Chapitre 3
Mesure de produits logiciels

1. Introduction

De nombreux travaux montrent que les attributs internes du produit logiciel ont un impact sur ses attributs externes.

– Une bonne structure interne donne une bonne qualité du produit
– Agir sur la structure interne permet de contrôler la qualité du produit
_ Les attributs internes peuvent être mesurés très tôt dans le cycle de développement
– Grâce à la prédiction, des actions correctives peuvent être entreprises, réduisant ainsi les coûts de maintenance

Attributs internes

Produit	Attributs internes (exemples)
Spécification	Taille, réutilisation, modularité, fonctionnalité, redondance, …
Conception	Taille, réutilisation, modularité, couplage, cohésion, fonctionnalité, …
Code	Taille, réutilisation, modularité, couplage, cohésion, fonctionnalité, complexité algorithmique, structuration du flot de contrôle, …
Données de test	Taille, couverture, …

Historiquement, un des premiers aspects mesurés du logiciel . Similaire à l'attribut taille d'un humain (attributs grandeur, poids), _ Par analogie, les attributs importants de la taille sont :

– Taille
– Taille fonctionnelle
– Complexité

2. Taille

_ C'est la taille physique d'un produit. Elle peut être mesurée aussi bien au niveau du code qu'au niveau de la spécification ou de la conception.

a. Code

– La mesure la plus utilisée est le Nombre de Lignes de Code
– Mesure ambiguë : différentes façons de la mesurer
• Inclusion ou non des commentaires et des déclarations
• Comment compter si une ligne qui contient plusieurs instructions

– La définition la plus répondue est :
« Est considérée comme *ligne de code* toute ligne qui n'est pas un commentaire ni une ligne vide. Le nombre d'instructions ou de parties d'instructions n'est pas important ». Ceci inclut les en-têtes de programmes, les déclarations et les instructions (exécutables et non exécutables)
– Cette mesure est désignée par NCLOC (*Non-Comment Lines Of Code)*

– Le nombre de commentaires peut être mesuré par le nombre de lignes de commentaires (CLOC).
– La taille physique totale est donc LOC = NCLOC + CLOC
– On peut ainsi mesurer la densité de commentaires par CLOC/LOC
– LOC peut être plus pratique à utiliser que NCLOC
• Elle est plus facile à extraire
• Elle renseigne sur le nombre de pages pour imprimer le programme, l'espace nécessaire pour le stockage et la gestion de la visualisation à l'écran

– Le nombre d'octets nécessaires au stockage du programme
• Mesure indépendante du type de langage
• Facile à extraire

– Le nombre de caractères CHAR est une autre mesure de la taille physique

- Facile à extraire
- Peut être une base pour le calcule de LOC LOC = CHAR/**a**
  où **a** est le nombre moyen de caractères par ligne de texte

– Le nombre d'instructions livrées (DSI)
- Mesure dépendante du langage de Programmation

b. Spécification et conception

– Il est très difficile de trouver une mesure similaire à LOC ou CHAR
– La nature des documents est très variée et il est difficile de déterminer les types d'objets atomiques à compter (comme les caractères ou les lignes dans le cas du code)
– Les documents sont composés de texte et de diagrammes

– Dans l'industrie, une mesure utilisée est le nombre de pages
- Avantage : uniformise le texte et les diagrammes
- Inconvénient : comment gérer les pages contenant du texte et des diagrammes

– Une idée serait de mesurer le texte et trouver une unité de mesure pour les diagrammes

– De Marco propose une méthode de développement du logiciel mettant en oeuvre 3 vues. Cette méthode permet de bien identifier les objets atomiques.
– Aucune mesure de taille physique n'est cependant Définie

Vue	Diagramme	Objets atomiques
Fonctionnelle	Flot de données Dictionnaire de données	Ellipses Éléments de données
Données	Entité/Relation	Entités, relations
États	État/Transition	États, transition

Prédiction

– Il est très utile de prédire la taille physique du code (LOC) très tôt dans le cycle de développement

$$\alpha = \frac{taille\_code}{taille\_conception}$$

– Soit α le ratio d'expansion obtenu à partir de données historiques
– Soit *Si* la taille de conception d'un module *i*

$$LOC = \alpha \sum_{i=1}^{m} S_i$$

– α dépend de chaque organisation
– Une autre façon de prédire LOC a été proposée par Walston & Felix

$$D = 49L^{1.01}$$

– *D* est la taille de la documentation en pages et L la taille du code en milliers de LOC

3. Taille fonctionnelle: fonctionnalité

Intuitivement, c'est le nombre de fonctions fournies dans le logiciel. Il existe deux travaux reconnus pour la mesure de la fonctionnalité.

