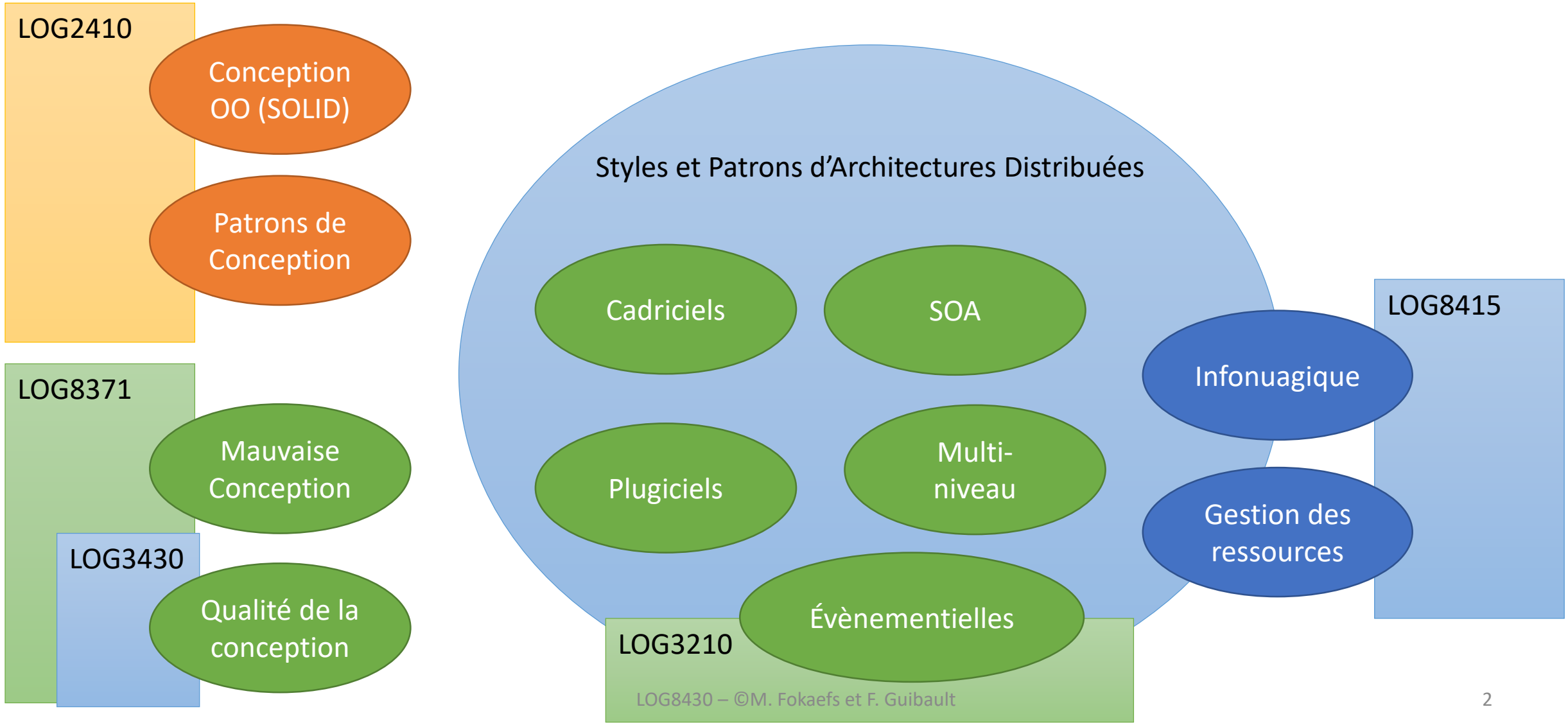
POLYTECHNIQUE MONTRÉAL
UNIVERSITÉ D'INGÉNIERIE

LOG8430 : Qualité de Conception

Carte du cours



Précédemment

LOG2410

Conception OO (SOLID)

Patrons de Conception

Styles et Patrons d'Architectures Distribuées

Aujourd'hui

LOG2410

Conception OO (SOLID)

Patrons de Conception

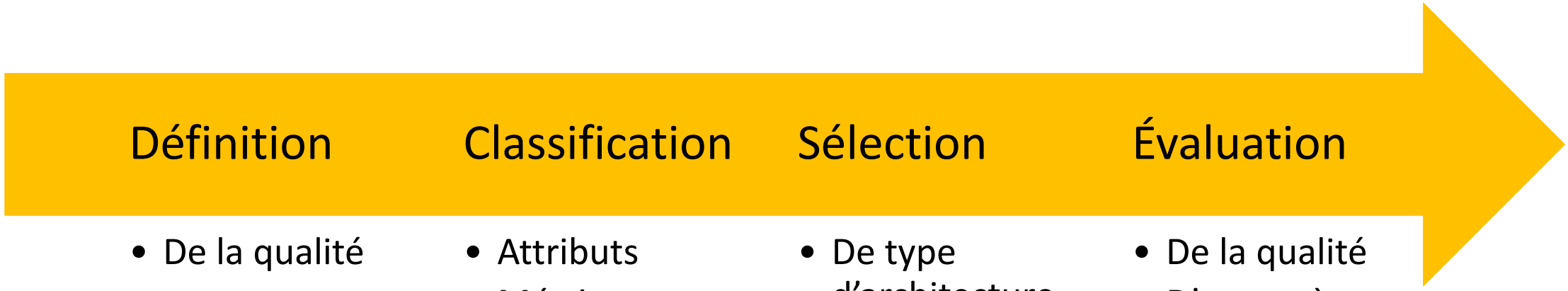
LOG8371

LOG3430

Qualité de la conception

Styles et Patrons d'Architectures Distribuées

Agenda



- De la qualité
- Attributs
- Métriques
- ISO 9126/25010
- De type d'architecture
- Selon des critères de qualité
- De la qualité
- D'un système existant

