LOG8371 : Ingénierie de la qualité en logiciel

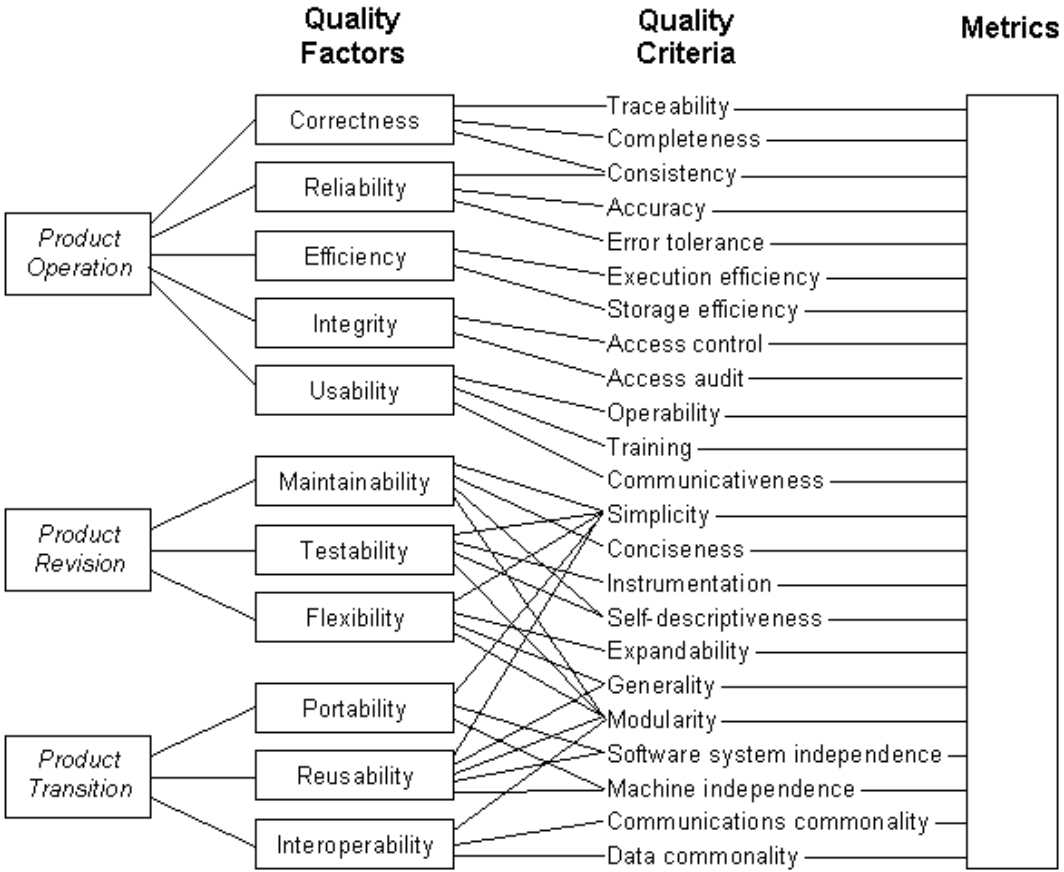
Les modèles CMMI, ISO 25000 et Quamoco

Hiver 2017

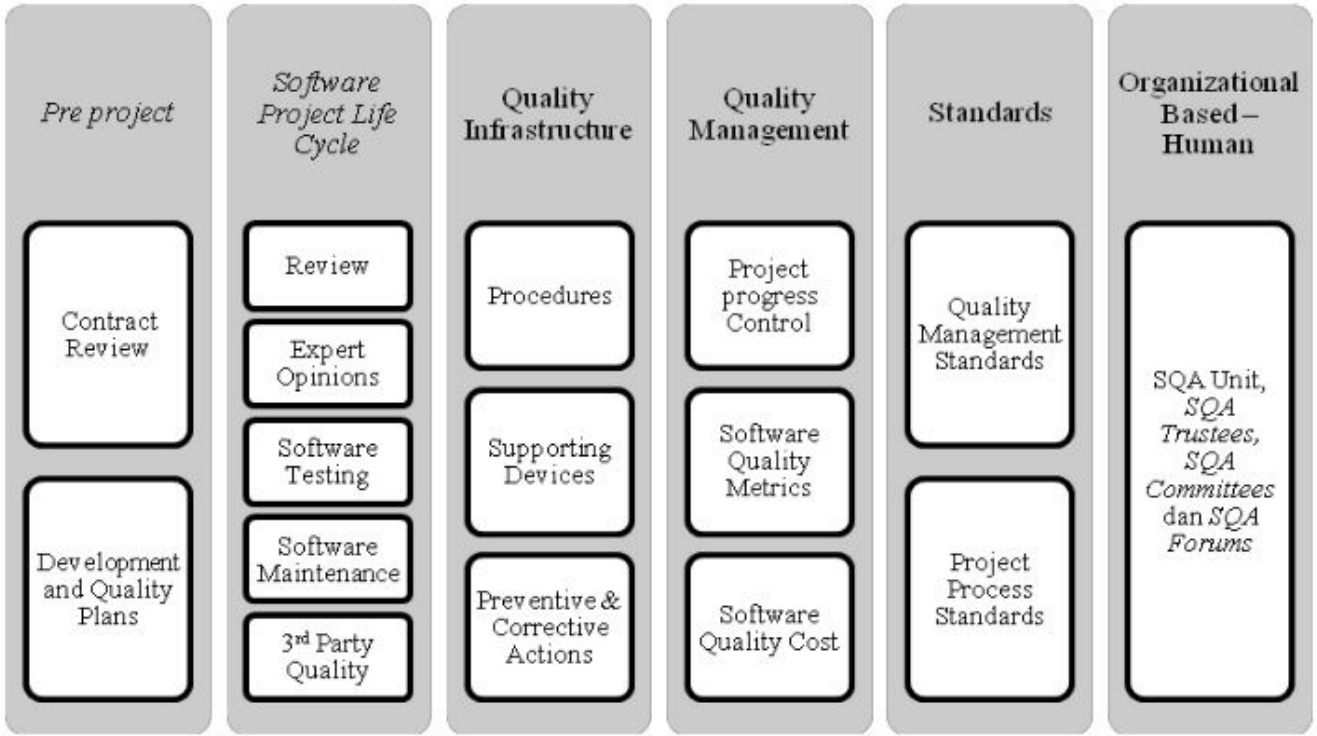
POLYTECHNIQUE MONTRÉAL
LE GÉNIE EN PREMIÈRE CLASSE

Fabio Petrillo
Chargé de Cours

Modèle de McCall's



DANIEL GALIN'S QUALITY SHRINE



EVALUATION OF METHODOLOGIES AGAINST QUALITY FACTORS.

Quality Parameters	Agile Methodologies and quality activities				
	XP (extreme programming)	**Srcum**	**Crystal methodologies**	**Feature driven development**	**RUP**
Correctness	User stories	Review meetings	Interviews and formal meetings, printing whiteboards	Feature lists prepared with customers	Inception phase. Interviews with stakeholders
Robustness	Generic OO design	OO Design	Tuning methods	OO design	OO design
Extendibility	Simple design	Prioritization	Versioning systems	Simple design + versioning system	UML design
Reusability	OO design	Existing design	Configuration management tools	OO design	UML design, COTS development
Compatibility	Inherit in OO deign	OO design	Version control	Version control	Inherit in OO design
Efficiency	Pair programming	Team meeting	White boards	Class code ownership	UML modeling
Portability	OO design practices	OO designs	Compilation tools	Configuration management support	Component Based architecture
Timeliness	Incremental and iterative development	Iterative	Incremental 4-6 weeks	Iterative development	Iterative development
Integrity	OO design	OO design	Common object model	Class code ownership	UML modeling
Verification and validation	Unit testing	Testing and reviews	Regression testing , Interface testing	Inspection	Acceptance testing
Ease of use	Simple design	Simple design	GUI Designs, User manuals	Feature list	UML modeling
Maintainability	Simple design	Compatible with existing system	Versioning system	OO design and versioning control	UML compliances
Performance	Productioniz-ing phase	Sprint phase	Common object Model	Class code ownership	OO design
Cost effectiveness	Iterative development	Iterative development	Incremental delivery on regular basis	Reusable design features in iterative manner	Iterative development with OO design
Correctness	User stories	Review meetings	Interviews and formal meetings, printing whiteboards	Feature lists prepared with customers	Inception phase. Interviews with stakeholders
Robustness	Generic OO design	OO Design	Tuning methods	OO design	OO design
Extendibility	Simple design	Prioritization	Versioning systems	Simple design + versioning system	UML design
Reusability	OO design	Existing design	Configuration management tools	OO design	UML design, COTS development
Compatibility	Inherit in OO deign	OO design	Version control	Version control	Inherit in OO design
Efficiency	Pair programming	Team meeting	White boards	Class code ownership	UML modeling
Portability	OO design practices	OO designs	Compilation tools	Configuration management support	Component Based architecture
Timeliness	Incremental and iterative development	Iterative	Incremental 4-6 weeks	Iterative development	Iterative development
Integrity	OO design	OO design	Common object model	Class code ownership	UML modeling
Verification and validation	Unit testing	Testing and reviews	Regression testing , Interface testing	Inspection	Acceptance testing
Ease of use	Simple design	Simple design	GUI Designs, User manuals	Feature list	UML modeling
Maintainability	Simple design	Compatible with existing system	Versioning system	OO design and versioning control	UML compliances
Performance	Productioniz-ing phase	Sprint phase	Common object Model	Class code ownership	OO design
Cost effectiveness	Iterative development	Iterative development	Incremental delivery on regular basis	Reusable design features in iterative manner	Iterative development with OO design

Mehreen Sirshar and Fahim Arif, Evaluation of Quality Assurance Factors in Agile Methodologies. International Journal of Advanced Computer Science, Vol. 2, No. 2, Pp. 73-78, Feb. 2012.

