QO, QI, QM –what are they?

Understanding terminology



# Quality model elements

**Quality Objective**

* Highest level concepts
* Describe one characteristic of a process, system or product
* Different levels of a product or process may have different quality objectives
* Different types of objectives
  + PRODUCT: Success factors (currently reasonably good)
  + PRODUCT: Challenges (currently unsatisfactory)
  + PROCESS: Continuous (require continuous actions)
  + PROCESS: Step-wise (require actions by certain time)

**Quality Indicator**

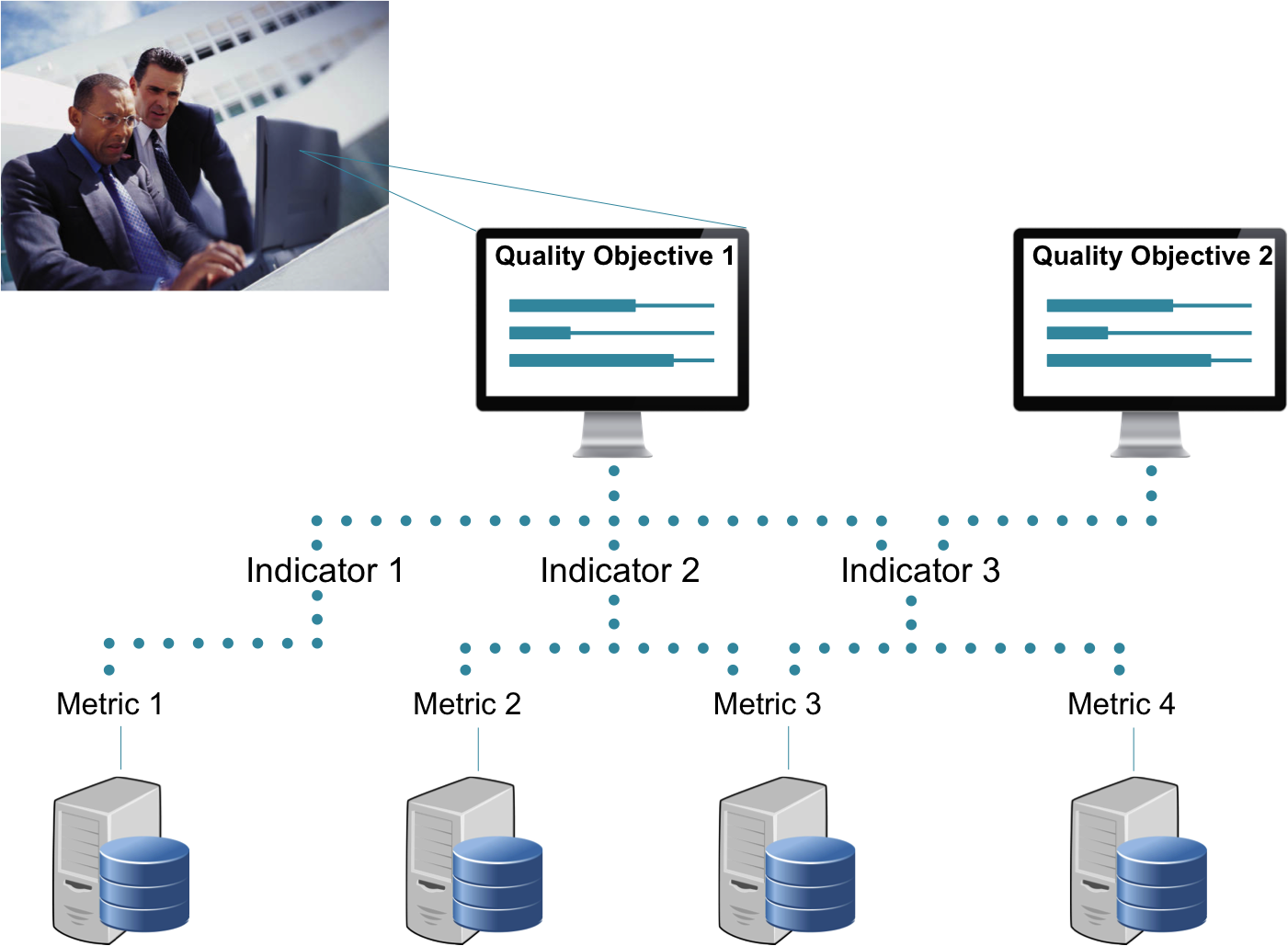
* A way for evaluating one aspect of a quality objective
* “What indicates that the quality of an objective is high or low?”