Après cette séance…

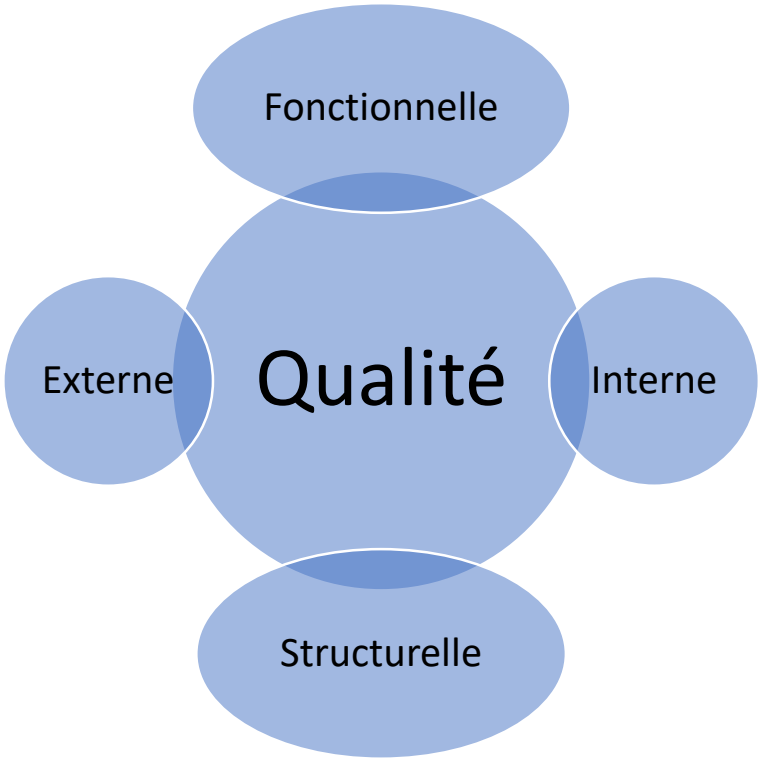
- Vous pourrez reconnaitre et prioriser les attributs de qualité selon les exigences du système.
- Vous apprécierez l'importance d'une haute qualité et de maintenir ce niveau de qualité.
- Vous pourrez mesurer la qualité pour évaluer les logiciels et assurer leur qualité, et pour identifier les instances de mauvaise qualité (en combinaison avec le prochain cours).
- Vous acquerrez une partie des compétences nécessaires pour l'assurance de la qualité (QA).

Qu’est-ce que la qualité de la conception?

…beaucoup de choses!

Définition

- En fait, la qualité inclut plusieurs notions et concepts, qui sont utilisés ensemble pour déterminer et confirmer que le logiciel fonctionne comme il faut et comme désiré.
- On va se concentrer sur la qualité interne structurelle.
- **La qualité est un critère fondamental pour la sélection d'une architecture et pour la conception du logiciel.**



Divers perspectives de la qualité

- La qualité pour l'utilisateur : Le logiciel est utile et il fonctionne comme prévu dans le contexte d'utilisation ?
- La qualité selon la valeur : Le logiciel est évalué basé sur la valeur qu'il produit.
- La qualité du produit : La qualité est mesurée selon les caractéristiques du produit.
- La qualité pour le fabricant : Le logiciel est évalué selon sa conformité aux exigences.

Divers perspectives de la qualité

- La qualité pour l'utilisateur : Le logiciel est utile et il fonctionne comme prévu dans le contexte d'usage.
- La qualité selon la valeur : Le logiciel est évalué basé sur la valeur qu'il produit.
- La qualité du produit : La qualité est mesurée selon les caractéristiques du produit.
- La qualité pour le fabricant : Le logiciel est évalué selon sa conformité aux exigences.

Quelle est la meilleure?