Les avantages à utiliser les standards de SQA

- Appliquer des méthodologies de développement et de maintenance au niveau professionnel
- Meilleure coordination et compréhension mutuelle entre les équipes de développement
- Meilleure coopération entre développeurs et participants extérieurs
- Entraînement, support, certifications et audit
- Effort international (et réputation)
  - ISO (International Organization for Standardization)
  - IEEE (Institute of Electrical and Electronics Engineers) Computer Society
  - ANSI (American National Standards Institute)

Classification des standards SQA

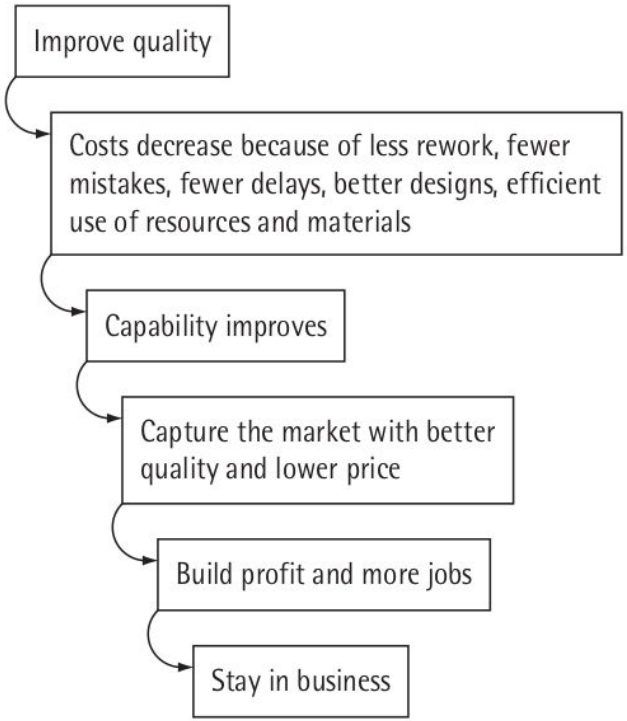
- Standards de qualité de **management** (“Quoi”)
  - Se concentrent sur l’organisation du système SQA, l’infrastructure et les exigences
  - **Système** d’assurance qualité logicielle
  - ISO 9000-3 et “Capability Maturity Model” (**CMM**)
- Standards de gestion de projet (“Comment”)
  - Se concentrent sur les méthodologies de gestion du développement logiciel et de la maintenance
  - Étapes, documentation, conception, revues, tests,...
  - ISO 25010

Capability Maturity Model (Intégration) – CMM(I)

Capability Maturity Model (Intégration) – CMM(I)

- Software Engineering Institute (SEI) de l'université de Carnegie Mellon
- 1986 - Premiers pas / 1991 - Version 1.0 / 1993 - Version 1.1 / 1995 - Livre
- Un cadre de la maturité des processus informatiques aidant les entreprises à améliorer leurs processus de développement
- Requête du gouvernement fédéral américain (Département de la défense)
- CMM a eu un impact énorme sur la communauté des développeurs
- Basé sur les principes de "Qualité Totale" (total quality management ou TQM)
- La réaction en chaîne de Deming

La réaction en chaîne de Deming



CMMI® for Development, Version 1.3

CMMI-DEV, V1.3

CMMI Product Team

Improving processes for developing better products and services

November 2010

TECHNICAL REPORT

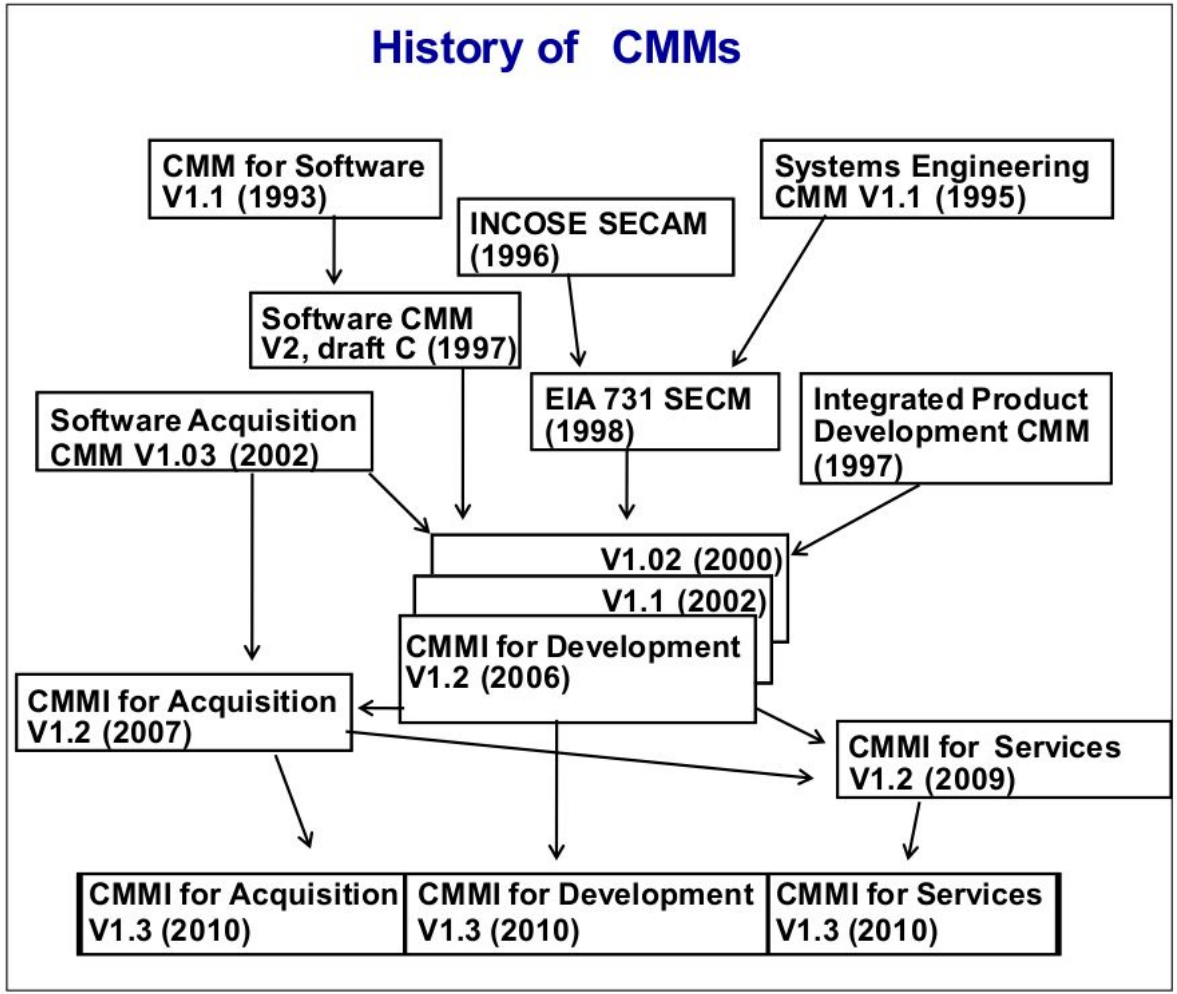
CMU/SEI-2010-TR-033
ESC-TR-2010-033

~ 500 pages

Software Engineering Process Management Program

http://www.sei.cmu.edu

History of CMMs



Capability Maturity Model (Intégration) – CMM(I)

- **Idée principale**: “L’utilisation de **méthodes de management plus élaborées** basées sur une approche **quantitative** améliore la capacité de l’entreprise à controler la qualité et augmente la **productivité** du processus de développement.”
- Le modèle CMMI est un **cadre**
  - Il ne spécifie pas de documentation quelconque
- Il permet l’utilisation de n’importe quel modèle de cycle de vie
- Indépendant de la technologie utilisée

Capability Maturity Model (Intégration) – CMM(I)

- Modèle sur **5 niveaux** de capacité de maturité
  - Guide dans le choix des améliorations à apporter
  - Priorisation
  - Amélioration progressive et graduelle
- 25 Domaines de processus


MATURITY LEVEL 5
Optimizing
Stable and flexible. Organization is focused on continuous improvement and is built to pivot and respond to opportunity and change. The organization's stability provides a platform for agility and innovation.
5
MATURITY LEVEL 4
Quantitatively Managed
Measured and controlled. Organization is data-driven with quantitative performance improvement objectives that are predictable and align to meet the needs of internal and external stakeholders.
4
MATURITY LEVEL 3
Defined
Proactive, rather than reactive. Organization-wide standards provide guidance across projects, programs and portfolios.
3
MATURITY LEVEL 2
Managed
Managed on the project level. Projects are planned, performed, measured, and controlled.
2
MATURITY LEVEL 1
Initial
Unpredictable and reactive. Work gets completed but is often delayed and over budget.
1

Niveau 1 de maturité - Initial

- Le processus de développement se caractérise comme étant *ad hoc* (spécialisé)
- Peu de processus sont définis
- Environnement instable
- La réussite dépend des efforts individuels et d'investissements personnels exceptionnels
- Difficile de prédire le niveau de performance
- Apprentissage par l'expérience alors que tout est nouveau et unique
- Processus d'abandon dans les temps de crise
- Incapacité à reproduire des succès passés

Niveau 2 de maturité - Géré et reproductible

- Les processus sont planifiés et exécutés en accord avec la politique en place
- Le projet emploie des personnes compétentes qui ont les ressources adéquates pour produire ce qui est attendu
- Inclut des acteurs pertinents, qui sont surveillés, contrôlés, révisés et qui sont évalués sur leur adhésion aux descriptions du processus
- Les projets sont réalisés et gérés en accord avec la planification documentée
- La gestion est **réactive**

Niveau 2 de maturité - types de processus

- CM - Gestion de la configuration
- MA - Mesures et Analyses
- PPQA - Assurance qualité produit et processus
- REQM - Gestion des exigences
- SAM - Gestion des accords avec les fournisseurs
- SD - Prestation de service
- WMC - Surveillance et contrôle du travail
- WP - Planification du travail

Niveau 3 de maturité - types de processus

- Les processus sont bien définis et compris et sont décrits à l'aide de standards, procédures, outils et méthodes
- Ils sont cohérents dans **toute** l'**entreprise**
- Description plus rigoureuse
  - Définit clairement les buts, entrées et sorties et leurs critères, activités et rôles, mesures et étapes de vérification
- Les processus sont gérés de manière plus **proactive**

Niveau 3 de maturité - types de processus

- CAM - Gestion des capacités et des disponibilités
- DAR - Analyse des décisions et résolution
- IRP - Résolution et prévention d’incidents
- IWM - Gestion du travail intégré
- OPD - Définition des processus d’organisation
- OPF - Focus sur les processus organisationnel
- OT - Formation à l’organisation
- RSKM - Gestion du risque
- SCON - Continuité de service
- SSD - Développement du système de services
- SST - Transition du système de services
- STSM - Gestion des services stratégiques

Niveau 4 de maturité - Gestion quantitative

- L’entreprise établit pour les projets des objectifs **quantitatifs** de qualité et de performance des processus
- Les résultats des processus et la qualité sont perçus avec une vision statistique
- **Prévisibilité** des performances des processus
- Analyse statistique de données issues d’une analyse fine des processus

Niveau 4 de maturité - Types de processus

- OPP - Performance des processus organisationnels
- QWM - Gestion du travail quantitative

Niveau 5 de maturité - Optimisation

- L’entreprise **améliore** continuellement ses processus, en se basant sur une compréhension **quantitative** de ses **objectifs** opérationnels et de ses besoins en terme de **performance**
- Révisions continuelles afin de refléter l’**évolution** des objectifs et des performances organisationnelles
- Les effets des améliorations déployées sont mesurés et analysés quantitativement et statistiquement
- La performance en terme d’organisation est atteinte grâce à l’utilisation de données rassemblées sur plusieurs projets

Niveau 5 de maturité Maturity - types de processus

- CAR - Analyse des causes et résolution
- OPM - Gestion de la performance organisationnelle

