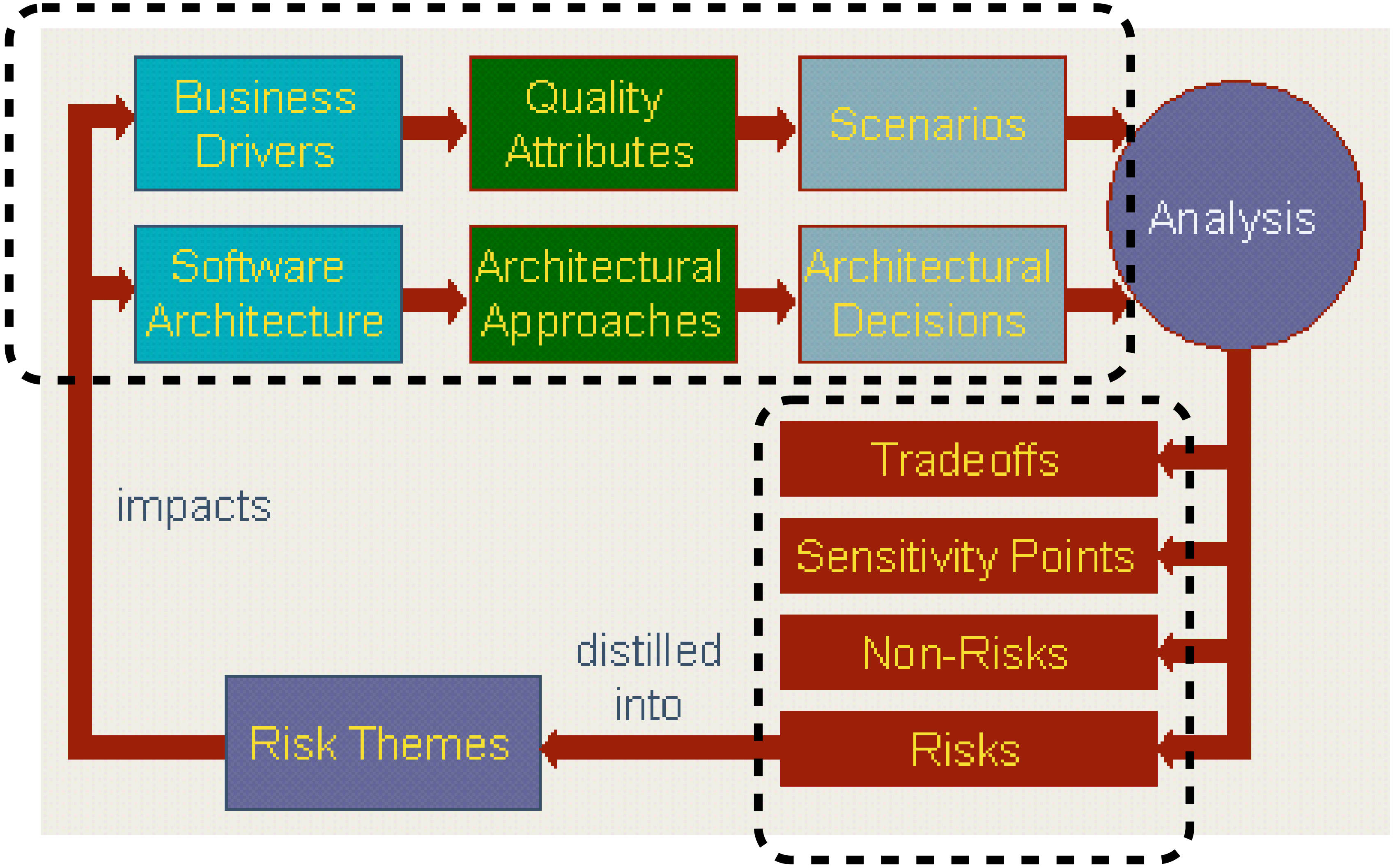
Testing

- 1. Have test resources and tools been identified and acquired?
- 2. Have both white and black box tests been specified?
- 3. Have all the independent logic paths been tested?
- 4. Have test cases been identified and listed with expected results?
- 5. Are timing and performance to be tested?
- ...

Sample Review Process: ATAM

ATAM = Architecture Tradeoff Analysis Method (see chapter “Software Architecture”)

Review these …



… to arrive at these!