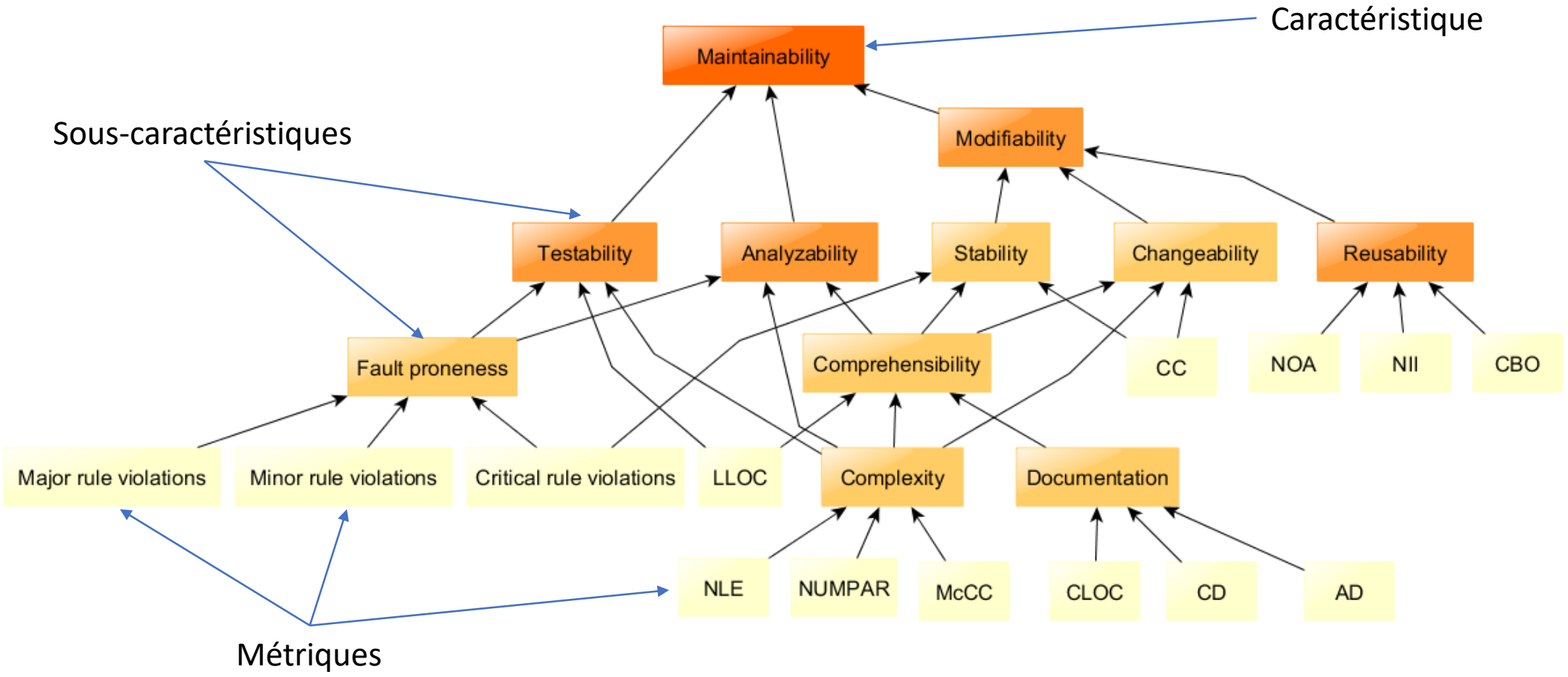
Avant de comparer/évaluer la qualité

- On a besoin de :
- **Un modèle**
  - Quels sont les critères selon lesquels on va évaluer le logiciel?
  - Quelles sont nos priorités pour la conception, l'implémentation et l'utilisation du logiciel?
- **Des métriques**
  - Internes, pendant le développement (mesurées de façon statique)
  - Externes, pendant l'exécution (mesurées de façon dynamique)
  - À l'usage, pendant l'utilisation (mesurées par des humains)
- **Une procédure**
  - Concevoir, planifier, exécuter l'évaluation.
  - Définir le cadre de l'interprétation des résultats.

Le modèle ISO/IEC 9126 (le vieux standard)

Fonctionnalité	Fiabilité	Utilisabilité	Efficacité	Maintenabilité	Portabilité
• Pertinence • Précision • Interopérabilité • Sécurité • Conformité	• Maturité • Tolérance aux pannes • Récupérabilité	• Compréhensibilité • Facilité d'apprentissage • Opérabilité • Attraction	• Utilisation du temps • Utilisation des ressources	• Analysabilité • Changeabilité • Stabilité • Testabilité	• Adaptabilité • Installabilité • Coexistence • Remplaçabilité