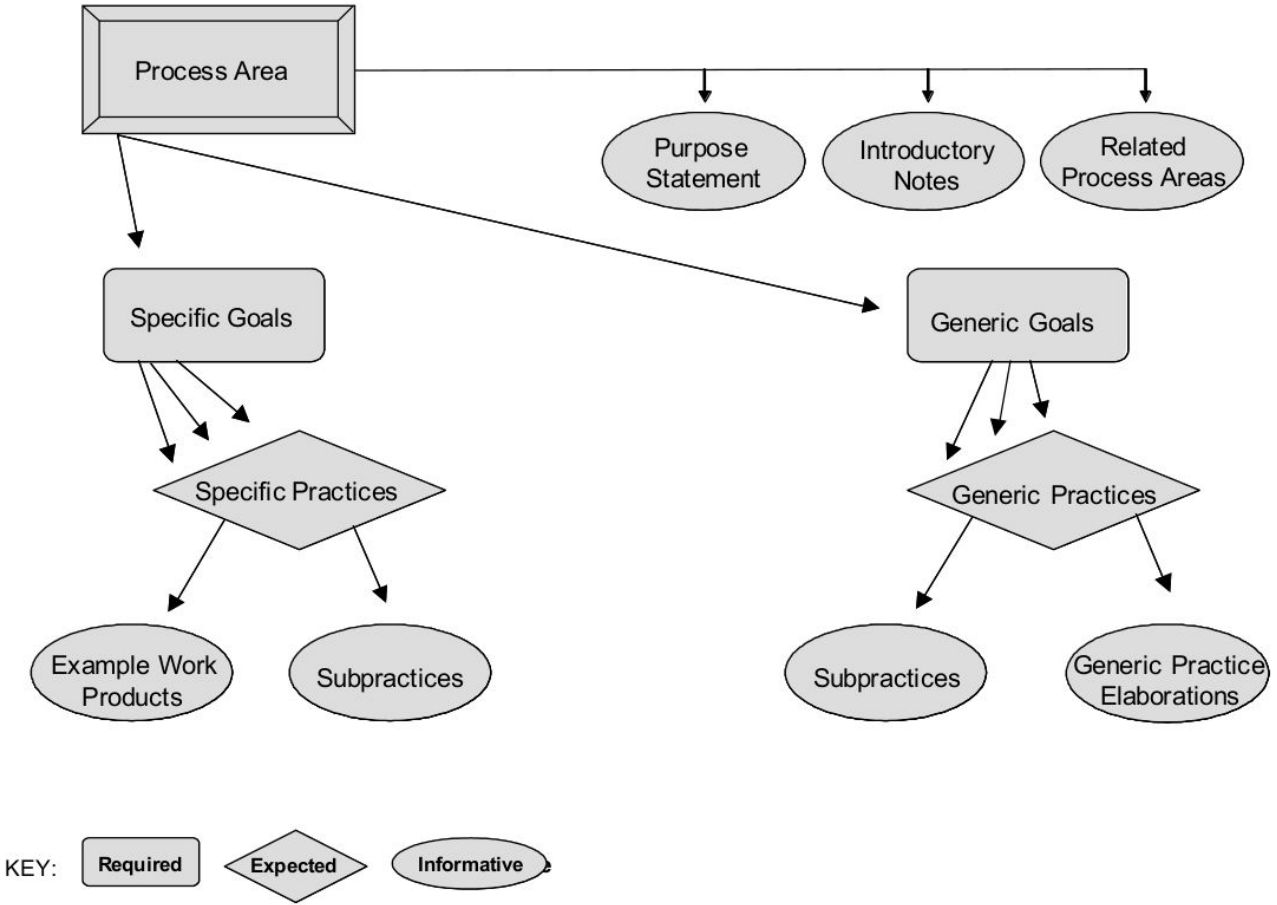


Figure 2.1: CMMI Model Components

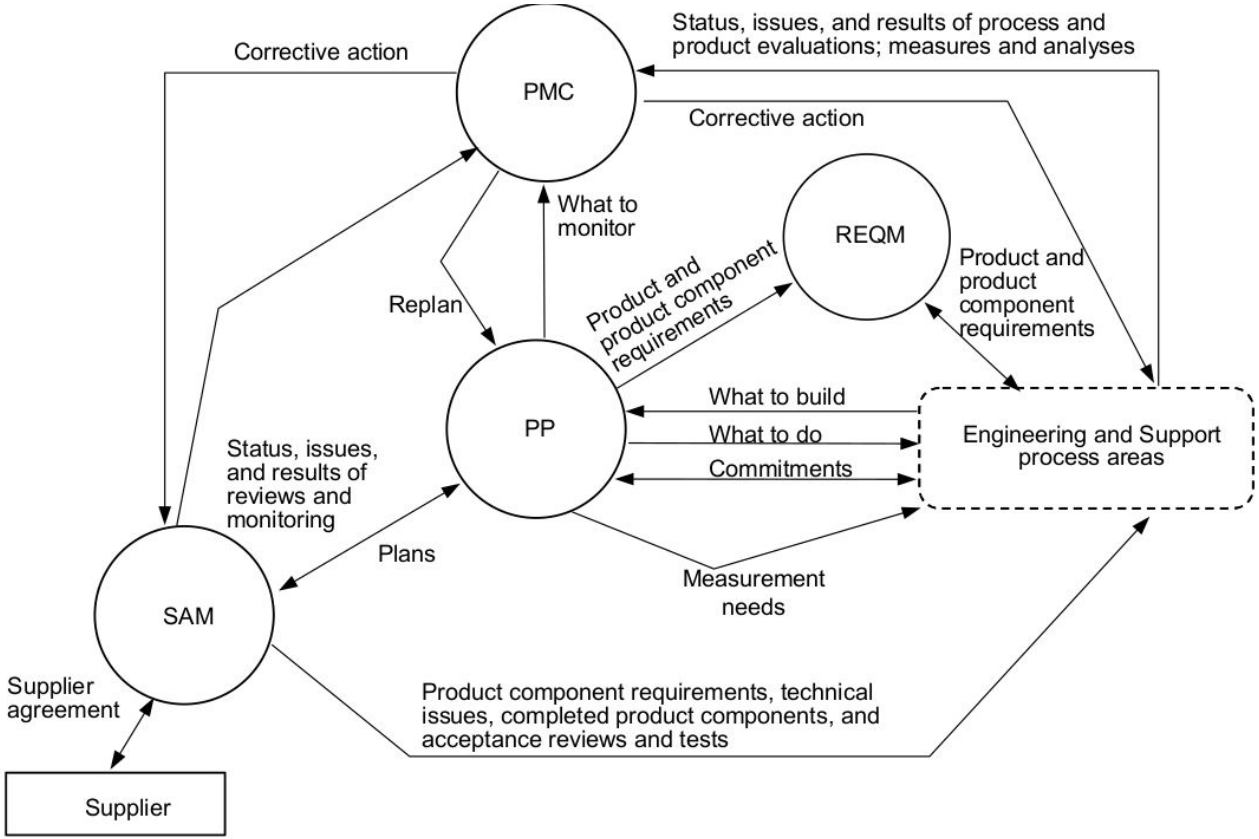
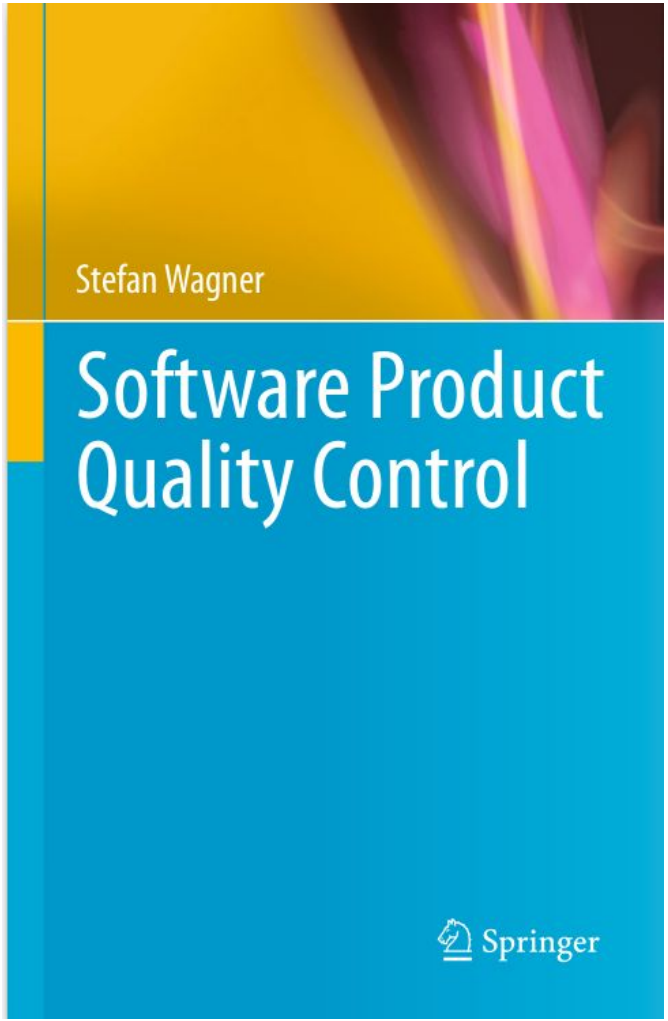


Figure 4.3: Basic Project Management Process Areas


Stefan Wagner
Software Product Quality Control
Springer

Enjeux de la CMMI

- Une approche normative, **prescriptive** pour améliorer les processus
- Un processus **idéal**, décrit par des standards, que **toute** entreprise doit atteindre
- Problème : évaluer la **qualité** d'une manière **indépendante** du **produit** lui-même
  - Postulat : de bons processus font des bons logiciels
  - Vérifié pour l'industrie (TQM)
  - **Beaucoup moins clair pour les processus de développement**

Table 1.1 Delivered defects per thousand function point at CMM levels (based on [110])

CMM level	Minimum	Average	Maximum
1	150	750	4,500
2	120	624	3,600
3	75	473	2,250
4	23	228	1,200
5	2	105	500

Enjeux de la CMMI

- Une approche normative, **prescriptive** pour améliorer les processus
- Un processus **idéal**, décrit par des standards, que **toute** entreprise doit atteindre
- Problème : évaluer la **qualité** d'une manière **indépendante** du **produit** lui-même
  - Postulat : de bons processus font des bons logiciels
  - Vérifié pour l'industrie (TQM)
  - **Beaucoup moins clair pour les processus de développement**
- **Utiliser CMMI comme un guide, mais à ne pas suivre au pied de la lettre**
- **Atteindre les différents "niveaux" n'est pas le but !**
  - Le niveau 2 de CMMI n'est pas forcément mieux que le niveau 2.
- Le but n'est pas de suivre le processus mais de rendre le **client satisfait** !