Product and Process Standards

Product standards define characteristics that all components should exhibit.
Process standards define how the software process should be conducted.

Product standards	Process standards
Design review form	Design review conduct
Document naming standards (++)	Configuration management (++)
Procedure header format	Version release process
Coding conventions standard (++)	Project plan approval process
Project plan format	Change control process (++)
Change request form (+)	Test recording process (+)

Problems

- Not always seen as relevant and up-to-date by software engineers
- May involve too much bureaucratic form filling
- May require tedious manual work if unsupported by software tools

Sample Java Code Conventions

https://www.oracle.com/technetwork/java/codeconventions-150003.pdf

4.2 Wrapping Lines

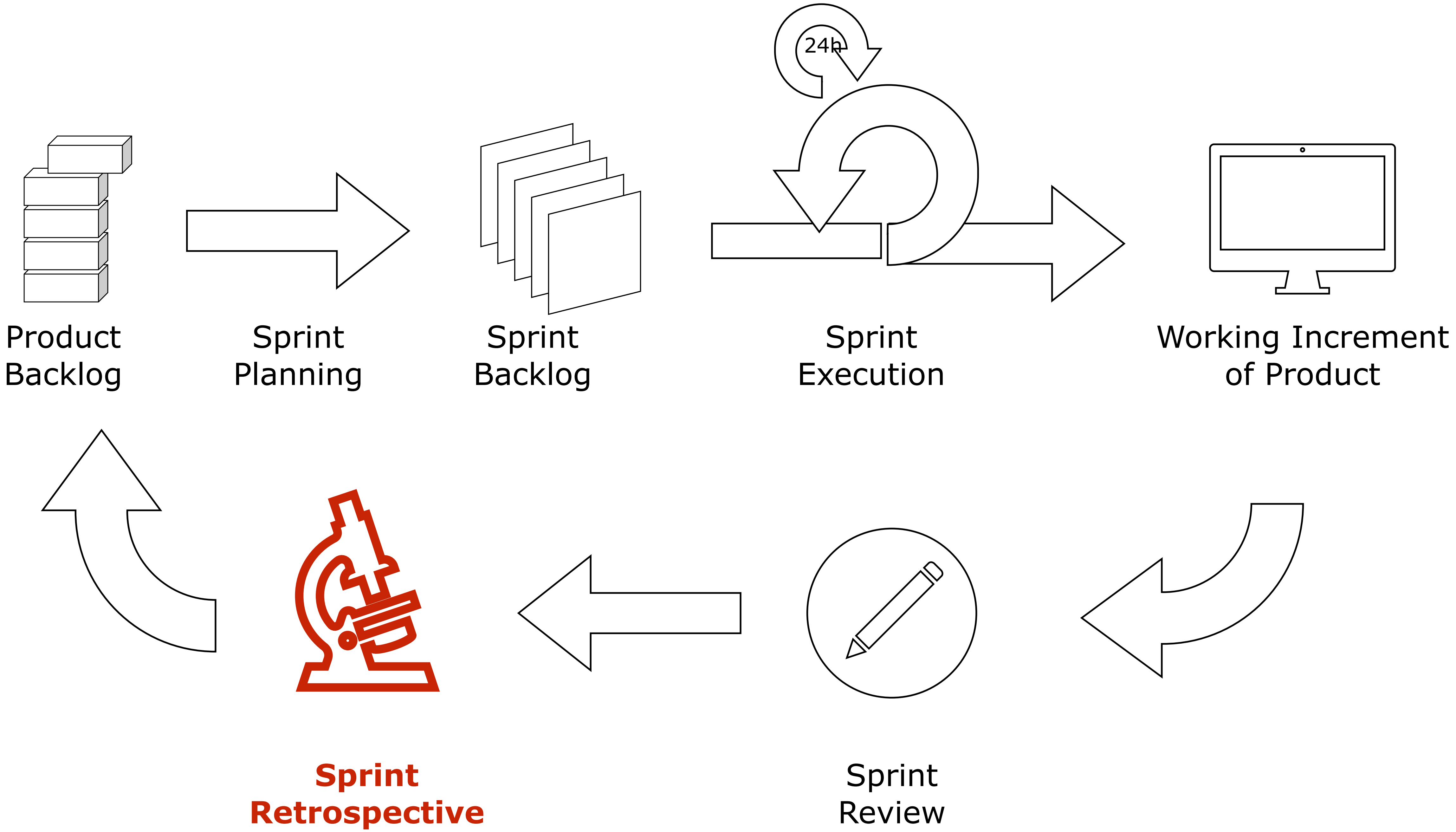
- When an expression will not fit on a single line, break it according to these general principles:
  + Break after a comma.
  + Break before an operator.
  + Prefer higher-level breaks to lower-level breaks.
  + Align the new line with the beginning of the expression at the same level on the previous line.
  + If the above rules lead to confusing code or to code that's squished up against the right margin, just indent 8 spaces instead.