- Les points de fonctions (Albrecht)
- Les métriques BANG (De Marco)

L'objectif de ces deux travaux est de produire des mesures de taille qui permettent de bien prédire le coût et l'effort de développement

3.1 Points de fonctions

– L'objectif est de mesurer la quantité de fonctionnalité offerte par un logiciel à partir de sa spécification
– Pour calculer le nombre de points de fonctions FP *(function points),* il faut d'abord calculer le compte de fonctions non ajusté UFC *(unadjusted function count)*

– Pour calculer UFC, il faut déterminer à partir d'une certaine représentation du logiciel, le nombre d'éléments des types suivants
• **Entrées externes** : *éléments de données fournis par l'utilisateur (tels que des noms de fichiers et des sélections de menus). Ceci n'inclut pas les requêtes de l'utilisateur*
• **Sorties externes** : *éléments de données fournis à l'utilisateur (tels que les états et les messages)*
• **Requêtes externes** : *entrées interactives exigeant une réponse*
• **Fichiers externes** *: interfaces compréhensibles par une machine avec d'autres systèmes*
• **Fichiers internes** : *Fichiers principaux logiques dans le Système*

Exemple (vérificateur d'orthographe)

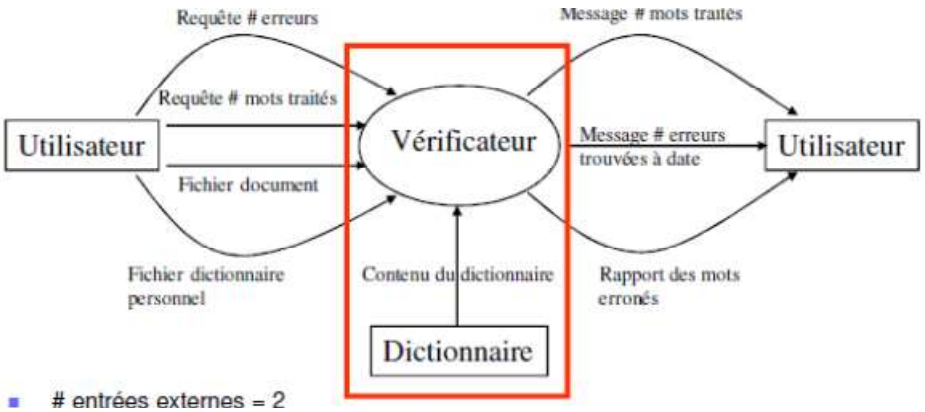
– **Spécification** :

• Le vérificateur accepte en entrée un fichier document et, optionnellement, un fichier dictionnaire personnel

• Le vérificateur affiche tous les mots qui ne sont ni dans le fichier dictionnaire, ni dans le fichier dictionnaire personnel

• L'utilisateur peut à tout moment demander le nombre de mots traités et le nombre d'erreurs trouvées durant le processus de vérification

Points de fonctions



- # entrées externes = 2
- # sorties externes = 3
- # requêtes externes = 2
- # fichiers externes = 2
- # fichiers internes = 1

– Complexité subjective

- Échelle ordinale
- Valeurs : simple, moyen et complexe

– Les entreprises ont développé des heuristiques pour attribuer une valeur à un type d'éléments

– Facteur de pondération (fpond)

- Un coefficient qui correspond à une combinaison (type d'éléments – complexité subjective)

– Calcul de UFC

Exemple (Vérificateur)

• Quelle est la valeur de UFC si tous les éléments sont de complexité moyenne?
• Quelle est la valeur de UFC si le fichier dictionnaire était complexe

Types d'éléments	Facteur de pondération		
	simple	Moyen	complexe
Entrées externes	3	4	6
Sorties externes	4	5	7
Requêtes externes	3	4	6
Fichiers externes	7	10	15
Fichiers internes	5	7	10

– Calcul du nombre de points de fonction FP $FP = UFC * TFC$

– Facteur de complexité technique (TFC)
• C'est un facteur d'ajustement de UFC dans une plage de ±35% (0.65 <=TFC<=1.35)
• Facteurs influençant la complexité

F1 Back-up fiable et récupération	F2 Communication de données
F3 Fonctions distribuées	F4 Exigences de performances
F5 Configuration très utilisée	F6 Saisie de données en ligne
F7 Facilité d'opération	F8 Mise à jour en ligne
F9 Interface complexe	F10 Traitements complexes
F11 Facilité de Réutilisation	F12 Facilité d'installation
F13 Sites multiples	F14 Facilité de changement