Standards ISO/IEC pour la qualité logicielle

Standards de qualité logicielle ISO/IEC - **SQuaRE**

- Proposé par :
  - ISO - International Organization for Standardization
  - IEC - International Electrotechnical Commission
- Séries ISO/IEC 25000
  - **SQuaRE**: Software product Quality Requirements and Evaluation
  - La série de standards
- Collection unifiée couvrant principalement 2 processus :
  - Spécification des exigences de qualité logicielle
  - Évaluation de la qualité logicielle
- Remplacent ISO/IEC 9126:1991 et ISO/IEC 14598
- Ils établissent des critères pour la spécification des **exigences**, **mesures** et **évaluation** du logiciel

SQuaRE fournit :

- Des termes et définitions
- Des modèles de référence
- Un guide général
- Des guides individuels à chaque département
- Des standards internationaux pour les spécifications, les exigences, la planification, la gestion, les mesures et l’évaluation

Éléments de SQuaRE

- ISO/IEC 2500n - Quality **Management** Division
- ISO/IEC 2501n - Quality **Model** Division
- ISO/IEC 2502n - Quality **Measurement** Division
- ISO/IEC 2503n - Quality **Requirements** Division
- ISO/IEC 2504n - Quality **Evaluation** Division

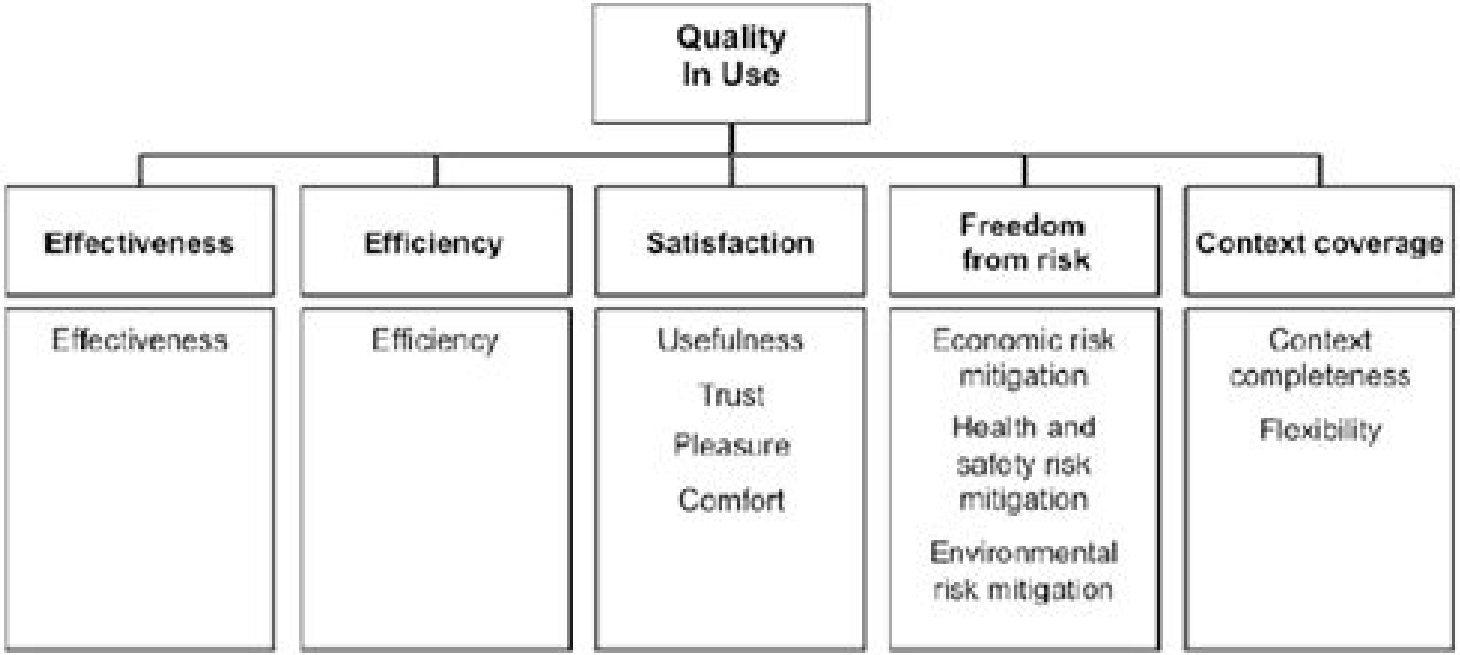
ISO/IEC 2500n - Branche de la gestion de la qualité

- Définit tous les modèles, termes et définitions communs auxquels se réfèrent les autres normes
- Références
- Suggestions pratiques de haut niveau
- **25000** - **Guide to SQuaRE**: modèles d'architecture, terminologie, apperçu de documents
- **25001** - **Planification** et **gestion**: fournit exigences et guide pour un rôle de support responsable de la gestion des exigences sur le produit et de son évaluation.
- "Gestion de la gestion de la qualité"

ISO/IEC 2501n - Branche des modèles de qualité

- Présente un modèle de qualité détaillé
- Qualité logicielle interne
- Qualité logicielle externe
- Qualité logicielle en utilisation
- Guide pratique sur l'utilisation du modèle de qualité
- **25010** - Modèle de qualité (notre objectif !)
- **25012** - Modèle de qualité pour les données

25010 - Modèle pour la qualité à l'utilisation



Quality in use measures -> ISO/IEC 25024

ISO/IEC 25010 - Modèle de qualité pour les logiciels

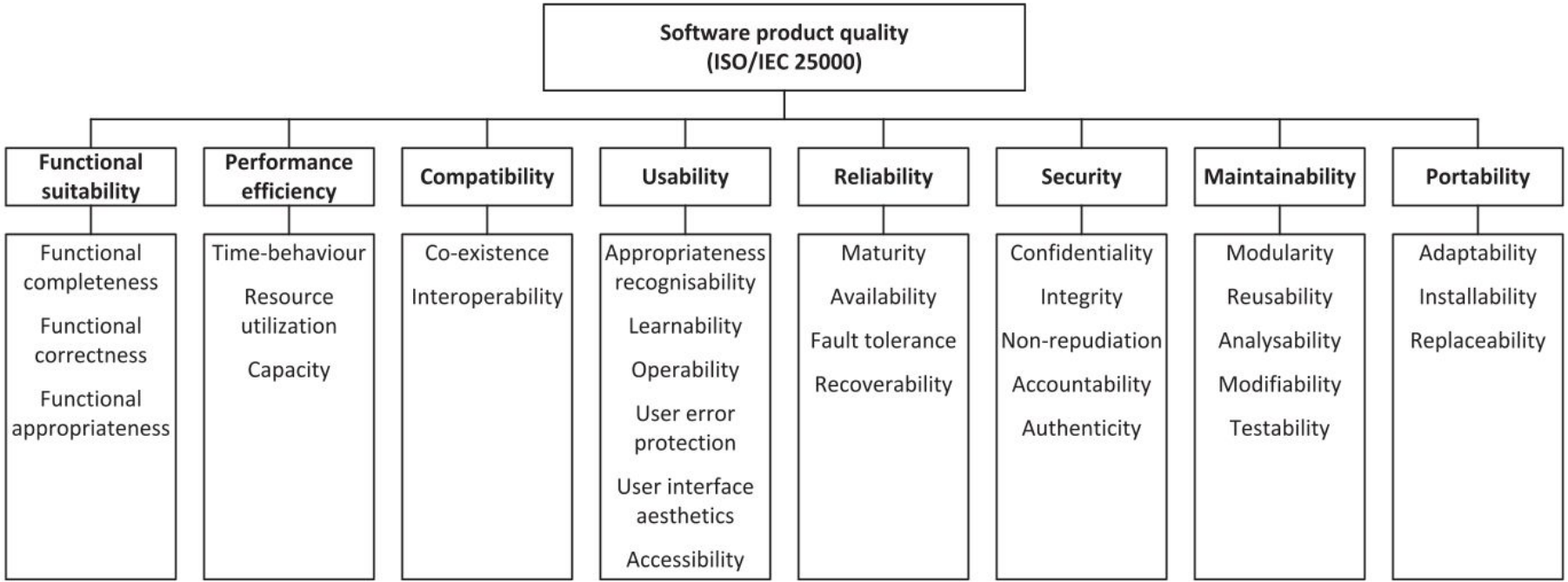


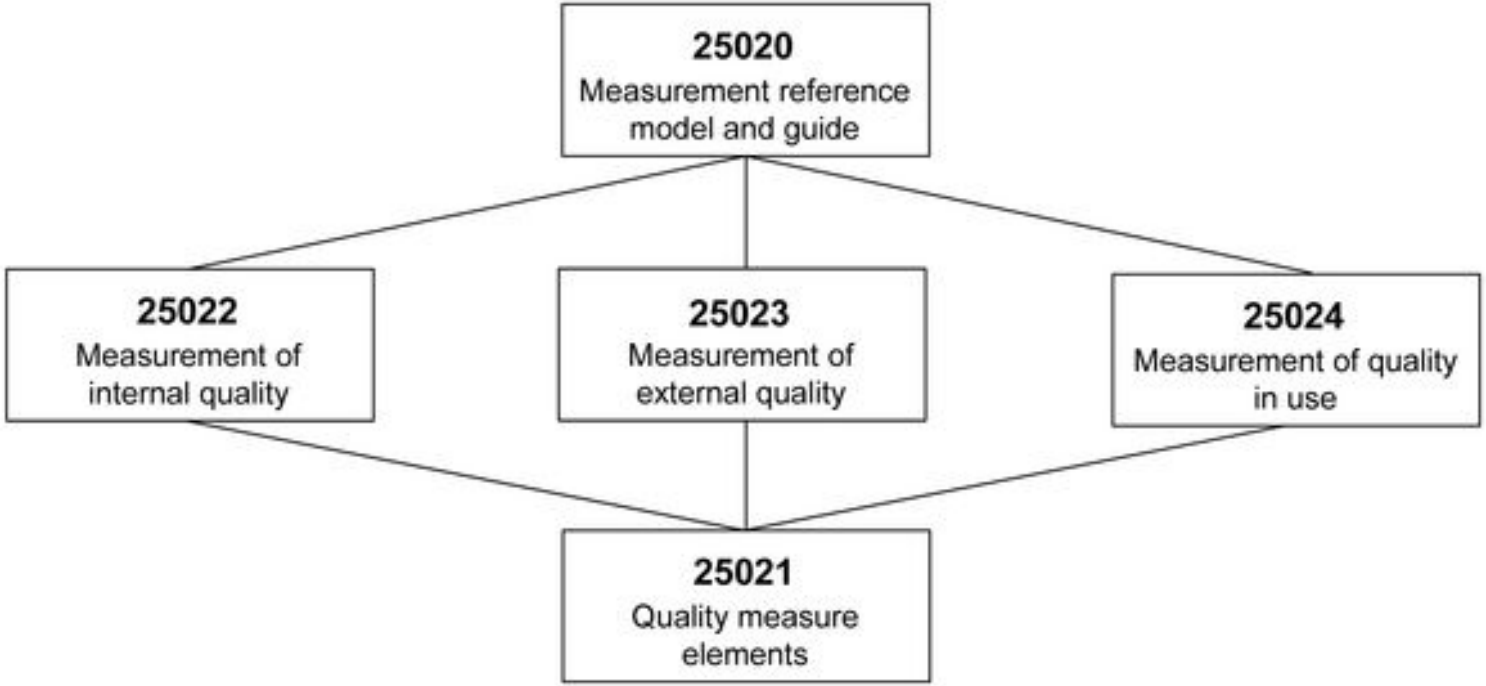
Table 1 — Examples of user needs for quality in use and product quality

User needs	Primary user	Secondary users		Indirect user
		Content provider	Maintainer	
	Interacting	Interacting	Maintaining or porting	Using output
Effectiveness	How effective does the user need to be when using the system to perform their task?	How effective does the content provider need to be when updating the system?	How effective does the person maintaining or porting the system need to be?	How effective does the person using output from the system need to be?
Efficiency	How efficient does the user need to be when using the system to perform their task?	How efficient does the content provider need to be when updating the system?	How efficient does the person maintaining or porting the system need to be?	How efficient does the person using the output from the system need to be?
Satisfaction	How satisfied does the user need to be when using the system to perform their task?	How satisfied does the content provider need to be when updating the system?	How satisfied does the person maintaining or porting the system need to be?	How satisfied does the person using the output from the system need to be?