…

10.3 Constants

- Numerical constants (literals) should not be coded directly, except for -1, 0, and 1, which can appear in a for loop as counter values.

Quality Culture: Continuous Improvement

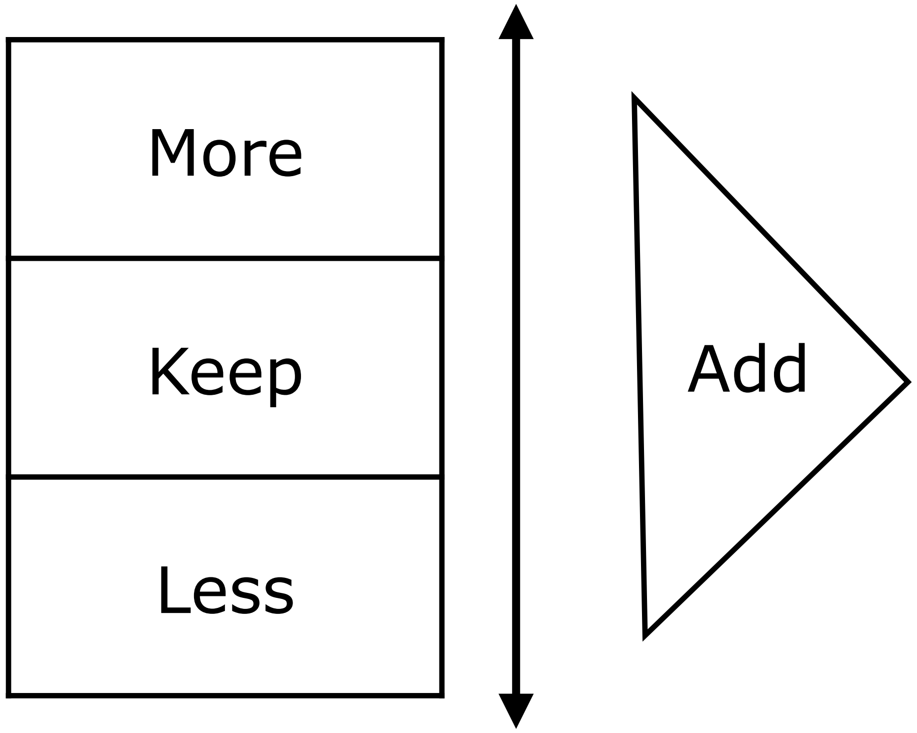


- What worked well this sprint that we want to continue doing?
- What didn’t work well this sprint that we should stop doing?
- What should we start doing or improve?

FUN RETROSPECTIVES *Activities and ideas for making agile retrospectives more engaging*