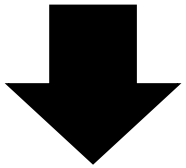
(Demi-)Évolution



Le modèle ISO/IEC 25010:2011

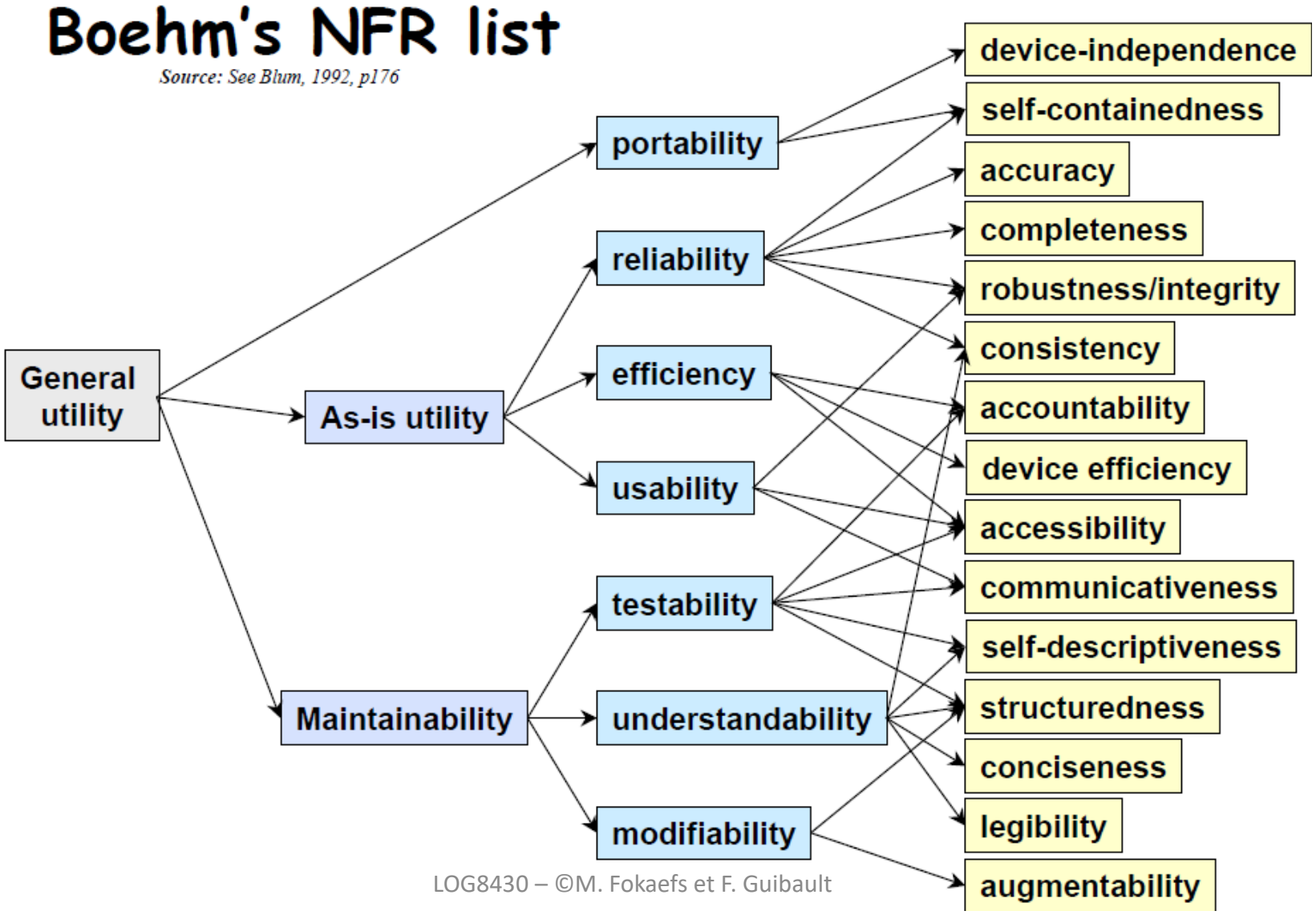
Pertinence Fonctionnelle	Fiabilité	Utilisabilité	Efficacité	Maintenabilité	Portabilité	Sécurité	Compatibilité
• Complétude • Correction • Adéquation	• Maturité • Disponibilité • Tolérance aux pannes • Récupérabilité	• Accessibilité • Facilité d'apprentissage • Opérabilité • Esthétique d'IU • Protection contre les fautes d'utilisateur • Facilité de reconnaissance	• Utilisation du temps • Utilisation des ressources • Capacité	• Modularité • Réutilisabilité • Analysabilité • Possibilité de modification • Testabilité	• Adaptabilité • Installabilité • Remplaçabilité	• Intégrité • Confidentialité • Responsabilité • Authenticité • Non-répudiation	• Coexistence • Interopérabilité