**Quality Metric**

* A way for measuring the state of a quality indicator
* Includes concrete factors that can be measured from e.g. company databases

**Quality Model**

* Quality model means the overall picture of QO, QI and QM



“A quality model includes one or several quality objectives. Quality objectives are described by one or several quality indicators. Quality indicators are measured by quality metrics.”

# Examples of quality objectives

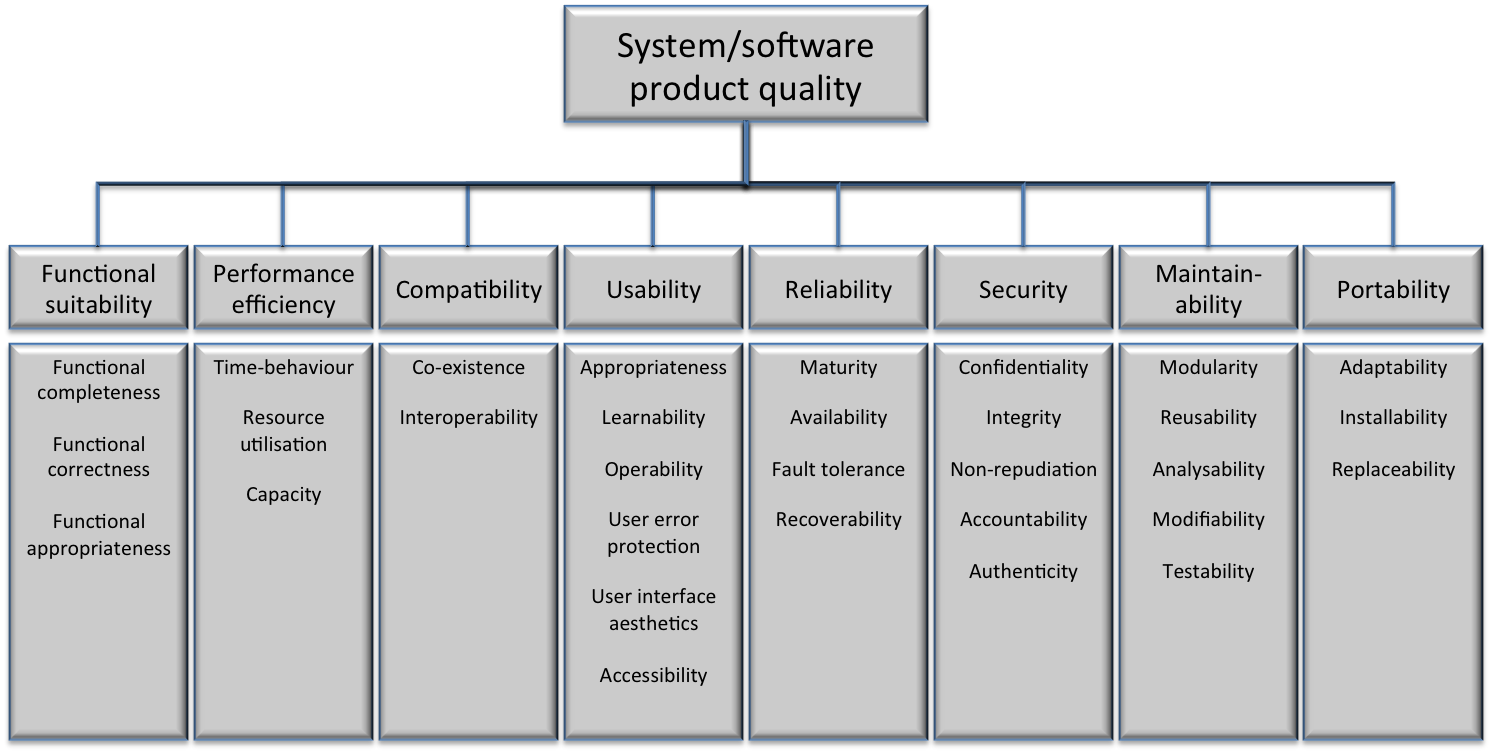


Figure 1. ISO/IEC 25010 quality model (ISO/IEC 2010)