CAPSTONE PROJECT

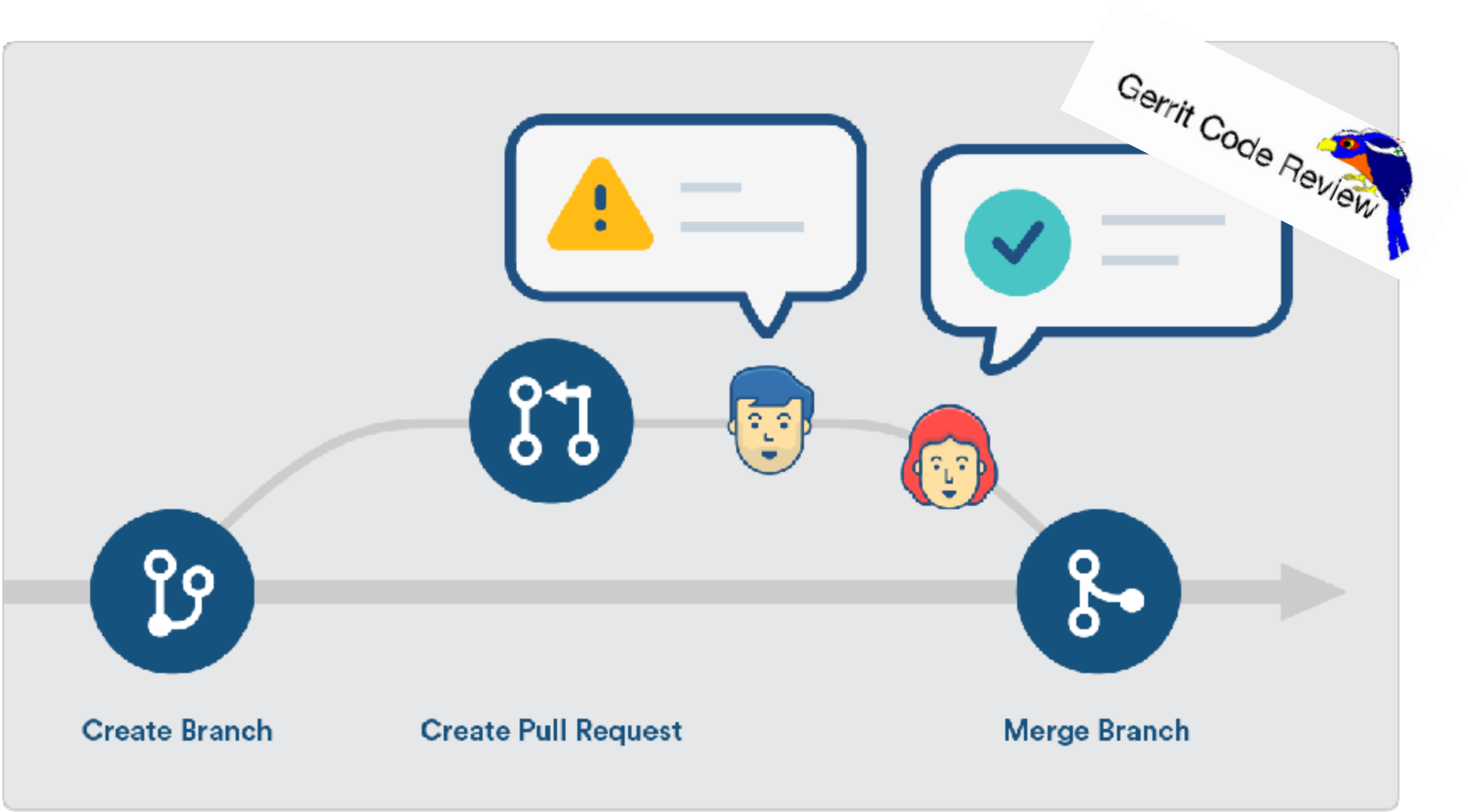
KALM (Keep - Add - Less - More)



DAKI (Drop - Add - Keep - Improve)