– Facteur de complexité technique
• Calcul de TFC

Pour chaque facteur on attribue une valeur dans une échelle ordinale (0,1,2,3, 4,5) où 0 veut dire que le facteur est non pertinent et 5 veut dire essentiel

$$TFC = 0.65 + 0.01 \sum_{i=1}^{14} Fi$$

(=0.65 si tous les Fi=0 et 1.35 si tous les Fi=5)

3.2 Les métriques BANG (De Marco)

– Deux métriques Bang selon le type du système
    • Fonctions
    • Données
– La métrique Bang « fonctions »
    • Consiste à compter le nombre de processus de bas niveau dans le DFD
    • Ce comptage de base est pondéré en fonction du type de processus et du nombre de noeuds « données » utilisés par le processus
- La métrique Bang « données »
    • Consiste à compter le nombre d'entités dans le diagramme Entités/Relations.
    • Ce comptage de base est pondéré en fonction du nombre de relations impliquant chaque entité

4. Mesure de la complexité

4.1. Rappel du graphe de flot de contrôle

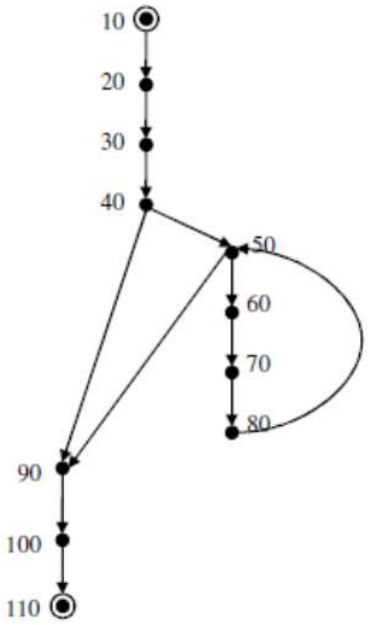
De nombreux travaux sur les métriques ont porté sur la structure du flot de contrôle
- Graphe de flot de contrôle
- Graphe orienté

• noeud = instruction du programme
• Arc = le flot de contrôle entre une instruction et une autre

Autres concepts
- Degré_entrée d'un noeud = nombre d'arcs incidents
- Degré_sortie d'un noeud = nombre d'arcs sortants
- Noeud procédure = un noeud de degré_sortie 1
- Noeud prédicat = un noeud de degré_sortie > 1
- Chemin = une séquence d'arcs consécutifs
- Chemin simple = une séquence d'arcs consécutifs sans Répétition

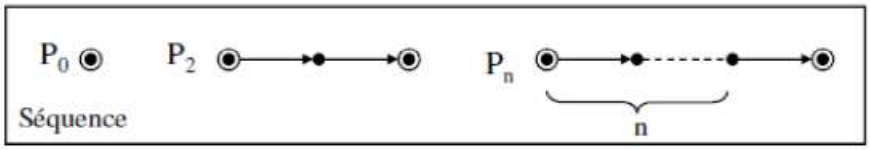
Exemple de graphe de flot de contrôle

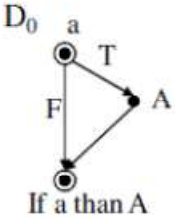
```
10 lire p
20 lire e
30 Cal := 1
40 Si e ~= 0 alors
50 Tant que e > 0 faire
60 cal := cal * p
70 e := e - 1
80 Fin_faire
90 Fin_si
100 écrire cal
110 fin
```