ISO/IEC 2502n - Branche de mesure de la qualité

- Modèle de référence de mesure de la qualité
- Définition mathématique des mesures de qualité
- Guide pratique à leur utilisation
- **25020** - Modèle de référence et guide pour la mesure
- **25021** - Éléments de mesure de la qualité
- **25022** - Mesure de la qualité interne
- **25023** - Mesure de la qualité externe
- **25024** - Mesure de la qualité à l'utilisation

ISO/IEC 2502n - Branche de mesure de la qualité



ISO/IEC 2503n - Branche des exigences de qualité

- Spécification des exigences de qualité
- Incitation aux exigences de qualité
- **25030** – Exigences de qualité

ISO/IEC 2504n - Branche d’évaluation de la qualité

- Exigences, recommandations et ligne directrice pour l’évaluation de produits logiciels
- **25040** - Guide et modèle de référence d’évaluation
- **25041** - Modules d’évaluation
- **25042** - Processus d’évaluation pour les **développeurs**
- **25043** - Processus d’évaluation pour les **acquéreurs**
- **25044** – Processus d’évaluation pour les **évaluateurs** eux-mêmes

ISO/IEC 2504n - Processus d’évaluation

1. Établir les exigences d’évaluation
    a. Établir le but de l’évaluation
    b. Obtenir les exigences de qualité du logiciel
    c. Identifier les parties du produit devant prendre part à l’évaluation
    d. Définir la rigueur de l’évaluation
2. Spécifier l’évaluation
    a. Sélectionner les métriques de qualité (modules d’évaluation)
    b. Définir les critères de décision pour les métriques de qualité
    c. Établir le critère de décision pour l’évaluation
3. Concevoir l’évaluation
    a. Planifier les activités d’évaluation

ISO/IEC 2504n - Processus d’évaluation (suite)

4. Réaliser l’évaluation
    a. Réaliser les mesures
    b. Appliquer les critères de décision pour les métriques de qualité
    c. Appliquer les critères de décision pour l’évaluation
5. Conclure l’évaluation
    a. Examiner les résultats
    b. Rédiger le rapport
    c. Analyser la qualité d’évaluation et fournir un retour à l’entreprise
    d. Gérer l'élimination des données d’évaluation

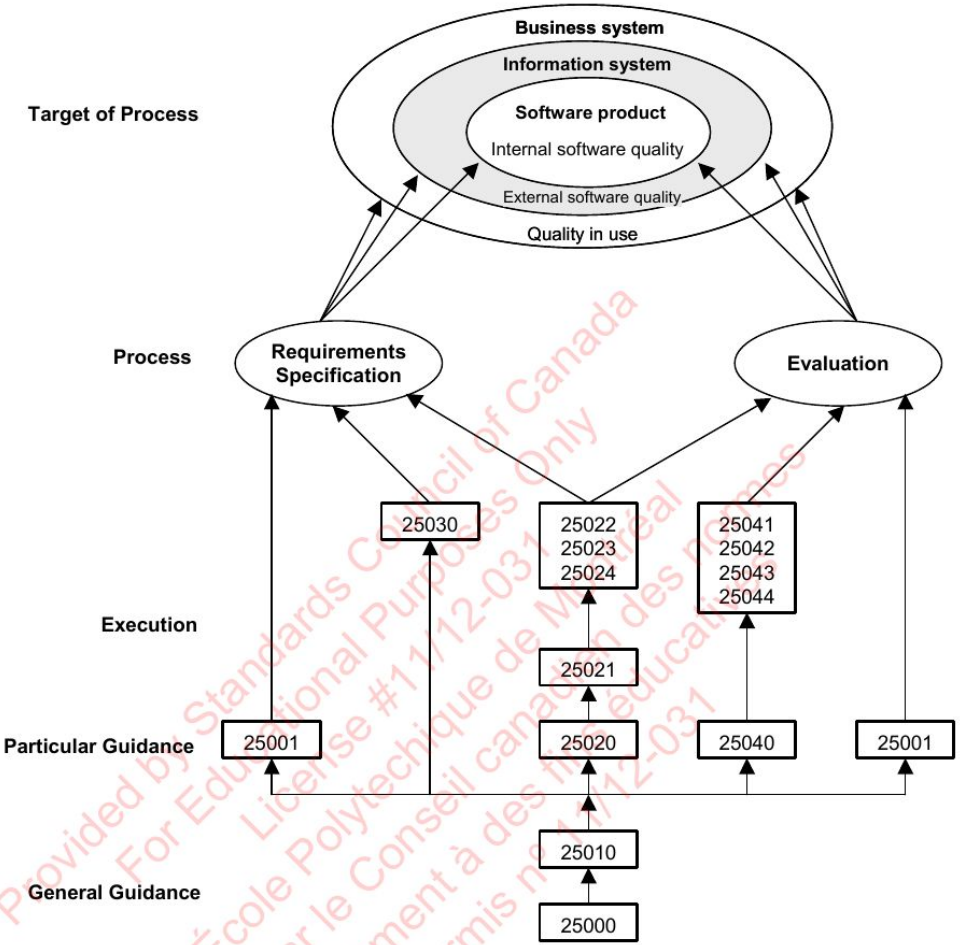


Figure 2 — SQuaRE general reference model

Information and Software Technology 62 (2015) 101–123

Contents lists available at ScienceDirect

Information and Software Technology

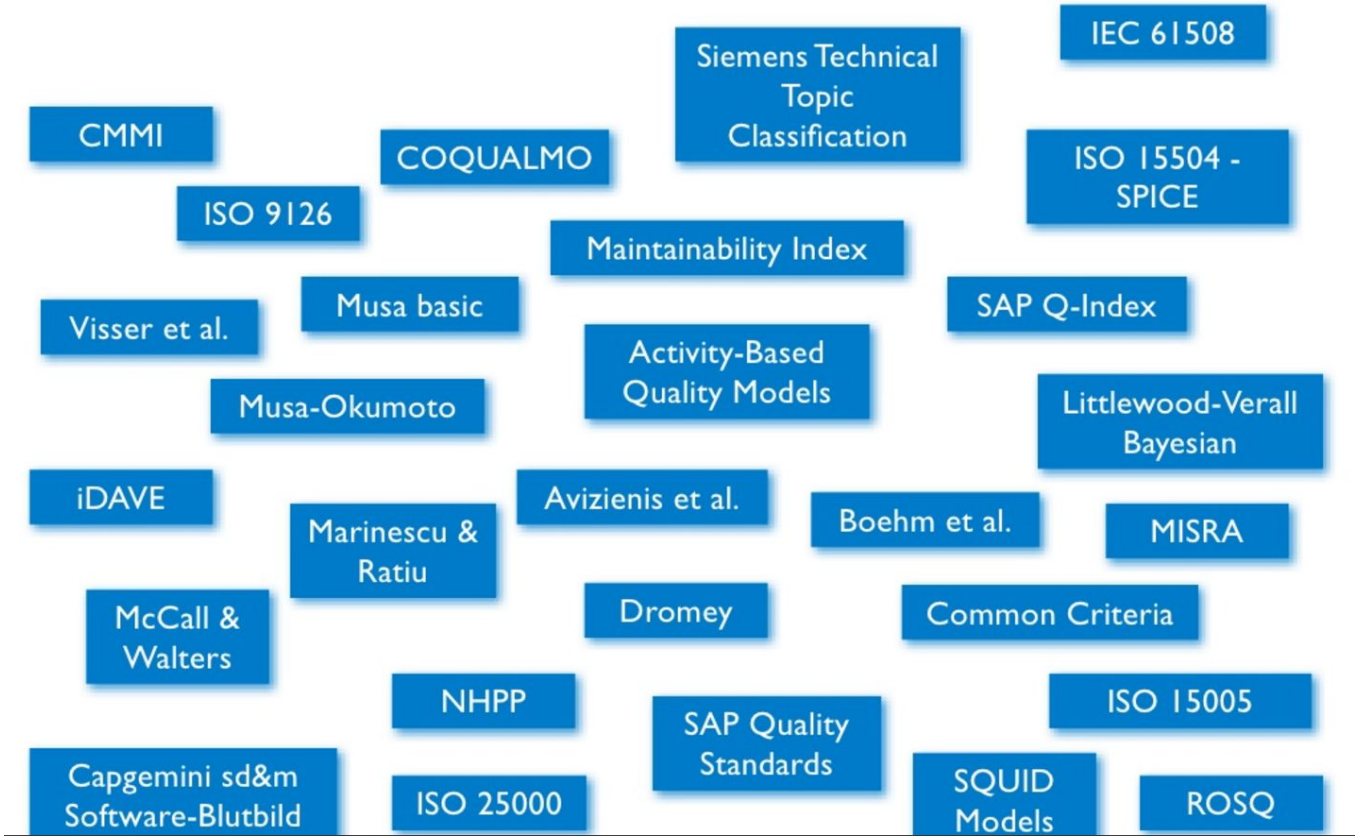
journal homepage: www.elsevier.com/locate/infsof

Operationalised product quality models and assessment: The Quamoco approach

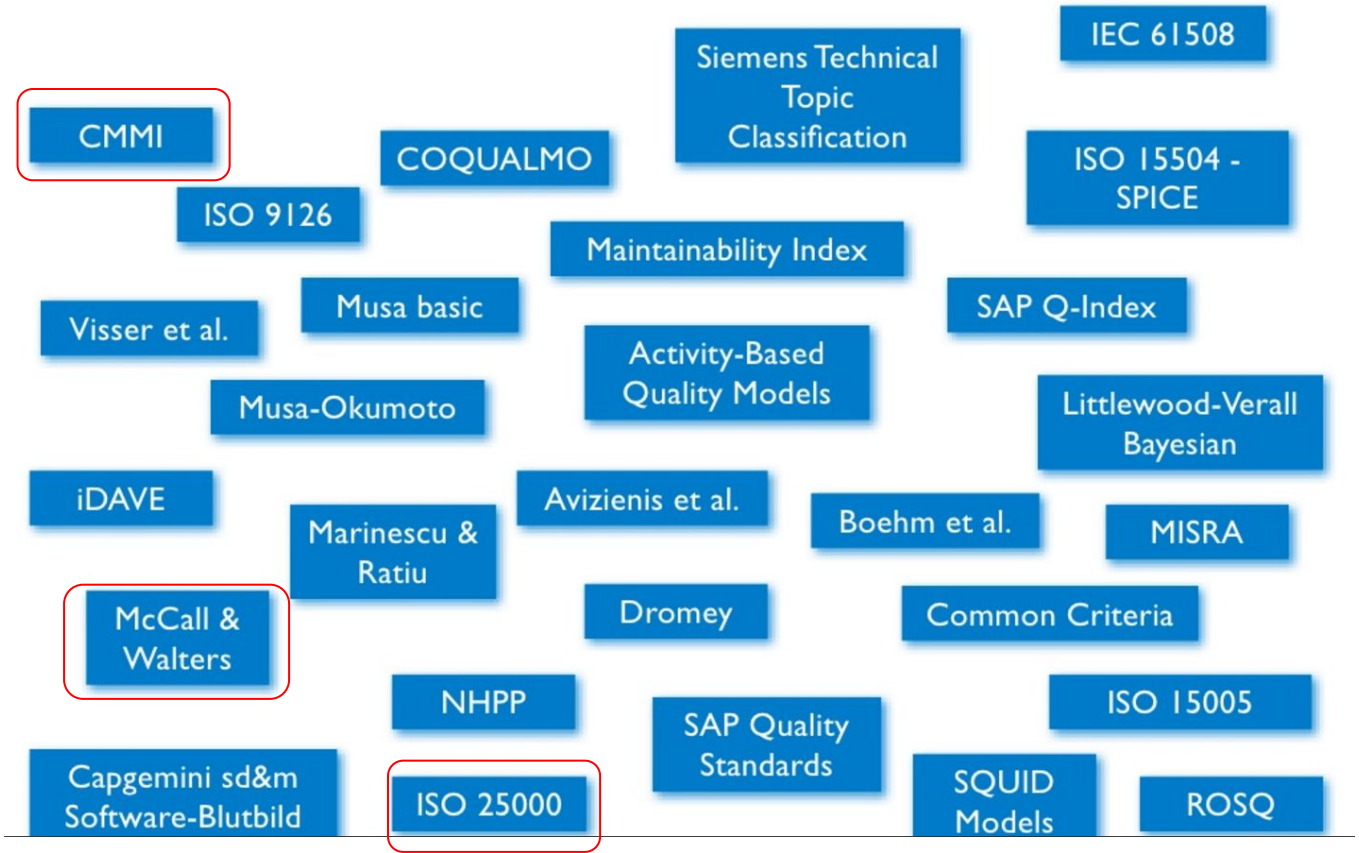
Stefan Wagner [a,*], Andreas Goeb [b], Lars Heinemann [b], Michael Kläs [c], Constanza Lampasona [c], Klaus Lochmann [d], Alois Mayr [e], Reinhold Plösch [e], Andreas Seidl [f], Jonathan Streit [g], Adam Trendowicz [c]