Drop	Add
Keep	Improve

Code Review on Pull Requests



© Amber Frauenholtz - Bitbucket — 5 elements of a perfect pull request

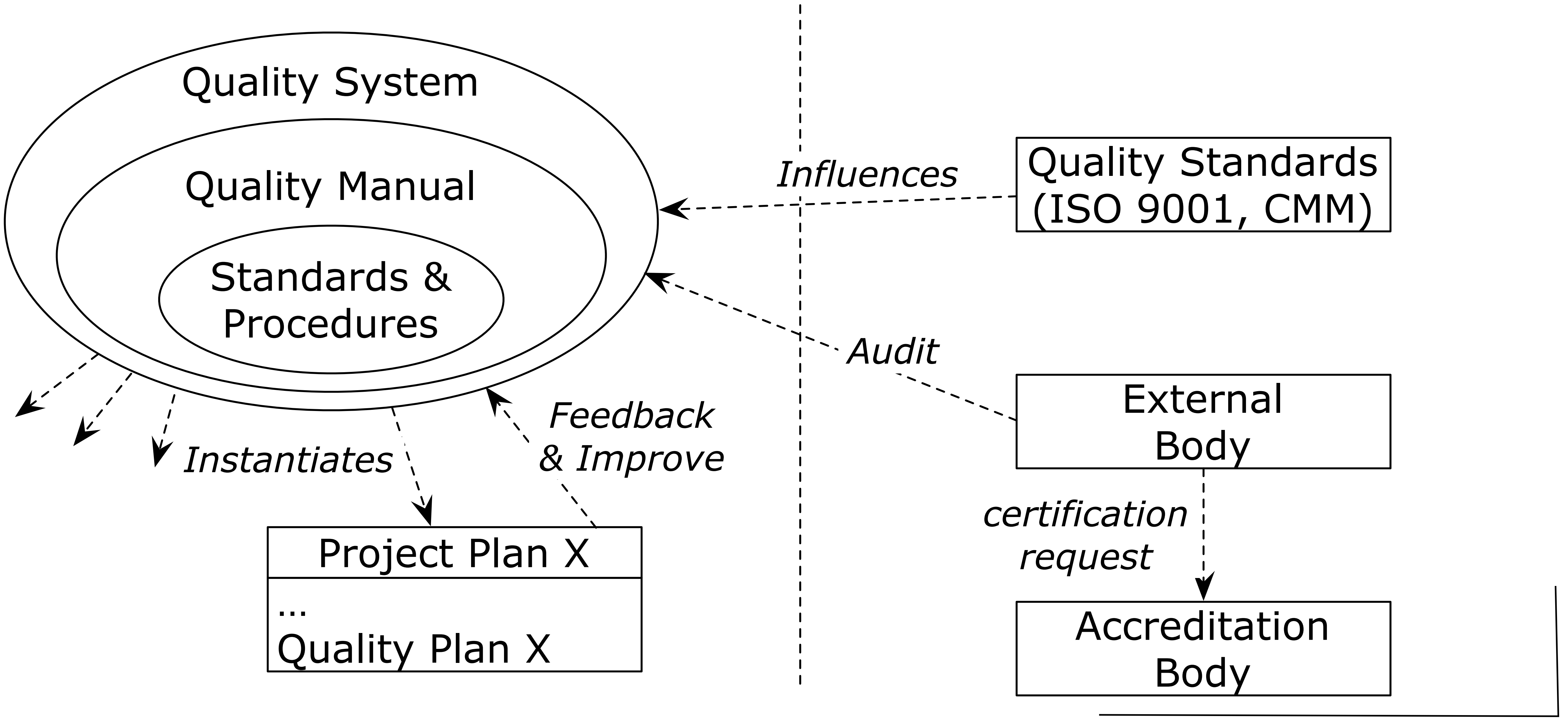
Quality System & Certification

When starting a project, the project will include a Quality Plan

- Ideally, such plan is an instance of the organization's *Quality System*

Certain customers require an externally reviewed quality system

- An organization may request to *certify* its quality system



ISO 9000

ISO 9000

- is an international set of standards for quality management applicable to a range of organisations from manufacturing to service industries.

ISO 9001

- is a generic model of the quality process, applicable to organisations whose business processes range all the way from design and development, to production, installation and servicing;

- ISO 9001 must be instantiated for each organisation
- ISO 9000-3 interprets ISO 9001 for the software developer

ISO = International Organisation for Standardization

- ISO main site: ISO [http://www.iso.org/]
- ISO 9000 family
  + [https://www.iso.org/iso-9001-quality-management.html]

ISO 9001

- Describes quality standards and procedures for developing products of any kind:

Management responsibility	Quality system
Control of non-conforming products	Design control
Handling, storage, packaging & delivery	Purchasing
Purchaser-supplied products	Product identification & traceability
Process control	Inspection and testing
Inspection and test equipment	Inspection and test status
Contract review	Corrective action
Document control	Quality records
Internal quality audits	Training
Servicing	Statistical techniques