Le modèle ISO/IEC 25010:2011

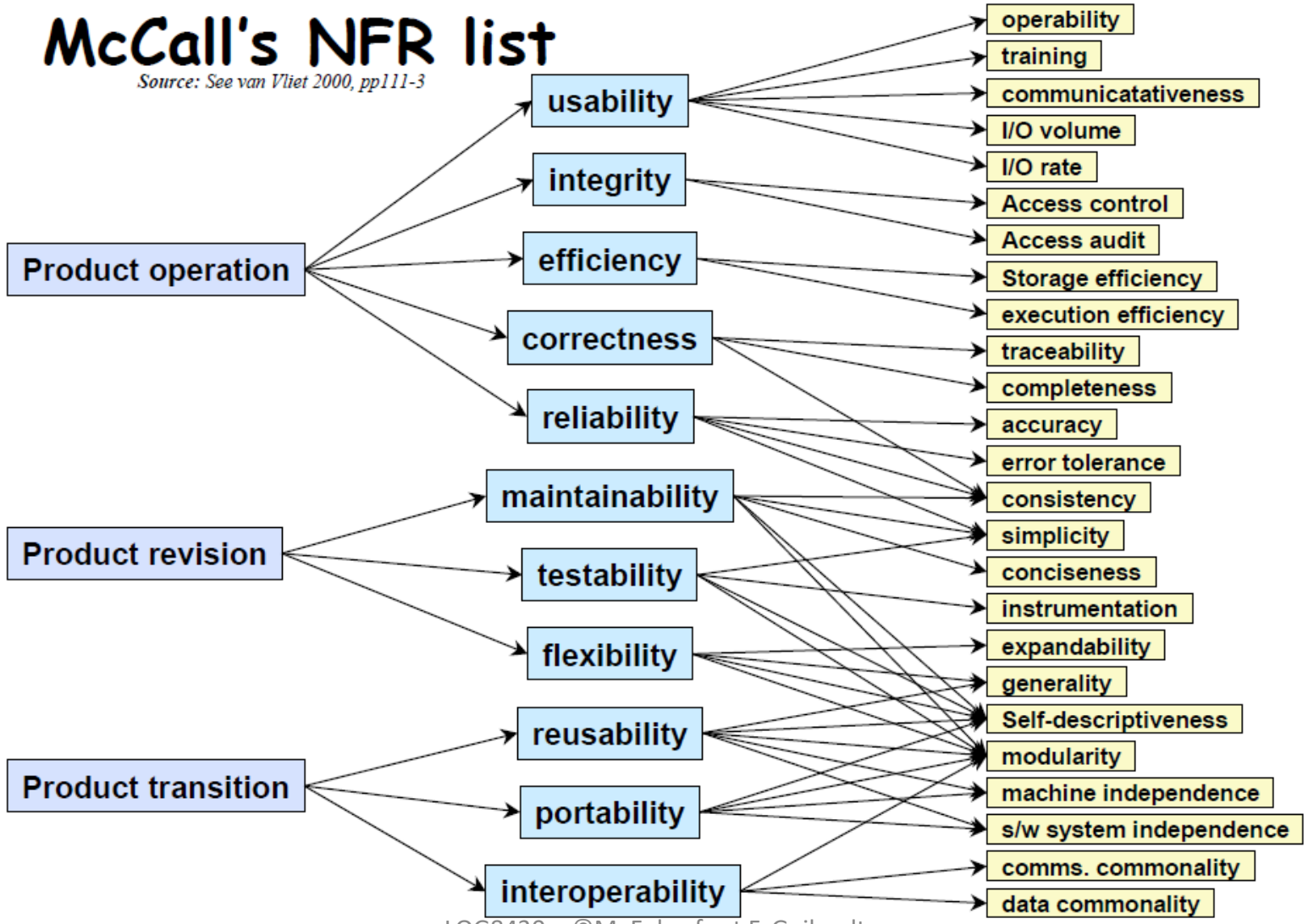


Pertinence Fonctionnelle	Fiabilité	Usabilité	Efficacité	Maintenabilité	Portabilité	Sécurité	Compatibilité
• Complétude • Correction • Adéquation	• Maturité • Disponibilité • Tolérance aux pannes • Récupérabilité	• Accessibilité • Facilité d'apprentissage • Opérabilité • Esthétique d'IU • Protection contre les fautes d'utilisateur • Facilité de reconnaissance	• Utilisation du temps • Utilisation des ressources • Capacité	• Modularité • Réutilisabilité • Analysabilité • Possibilité de modification • Testabilité	• Adaptabilité • Installabilité • Remplaçabilité	• Intégrité • Confidentialité • Responsabilité • Authenticité • Non-répudiation	• Coexistence • Interopérabilité

Autres modèles : Boehm



Autres modèles : McCall



Est-ce qu'on peut mesurer les critères?

Les Métriques

- Taille
  - Lignes du code (LOC)
- Complexité de l'interface
  - Nombre d'attributs et de méthodes ($SIZE2$)
  - Nombre de méthodes locales (NOM)
- Complexité structurelle
  - Complexité cyclomatique de McCabe (CC)
  - Nombre pondéré de méthodes (WMC)
  - Réponse pour une classe (RFC)
- Héritage
  - Profondeur de l'arbre d'héritage (DIT)
  - Nombre d'enfants (NOC)

Les Métriques (suite)

- Couplage
  - Couplage afférent (Ca)
  - Couplage entre les objets (CBO)
  - Autres…
- Cohésion
  - Manque de cohésion entre les méthodes ($LCOM$)
  - Cohésion de classe serrée (TCC)
  - Autres…
- Documentation
  - Manque de documentation (LOD)

Taille

- LOC : Elle compte le nombre des caractères de saut de ligne dans une méthode, une classe, une unité de compilation.
  - Il y a plusieurs variantes : lignes du code source (SLOC), lignes du code sauf la documentation et les lignes blanches.
- SIZE2 : Elle compte le nombre d'attributs et de méthodes d'une classe.
- NOM : Elle compte le nombre de méthodes déclarées dans une classe, pas les méthodes dérivées.
- Des valeurs absolues.
- Quand la LOC, la SIZE2 et la NOM augmentent :
  - Compréhensibilité, Attraction, Analysabilité, Possibilité de modification, Testabilité, Adaptabilité, Facilité d'apprentissage, Stabilité, Remplaçabilité, Interopérabilité, Sécurité, Tolérance aux pannes, Récupérabilité