***Functional suitability:*** degree to which a product or system provides functions that meet stated and implied needs when used under specified conditions

***Performance efficiency:*** performance relative to the amount of resources used under stated conditions

***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

***Usability:*** degree to which a product or system can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use

***Reliability:*** degree to which a system, product or component performs specified functions under specified conditions for a specified period of time

***Security:*** degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization

***Maintainability:*** degree of effectiveness and efficiency with which a product or system can be modified by the intended maintainers

***Portability:*** degree of effectiveness and efficiency with which a system, product or component can be transferred from one hardware, software or other operational or usage environment to another

# Determining indicators and metrics

**What should be taken into account while defining indicators and metrics?**

**How to elaborate a quality objective?**

1. Choose a quality objective
2. Follow the steps presented on the previous page for:
   * Defining the indicators for the objective
     + What indicates the status of them?
   * Defining the metrics for the indicator
     + Think about concrete measurements
     + Think about the ways that indicator could be measured
     + What data is needed for making the indicator visible?
     + What is the concrete artefact that is used
   * E.g. a source code module or feature or project
   * What / how wide is the focus?
3. Fill in the quality objective template

# Quality Objective Template

**General form for elaborating quality objectives**

|  |  |
| --- | --- |
| Name: | Short name for the objective. |
| Description: | Description that documents the objective on higher level than a single measure. |
| Rationale: | Motivates why the objective is important. Gains and risks; E.g., costs, business benefits, customer viewpoint. |
| Related factors: | Tentative ideas of what contributes, or prevents achieving this objective |
| One or more quality indicators | Indicators and associated metrics to indicate achieving this objective:  For each quality indicator:  Description to understand exactly what is the indicator and a measure for it, including the formula if aggregating several metrics.  Calculation formula if combines multiple metrics or is otherwise needed.  Current level of the indicator  Target levels of the indicator   * Lower breakpoint (useless, if below this level) * Higher breakpoint (competitive advantage, if above) * Target level (current realistic target)   Measuring interval, continuous or snapshot, how often  Easiness of collecting (We have it / Requires some work / Impossible)  Details of all the metrics that the indicator consists of. |

**Example of a filled form for elaborating objectives**

|  |  |
| --- | --- |
| Name: | Easy updateability |
| Description: | Updating the software should be quick and easy, ideally possible without deep technical or product knowledge |
| Rationale: | Direct cost savings related to updates. Reduced risk of errors during updates. |
| Related factors: | Robustness of the software, configurability, quality of installer software |
| Quality indicator 1 | Updating effort  Amount of average working hours used in making an update for a single customer installation. Measured as hours from the hour reporting system.  Current level: 3 h  Target levels:   * Lower breakpoint: 8h * Higher breakpoint: 1h * Target level: 15 min |
| Easiness of collecting: | Manual data exist; adapter to hour reporting system requires around 1 person week of work. |

# Evaluating the designed metrics

Here is a checklist to help evaluating the validity of your metrics (Kaner & Bond):

1. ***What is the purpose of this measure?***

Examples: evaluating project status or staff performance, a self-assessment or to informing external actors

1. ***What is the scope of this measure?***

Are you collecting data one single time, a single project or is it supposed to be measured continuously in all projects from now on? It may be that metrics that are good on a smaller scale will fail to give the right result, because other factors also affect the outcome.