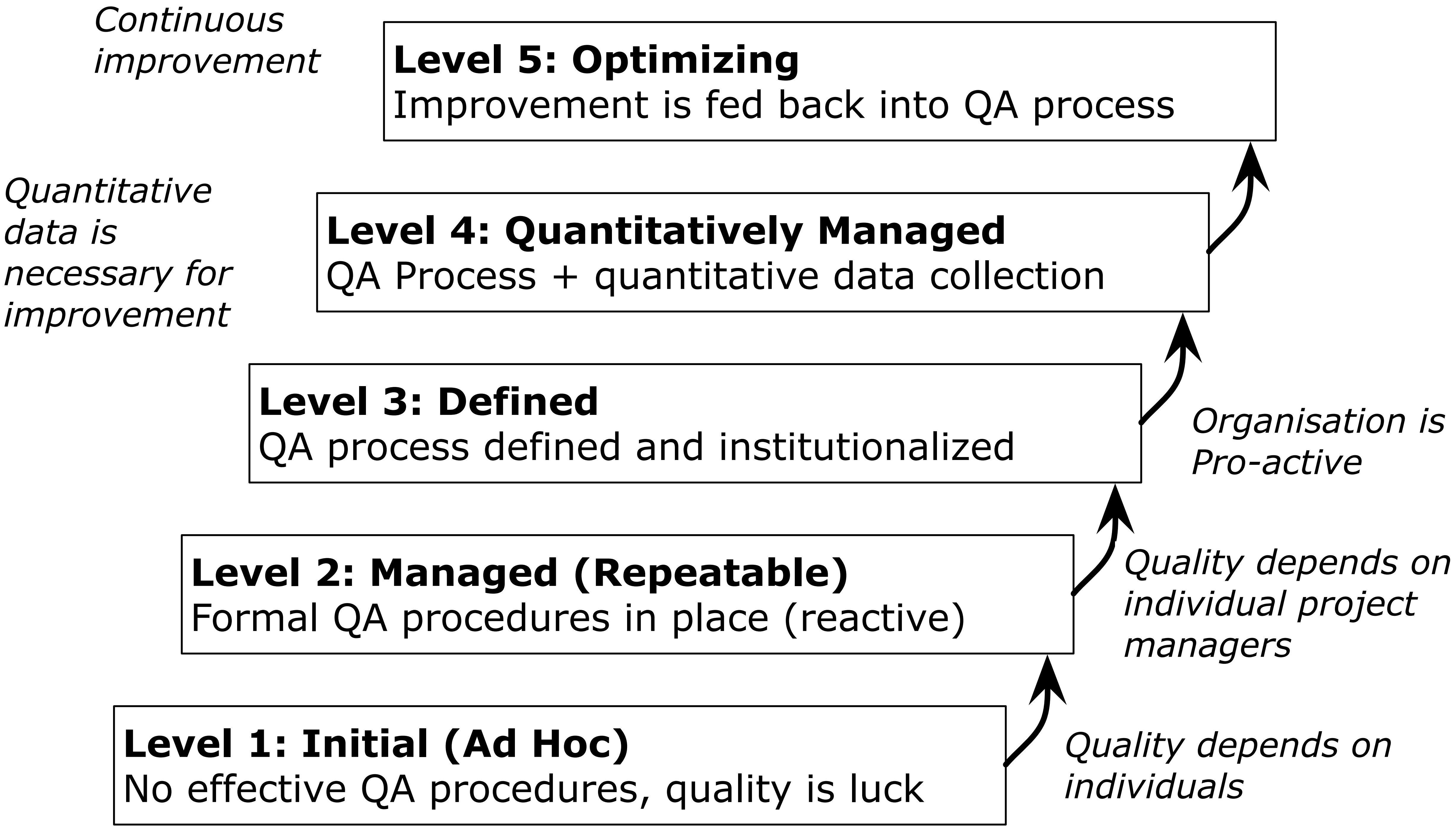
Capability Maturity Model

- Process maturity model from SEI (Software Engineering institute)
  + Initiated in 1991
  + Version 1.1 completed in January 1993
- Tool to evaluate the ability of government contractors to perform a contracted software project
  + assess how well contractors manage software processes
  + says little on individual projects
  + not necessarily applicable to commercial-off-the-shelf (COTS)

- CMM is now superseded by CMMI
  + **CMMI** = Capability Maturity Model Integration
  + Integrate software quality standard with standards other disciplines
  + Version 1.1 in 2002 – Version 1.2 in August 2006

- Essentially CMMI consists of
  + 5 maturity levels
    - Separate standards for development / services / acquisition
  + Core Process Area
    - identifies a cluster of related activities that, when performed collectively, achieve a set of goals considered important

CMMI: Overview



Continuous improvement

Level 5: Optimizing
Improvement is fed back into QA process

Quantitative data is necessary for improvement

Level 4: Quantitatively Managed
QA Process + quantitative data collection

Level 3: Defined
QA process defined and institutionalized

Organisation is Pro-active

Level 2: Managed (Repeatable)
Formal QA procedures in place (reactive)