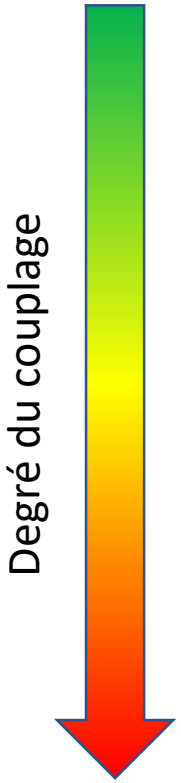
Complexité

- CC : Elle compte le nombre de chemins indépendants dans un morceau de code.
  - Définition1 : $CC = E - V + 2*P$, E=arêtes, V=nœuds, P=composantes connectés
  - Définition2 : compte le nombre d'instructions de branchement (if, while, for, do, switch), les opérandes logiques (II, &&) plus 1 (définition d'Eclipse Metrics et STAN).
- WMC : Elle compte la somme d'une métrique pour toutes les méthodes d'une classe. La métrique peut être la complexité, la LOC ou aucune (unweighted).
  - Définition :$WMC = \sum_{i=1}^{n} CC_i$
- RFC : Elle compte le nombre de méthodes publiques d'une classes et toutes les méthodes accédées directement par ces méthodes publiques.
- Des valeurs absolues.
- Quand la CC, la WMC et la RFC augmentent :
  - Compréhensibilité, Opérabilité, Attraction, Analysabilité, Possibilité de modification, Testabilité, Adaptabilité, Facilité d'apprentissage, Stabilité, Remplaçabilité, Interopérabilité, Sécurité, Tolérance aux pannes, Récupérabilité

Héritage

- DIT : Elle calcule la longueur maximale d'une classe dérivée à une classe de base dans la structure d'héritage du système.
- NOC : Elle compte le nombre de classes immédiatement dérivées d'une classe de base.
- Des valeurs absolues.
- Quand la DIT augment :
  - Compréhensibilité, Opérabilité, Attraction, Analysabilité, Possibilité de modification, Testabilité, Adaptabilité, Facilité d'apprentissage, Stabilité, Remplaçabilité, Interopérabilité, Sécurité, Tolérance aux pannes, Récupérabilité
- Quand la NOC augment :
  - Compréhensibilité, Opérabilité, Attraction, Analysabilité, Possibilité de modification, Testabilité, Adaptabilité, Facilité d'apprentissage, Stabilité, Remplaçabilité,

Couplage (entre deux modules A et B)



Type	Caractéristiques	Désirabilité
Données	A et B utilisent des données partagées	Haute
Structure des données	A et B utilisent des structures de données partagées	OK
Contrôle	A utilise des méthodes de B	Nécessaire
Externe	A et B utilisent un « outil » commun imposé à l'externe. (format de données, appareil, protocole de communication)	Mal
Commun	A et B utilisent les mêmes variables globales.	Mal
Contenu	A utilise le code de B directement ou change l'état de B directement.	Très Mal!

Métriques de Couplage

- CBO : Elle compte le nombre de classes qui sont « couplées » avec une classe. Deux classes A et B sont couplées, si la classe B utilise des méthodes de la classe A.
- Ca : Elle compte le nombre de classes externes qui sont couplées (par la définition de la CBO) avec des classes d'un paquet par rapport au couplage d'entrée (c.-à-d. les classes externes utilisent des méthodes des classes internes).
- Des valeurs absolues.
- Quand la CBO augment :
  - Compréhensibilité, Opérabilité, Attraction, Analysabilité, Possibilité de modification, Testabilité, Adaptabilité, Facilité d'apprentissage, Interopérabilité, Sécurité, Tolérance aux pannes, Récupérabilité
- Quand la Ca augment :
  - Opérabilité, Attraction, Facilité d'apprentissage, Stabilité, Remplaçabilité, Interopérabilité, Sécurité, Tolérance aux pannes

Cohésion (entre les parties d'un module)

Type	Caractéristiques	Désirabilité
Données	Toutes font partie d'une abstraction des données	Très Haute
Fonctionnelle	Toutes résolvent le même problème.	Haute
Séquentielle	La sortie d'une est l'entrée d'une autre.	OK
Communication	Des opérations qui utilisent la même entrée ou la même sortie.	Modéré
Procédurale	Une série d'opérations qui doivent s'exécuter en ordre.	Mal
Temporelle	Des éléments qui fonctionnent en même temps.	Mal
Logique	Des éléments qui font des tâches logiquement semblables.	Très Mal!
Par coïncidence	Des éléments qui n'ont aucune relation conceptuelle mais qui ont du code répliqué.	Très Mal!