[a] Institute of Software Technology, University of Stuttgart, Stuttgart, Germany
[b] CQSE GmbH, Garching, Germany
[c] Fraunhofer Institute for Experimental Software Engineering IESE, Kaiserslautern, Germany
[d] Institut für Informatik, Technische Universität München, Garching, Germany
[e] Department of Business Informatics, Johannes Kepler University Linz, Linz, Austria
[f] BMW AG, Munich, Germany
[g] itestra GmbH, Munich, Germany

Modèles de qualité logicielle



Modèles de qualité logicielle



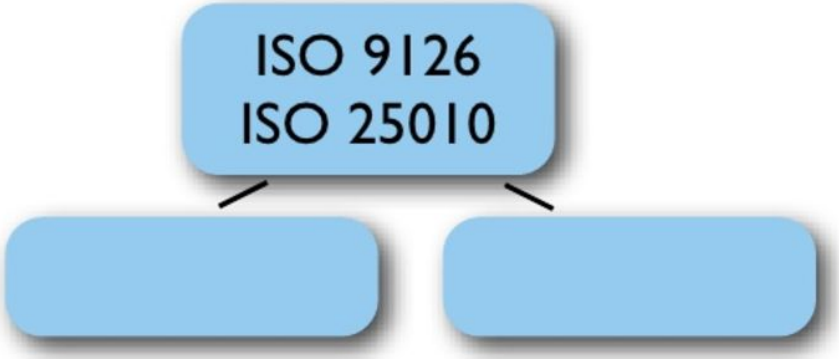
L’utilisation des modèles qualité en pratique

“The -ilities are good for management talk only.”

-Développeur anonyme

(Wagner, 2010)

Quality attribute

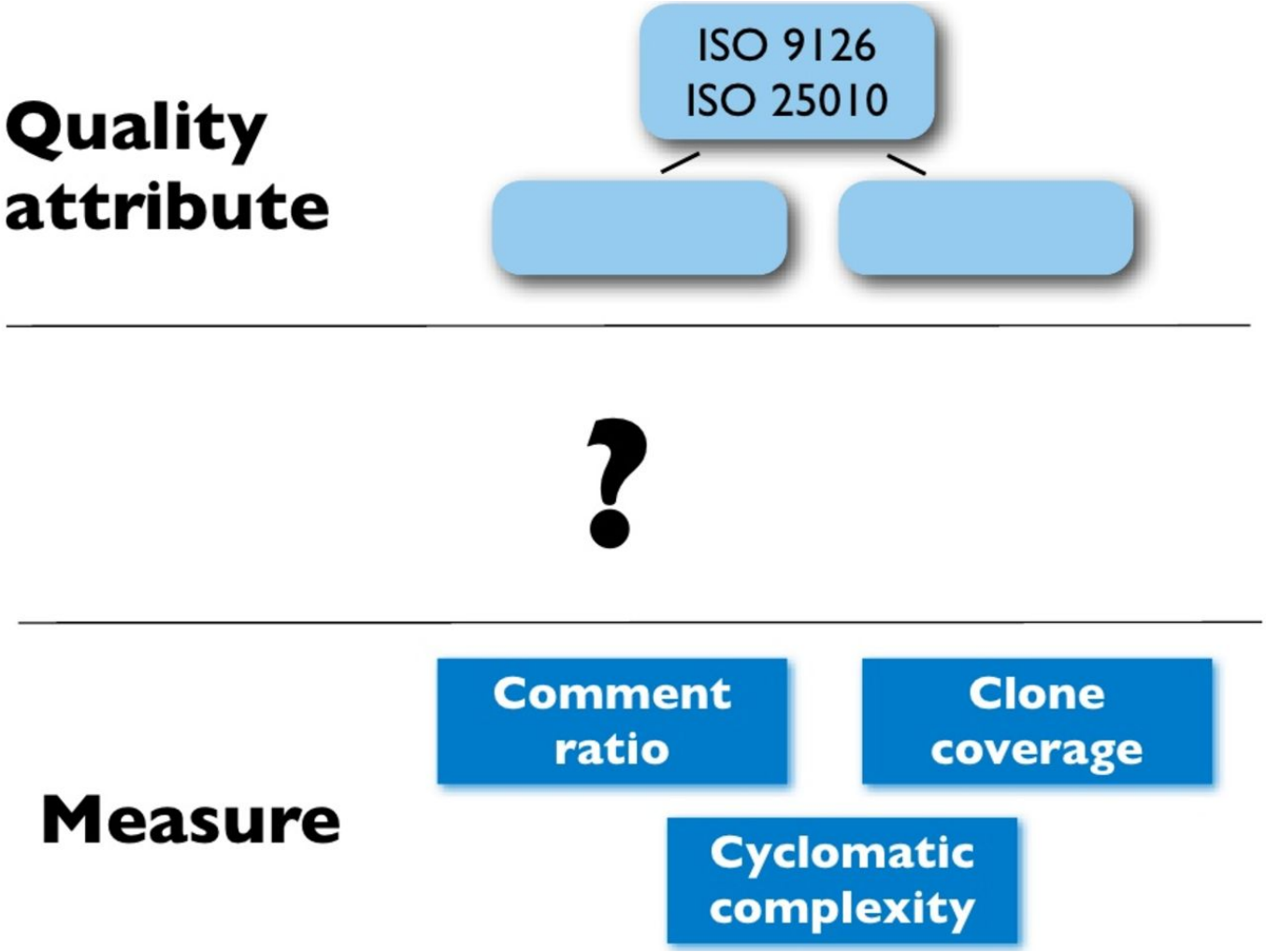


Measure

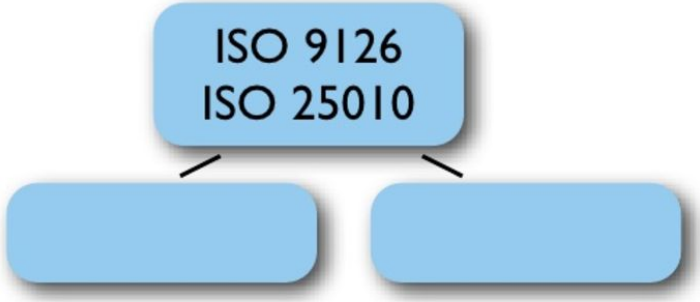
Comment ratio

Clone coverage

Cyclomatic complexity


Quality attribute
ISO 9126
ISO 25010
?
Measure
Comment ratio
Clone coverage
Cyclomatic complexity