1. ***What is the relationship of the attribute to the metric value?***

This is the important construct validity problem: How do we know that the metric measures that attribute in a good way?

|  |  |
| --- | --- |
| **Quality Attribute** | **Quality metric** |
| *What attribute are we trying to measure?* | *What is the metric (the function that assigns a value to the attribute)?* |
| *What is the natural scale of the attribute we are trying to measure?* | *What is the natural scale for this metric?* |
| *What is the natural variability of the attribute?* | *What is the natural variability of readings from this instrument?* |

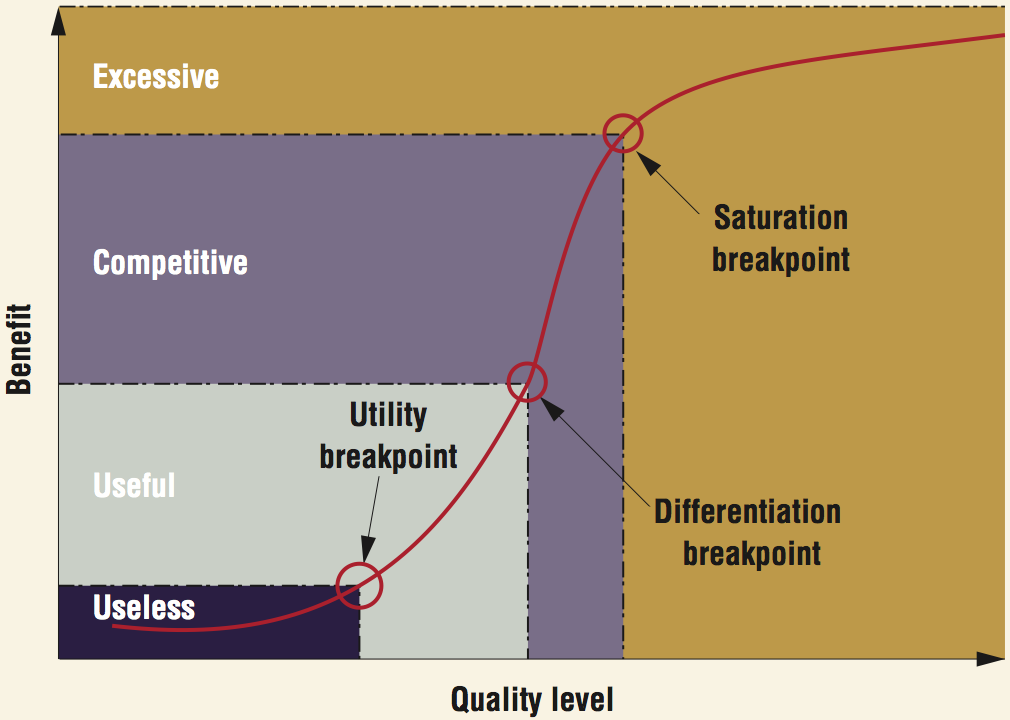
If you have clear idea of what you actually want to measure, it is easier to establish a good relationship between what you measure and the idea. Some attributes are easy to decide how to measure, such as length. It is more difficult to decide on the correct scales for attributes such as skills, code quality or testing thoroughness. It is natural that there is some variation in an attribute. What is the natural variation of the amount of tests a person can go through each day? And even more important - what causes these natural variations to occur?

The scale of the attribute you want to measure and the metric you decide to use can differ. For example, thoroughness of testing cannot be measured on a natural scale, but having an expert evaluate and rank different test artefacts according to quality follows an ordinal scale. This means the normal measurement error, and includes errors that we may do something about (systematic error) and those beyond our possible or reasonable control (random error).

1. ***What measuring instruments do we use to perform the measurement?***

Examples of metrics are counting (bugs, number of items in the backlog, lines of code), matching (“This requirement is equally complex to that one”), comparing (code quality) and timing (time to finish a requirement). This can be done manually or automatically.

1. ***What are the natural and foreseeable side effects of using this instrument?*** Introducing metrics to measure an attribute can be a positive thing. On the other hand it can also give way to a set of unforeseen and negative results, and the less tightly linked a measure is to the underlying attribute, the more side effects can occur. For example, if quality of a coder is measured by number of finished requirements per time, this could lead to developers to cherry pick the simple tasks.



Quality metrics can also be evaluated by using the QUPER-model, from which the first phase is presented above. It guides to think carefully about the values of the metric: if the values are too unfocused, the metric might be useless. On the contrary, if the values are too specific or strict, the metric might require too much effort for too little advantage.