Métriques de Cohésion

- LCOM : Plus de 6 définitions!
- On va utiliser celle de Henderson-Sellers :

$$LCOM96a = \frac{(\frac{1}{a}\sum_{j=1}^{a}\mu(Aj)) - m}{1 - m}$$

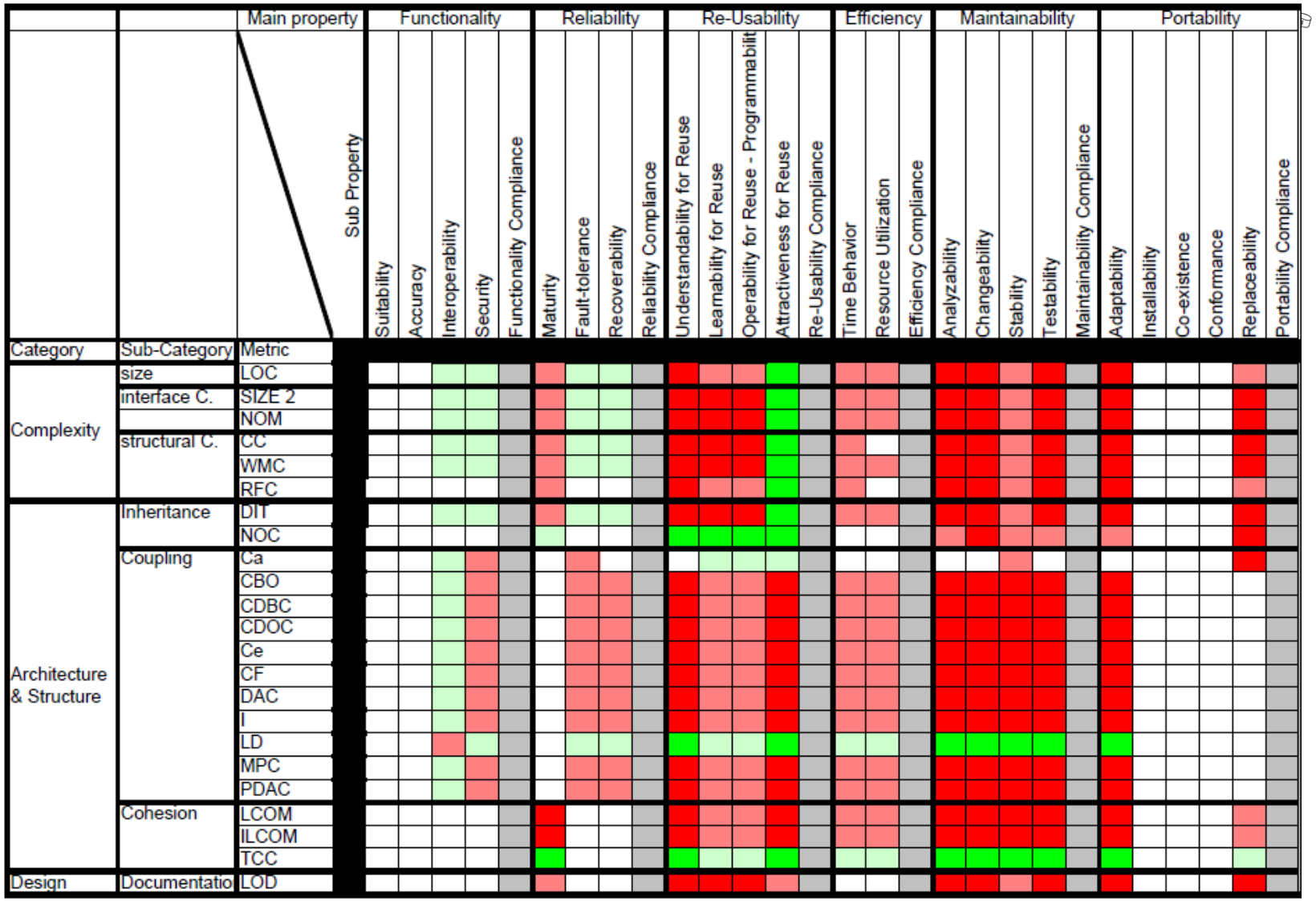
  - α est le nombre d'attributs d'une classe, *m* est le nombre de méthodes, *μ(Aj)* est le nombre de méthodes qui accèdent à l'attribut Aj.
- Mais pourquoi est-ce si difficile de calculer LCOM?
  - Il y a des classes sans attributs, ou avec une seule méthode, ou avec un seul attribut.
  - L'héritage complique la situation au niveau des attributs et des méthodes hérités.
  - Il y a les méthodes d'accès et de modification.
- TCC = NDP/NP, NDP = nombre de paires de méthodes qui accèdent aux attributs directement, NP = nombre de paires de méthodes (n(n-1)/2)
- Des pourcentages (valeurs normalisées) LCOM=0 ➔ Cohésion maximale, TTC=1 ➔ Cohésion maximale
- Deux des métriques les plus fiables.
- Quand la LCOM diminue ou la TCC augment :
  - Maturité, Compréhensibilité, Attraction, Analysabilité, Possibilité de modification, Testabilité, Adaptabilité, Facilité d'apprentissage, Stabilité, Remplaçabilité

Documentation

- LOD : En assumant un commentaire par classe et un commentaire par méthode, elle compte le pourcentage des éléments qui ne sont pas commentés.
  - LOD = 1 – #comm/(m+c+1), #comm=nombre de commentaires, m=nombre de méthodes, c=nombre de classes internes.
- Pourcentage (valeur normalisée)
- Quand la LOD augment :
  - Compréhensibilité, Opérabilité, Attraction, Maturité, Analysabilité, Possibilité de modification, Testabilité, Adaptabilité, Facilité d'apprentissage, Stabilité, Remplaçabilité