Quality depends on individual project managers

Level 1: Initial (Ad Hoc)
No effective QA procedures, quality is luck

Quality depends on individuals

Core Process Areas

Abbr.	Name	Area	Level
CM	Configuration Management	Support	2
MA	Measurement and Analysis	Support	2
PMC	Project Monitoring and Control	Project Management	2
PP	Project Planning	Project Management	2
PPQA	Process and Product Quality Assurance	Support	2
REQM	Requirements Management	Project Management	2
DAR	Decision Analysis and Resolution	Support	3
IPM	Integrated Project Management	Project Management	3
OPD	Organizational Process Definition	Process Management	3
OPF	Organizational Process Focus	Process Management	3
OT	Organizational Training	Process Management	3
RSKM	Risk Management	Project Management	3
OPP	Organizational Process Performance	Process Management	4
QPM	Quantitative Project Management	Project Management	4
CAR	Causal Analysis and Resolution	Support	5
OPM	Organizational Performance Management	Process Management	5

Core Process Areas

Abbr.	Name	Area	Level
CM	Configuration Management	Support	2
MA	Measurement and Analysis	Support	2
PMC	Project Monitoring and Control	Project Management	2
PP	Project Planning	Project Management	2
PPQA	Process and Product Quality Assurance	Support	2
REQM	Requirements Management	Project Management	2
DAR	Decision Analysis and Resolution	Support	3
IPM	Integrated Project Management	Project Management	3
OPD	Organizational Process Definition	Process Management	3
OPF	Organizational Process Focus	Process Management	3
OT	Organizational Training	Process Management	3
RSKM	Risk Management	Project Management	3
OPP	Organizational Process Performance	Process Management	4
QPM	Quantitative Project Management	Project Management	4
CAR	Causal Analysis and Resolution	Support	5
OPM	Organizational Performance Management	Process Management	5

testing??

use-cases??

Conclusion: Reviews

Reviews and Inspections

- Low on ceremony, high on external product quality
- Side-effect: team ownership

- Very effective
  - + Empirical evidence shows that reviews find more errors than tests (+ reviews usually indicate a solution)

- Very cost effective
  - + Empirical evidence shows that reviews find errors more cheaply than tests

- However: tests find other errors than reviews
  - > Reviews must supplement testing

Conclusion: Quality Standards

Quality Standards (ISO9000 and CMMI)

- No guarantee for external *product* quality
  + There is NO empirical evidence that ISO, CMMI actually improve quality

- Adequate for process *quality*
  + ... on time and within budget

- Does not scale down
  + developing a quality system is an overhead
  + difficult for small enterprises (where most software development is done)

- Eliminate *coincidence*
  + ... eliminates creativity (to some degree)
  + often obstructed by people doing the work
- Tendency towards high ceremony
  + difficult for rapidly changing contexts (e-commerce)

Is a means, not a goal
Illustrates that quality is an important issue
Certification is a driving force

Summary (i)

You should know the answers to these questions

- Why is software quality more important than it was a decade ago?
- Can a correctly functioning piece of software still have poor quality? Why?
- If quality control can't guarantee results, why do we bother?
- What's the difference between an external and an internal quality attribute? And between a product and a process attribute?
- What's the distinction between correctness, reliability and robustness?
- How can you express the "user friendliness" of a system?
- Can you name three distinct refinements of "maintainability"? What do each of these names mean?
- What is meant with "short time to market"? Can you name 3 related quality attributes and provide definitions for each of them?
- Name four things which should be recorded in the review minutes.
- Explain briefly the three items that should be included in a quality plan.
- What's the relationship between ISO9001, CMMI standards and an organization's quality system? How do you get certified?
- Can you name and define the 5 levels of CMMI?
- Where would "use-cases" as defined in chapter 3 fit in the table of core process areas (p. 32)? Motivate your answer shortly.

Summary (ii)

You should be able to complete the following tasks

- Given a piece of code and a coding standard, review the code to verify whether the standard has been adhered to.

Can you answer the following questions?

- Given the Quality Attributes Overview table, argue why the crosses and blanks occur at the given positions.
- Why do quality standards focus on process and internal attributes instead of the desired external product attributes?
- Why do you need a quality plan? Which topics should be covered in such a plan?
- How should you organize and run a review meeting?
- Why are coding standards important?
- What would you include in a documentation review checklist?
- How often should reviews by scheduled?
- Could you create a review check-list for ATAM?
- Would you trust software from an ISO 9000 certified company? And if it were CMMI?
- You are supposed to develop a quality system for your organization. What would you include?
- Where would "testing" fit in the table of core process areas (p. 32). Does it cover a single row or not? Argue why (not)?