Quality attribute



Multitude of models

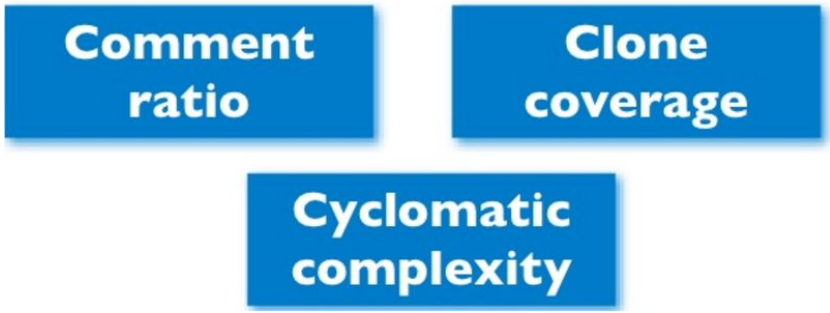
Too abstract

Not operationalised

Not adaptable

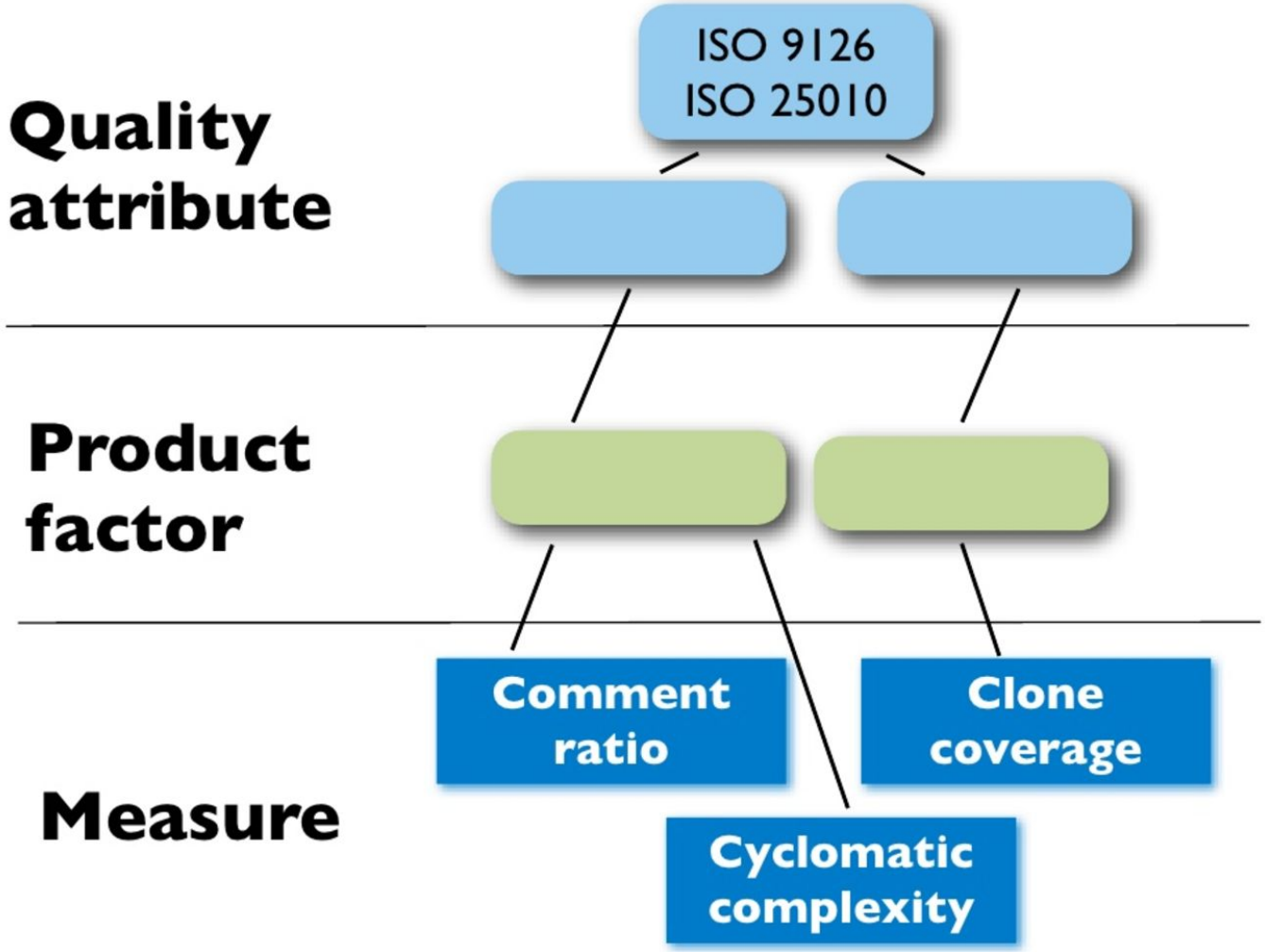
Unreproducible assessments

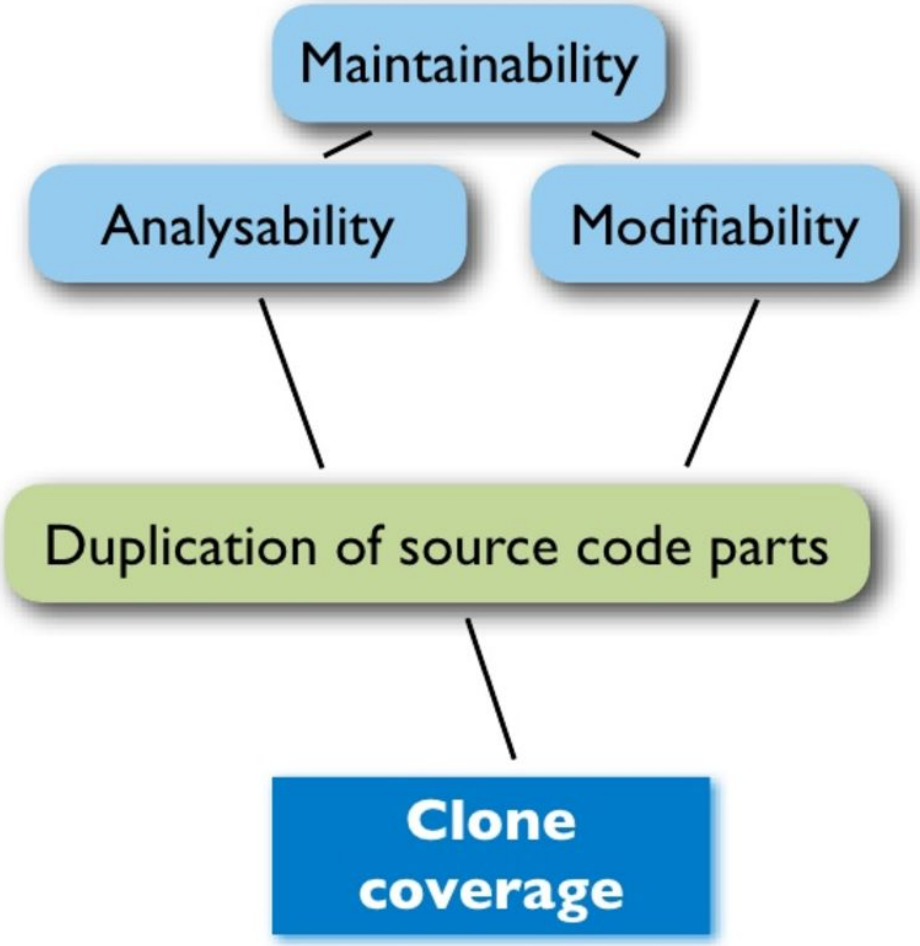
Measure

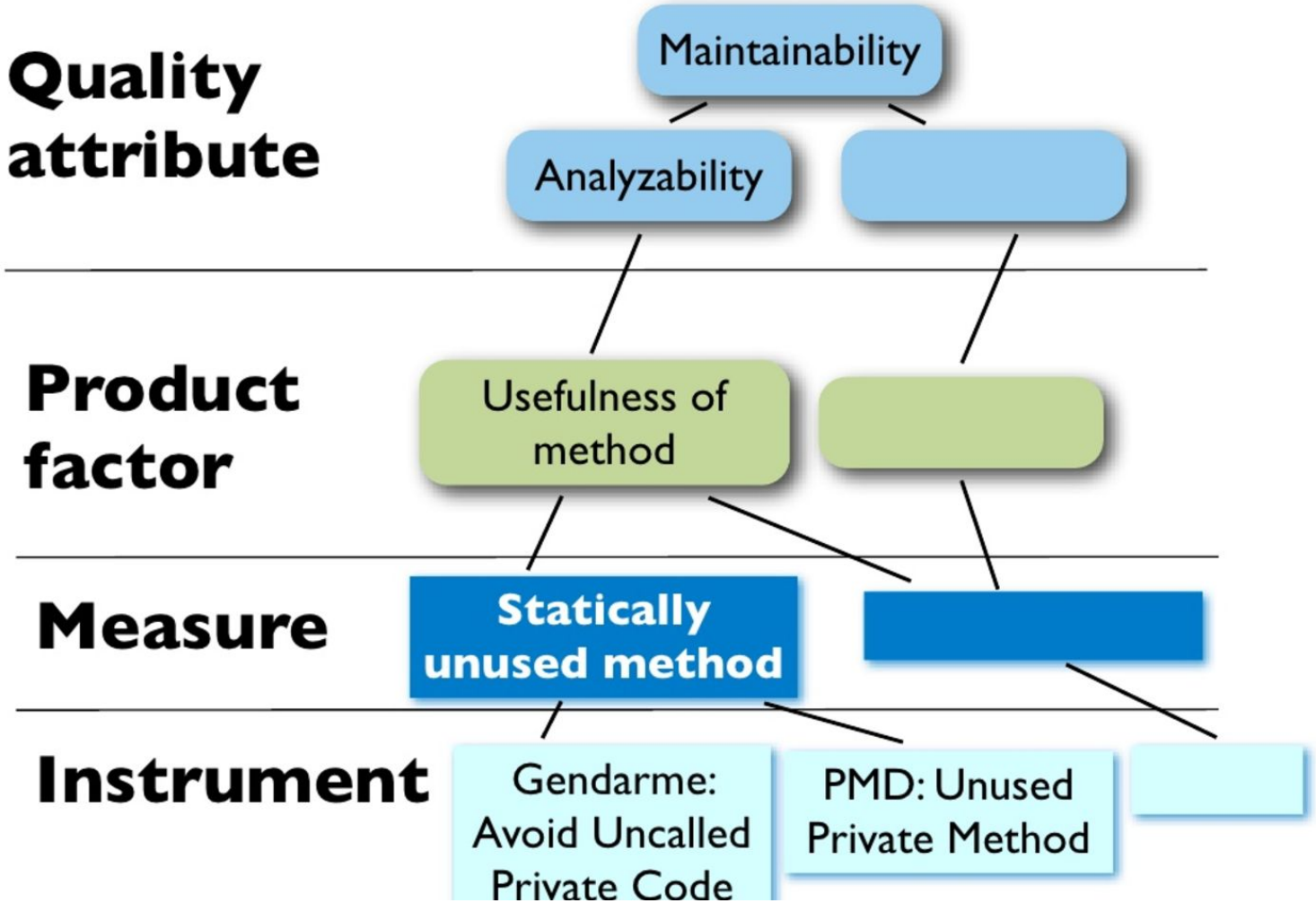


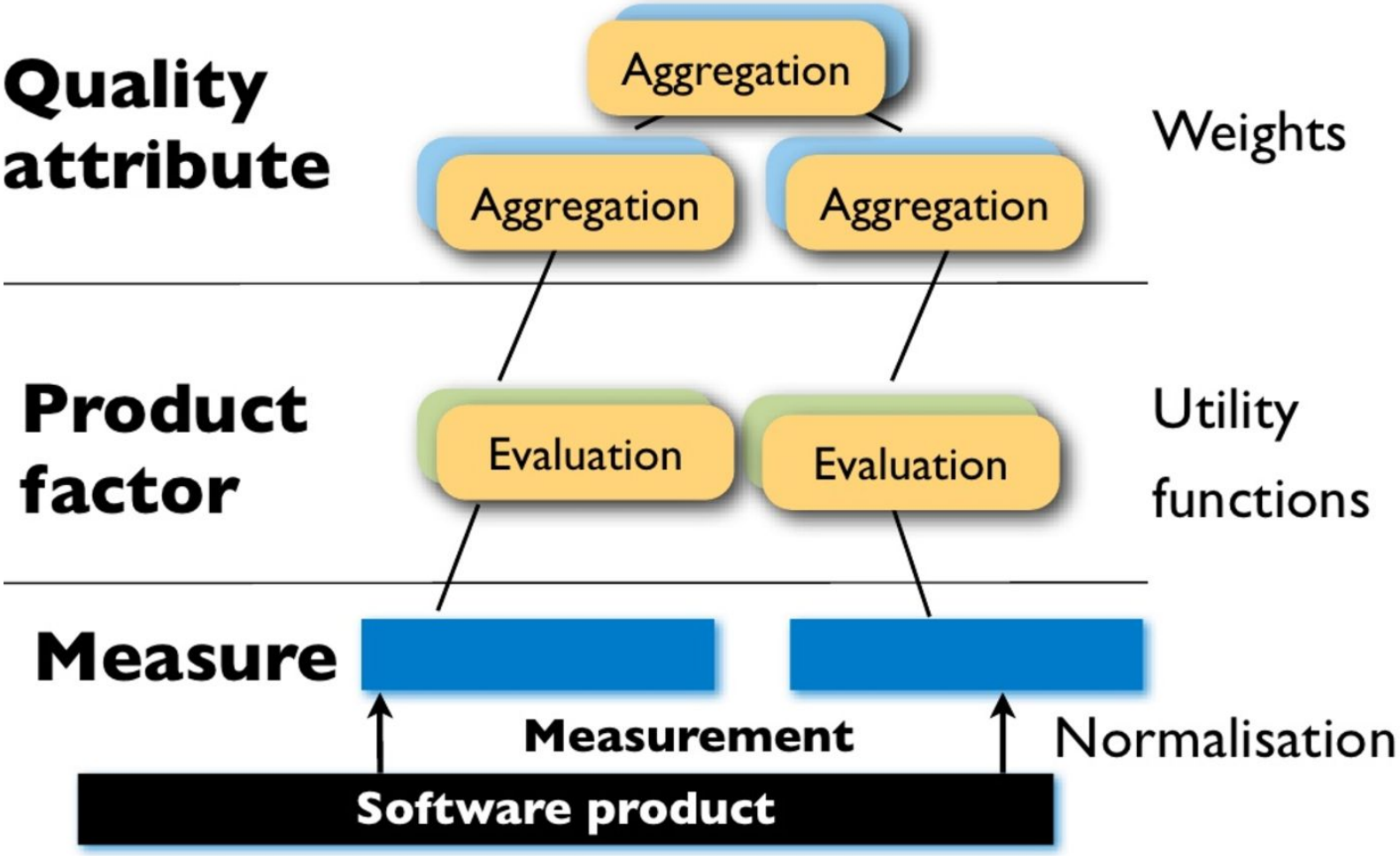
Differing definitions

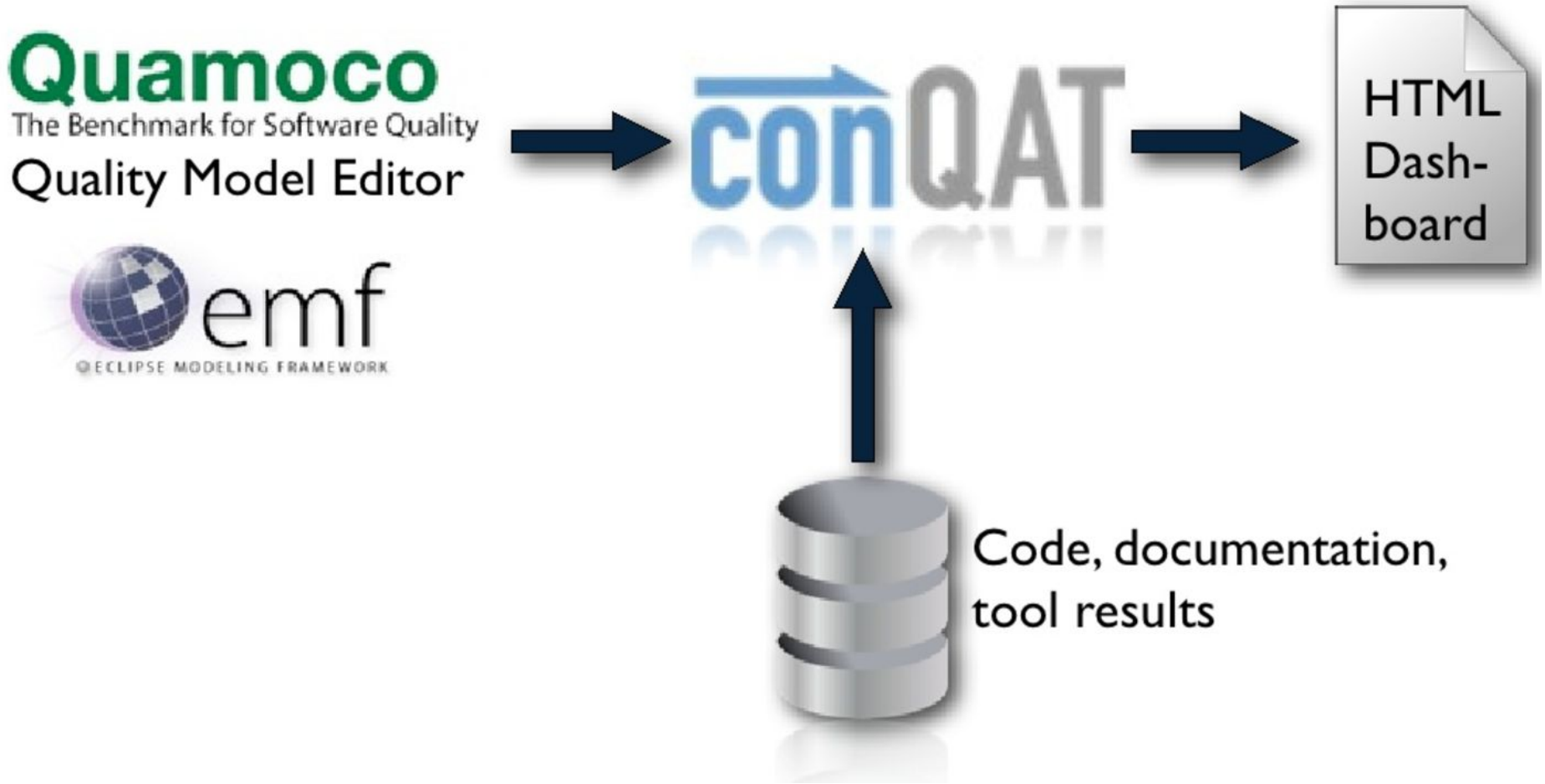
Unclear relationship to quality goals


Quality attribute
ISO 9126
ISO 25010
Product factor
Measure
Comment ratio
Clone coverage
Cyclomatic complexity


Maintainability
Analysability
Modifiability
Duplication of source code parts
Clone coverage


Quality attribute
Maintainability
Analyzability
Product factor
Usefulness of method
Measure
Statically unused method
Instrument
Gendarme: Avoid Uncalled Private Code
PMD: Unused Private Method

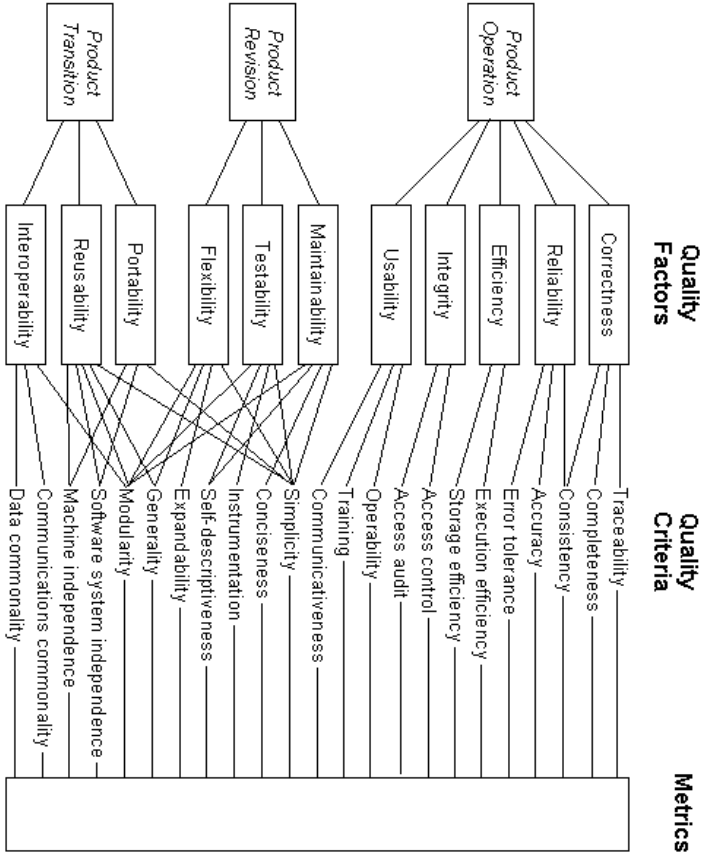

Quality attribute
Aggregation
Aggregation
Aggregation
Weights
Product factor
Evaluation
Evaluation
Utility functions
Measure
Measurement
Normalisation
Software product


Quamoco
The Benchmark for Software Quality
Quality Model Editor
emf
ECLIPSE MODELING FRAMEWORK
conQAT
HTML
Dash-
board
Code, documentation,
tool results

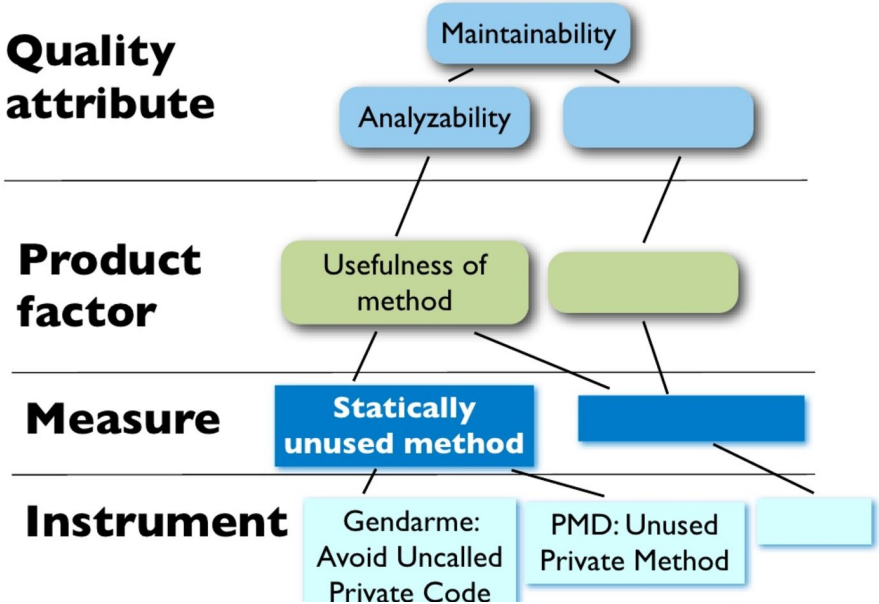
Published under
Apache License, Version 2.0

Available at
www.quamoco.de

McCall's Model (McCall, 1977)



Quamoco Model (Wagner, 2015)



Est ce que le manifeste Agile a du sens ?

Manifesto for Agile Software Development

We are uncovering better ways of developing
software by doing it and helping others do it.
Through this work we have come to value:

Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

That is, while there is value in the items on
the right, we value the items on the left more.

Est ce que Agile et CMMI/ISO sont **opposés** ?

Combiner CMMI/ISO et Agile

- CMMI et ISO apportent une grosse réflexion sur de nombreux aspects du génie logiciel
  - Beaucoup de principes sont utiles
- Agile est adaptatif et évolutif
  - Les différents niveaux de CMMI également !
- Ce sont des états d’esprits pour deux environnements différents

CMMi
maps large org:

centralistic
hierarchy
dominant
slow
process driven
push
non agile
men

Scrum
maps the new org:

federalistic
flat
collaborative
fast
innovative
pull
agile
women

http://www.slideshare.net/borisgloger/cmmi-scrum-keynote-recife-may-08

L’ingénierie de la qualité logicielle

- Approche Classique
  - Le degré de **conformité** aux exigences de fonctionnement, aux standards et aux bonnes pratiques
  - Basé sur les principes de “qualité totale” **(TQM**) principles
    - Processus = Qualité
  - Stratégie de gestion par ordres et contrôles -> Focus sur la **gestion**
  - **Contrats**
  - Utilise la **même approche** quel que soit le type de projet logiciel
- Approche Agile
  - Approche adaptative et **évolutive**
  - Un logiciel qui **fonctionne** sera la principale mesure du progrès réalisé
  - Définition de “**fait**”
  - **Collaboration avec le client**
  - Se concentre sur le **bon** produit, plutôt que sur la bonne manière de le construire
- Modèles qualité
  - McCall’s, éléments de Galin’s, CMMI, ISO 25000, Quamoco

Prochain cours:
Tests logiciels

Maintenant :

Reverse Exam