Matrice des métriques

Software Quality Matrix



Legend

not evaluated	
highly directly,	directly related
highly inversly,	inversly related
not related	

Bonus : Placement d'Entité

- L'ensemble des entités d'une classe est constitué de tous les attributs et de toutes les méthodes de la classe.
- L'ensemble des entités d'une méthode est constitué de tous les attributs et de toutes les méthodes qui sont utilisés par la méthode.
- L'ensemble des entités d'un attribut est constitué de toutes les méthodes qui accèdent à l'attribut.
- La distance entre une entité, *e* (méthode ou attribut), et une classe, *C:*

$$distance(e, C) = 1 - \frac{|S_e \cap S_C|}{|S_e \cup S_C|}, where\ S_C = \bigcup \{e_i\}.$$

- Le degré de placement des entités dans une classe:

$$EntityPlacement_C = \frac{\frac{\sum_{e_i \in C} distance(e_i, C)}{|entities \in C|}}{\frac{\sum_{e_j \notin C} distance(e_j, C)}{|entities \notin C|}}$$

Bonus : Placement d'Entité

- EP est une métrique qui combine les concepts de la cohésion et du couplage.
- Des violations de cette métrique sont résolues par des mouvements des membres de la classe ou par l'extraction des membres vers une nouvelle classe.
- Elle est utilisé dans l'outil JDeodorant (au prochain cours!) pour évaluer l'impact des refactorings sur la qualité du logiciel.

Sélection de type d'architecture selon les critères de qualité

Haoues, M., Sellami, A., Ben-Abdallah, H., & Cheikhi, L. (2017). A guideline for software architecture selection based on ISO 25010 quality related characteristics. International Journal of System Assurance Engineering and Management, 8(2), 886-909.

Comparaison des architectures logicielles par rapport aux caractéristiques et sous-caractéristiques de qualité (1)

ISO 25010 characteristics	ISO 25010 sub-characteristics	SOA	MDA	CBA	EDA	AOA
Functional Suitability	Functional Completeness	0	0	+	0	0
	Functional Correctness	+	0	+	0	+
	Functional appropriateness	0	0	0	0	0
Reliability	Maturity	−	0	0	−	0
	Availability	0	+	+	−	+
	Fault Tolerance	−	+	+	+	+
	Recoverability	0	0	+	0	+
Usability	Appropriateness Recognizability	+	0	0	0	0
	User interface aesthetics	−	+	0	0	0
	Operability	0	+	0	0	0
	User error protection	−	0	0	0	0
	Learnability	−	0	0	0	0
	Accessibility	+	+	0	+	0
Compatibility	Coexistence	+	+	0	0	0
	Interoperability	+	+	+	+	+

- SOA = Architecture orienté service
- MDA = Architecture orienté modèle
- CBA = Architecture des composantes
- EDA = Architecture évènementielle
- AOA = Architecture orienté aspect

Comparaison des architectures logicielles par rapport aux caractéristiques et sous-caractéristiques de qualité (2)

		SOA	MDA	CBA	EDA	AOA
Performance efficiency	Time Behaviour	−	0	+	+	0
	Resource Utilization	+	+	0	+	+
	Capacity	−	−	−	+	+
Portability	Adaptability	+	+	+	+	+
	Instability	−	−	−	0	+
	Replaceability	+	+	+	0	0
Security	Confidentiality	−	−	0	−	0
	Integrity	−	+	+	−	0
	Non-repudiation	0	−	0	0	0
	Accountability	0	0	0	0	0
	Authenticity	−	−	0	0	0
Maintainability	Modularity	+	+	+	+	+
	Reusability	+	+	+	+	−
	Analyzability	+	0	+	0	0
	Modifiability	+	+	+	−	+
	Testability	−	0	−	0	0

- SOA = Architecture orienté service
- MDA = Architecture orienté modèle
- CBA = Architecture des composantes
- EDA = Architecture évènementielle
- AOA = Architecture orienté aspect

Relations entre les caractéristiques de qualité

	Functional suitability	Performance efficiency	Usability	Compatibility	Reliability	Security	Maintainability	Portability
Functional suitability	+	−	+	0	+	−	+	0
Performance efficiency	0	+	−	−	0	−	−	−
Usability	+	±	+	0	+	0	±	0
Compatibility	0	0	0	+	0	−	±	+
Reliability	+	0	+	0	+	0	+	0
Security	0	−	−	−	+	+	0	0
Maintainability	+	−	0	+	±	±	+	+
Portability	0	−	0	+	0	0	+	+

Guide de sélection d'architecture

	Functional suitability	Performance efficiency	Usability	Compatibility
1st choice	CBA	EDA	MDA	SOA|MDA
2nd choice	SOA|AOA	AOA	EDA	CBA|EDA|AOA
3rd choice	MDA|EDA	MDA|CBA	CBA|AOA	
	Reliability	Security	Maintainability	Portability
1st choice	CBA|AOA	CBA	MDA	AOA
2nd choice	MDA	AOA	SOA|CBA	EDA
3rd choice	EDA	EDA	EDA|AOA	SOA|MDA|CBA