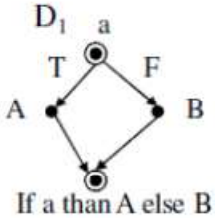


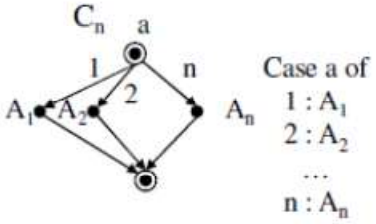
Structure du flot de contrôle

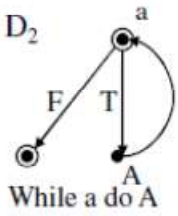
_ Patrons de base

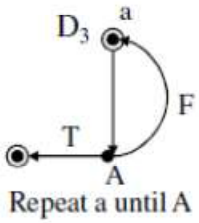




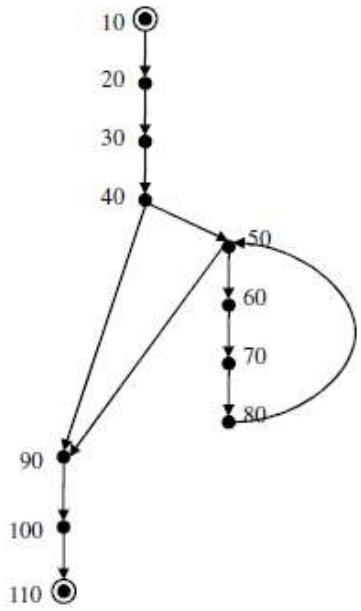


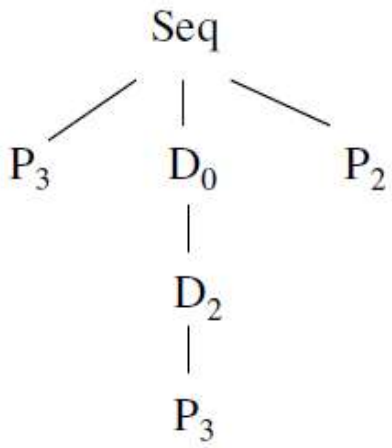






Arbre de décomposition





4.2. Mesure de la complexité McCABE

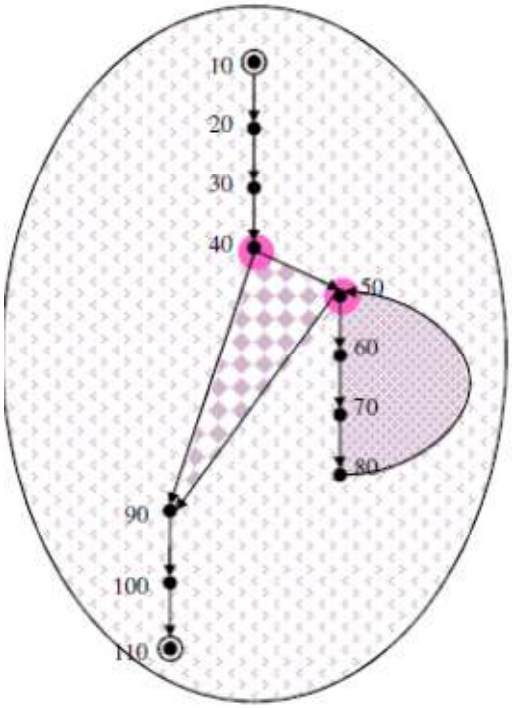
Complexité cyclomatique de McCabe

- C'est un modèle mathématique de mesure.
  - McCabe voit le programme comme un graphe dirigé.
  - Les instructions sont représentées par des nœuds.
  - Le flux de contrôle entre les instructions est représenté par des arcs.
  - C'est un nombre cyclomatique constitué par le nombre de chemins linéairement indépendant à travers un programme.
- Cette valeur est calculée par la formule suivante:

– Pour un graphe de flot de contrôle F comportant n noeuds (dont d nœuds prédicats) et e arcs, il existe trois façons de la mesurer

- $V(F) = e - n + 2$
- $V(F) = 1 + d$
- $V(F) = r$ (r est le nombre de régions de F)

Exemple



$$v(F) = 12 - 11 + 2$$
$$= 1 + 2$$
$$= 3$$

- Cette mesure est utilisée comme un indicateur de la complexité psychologique d'un programme.
  - Durant la maintenance, un programme ayant un nombre cyclomatique élevé (Cv >10) est considéré très complexe.
  - Cette valeur Cv peut être utilisée pour estimer le temps nécessaire pour comprendre et modifier un programme.

Le graphe de flux généré peut être utilisé pour identifier les chemins de tests possible durant la phase de test.

5. Couplage

Définition

– C'est le degré d'interdépendance entre modules

_ Types de couplage entre deux modules x et y (du moins mauvais au plus mauvais)

– Pas de couplage (R0) : x et y sont indépendant

– Couplage de données (R1) : x et y communiquent avec des paramètres

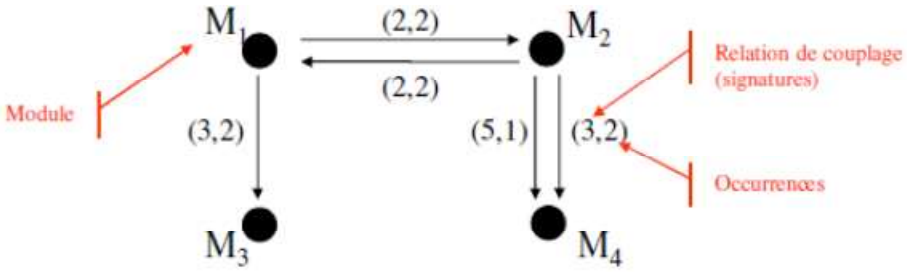
– Couplage de signature (R2) : x et y acceptent le même type d'enregistrement comme paramètre

– Couplage de contrôle (R3) : x passe un paramètre à y qui influence le contrôle

– Couplage par variables globales communes (R4) : x et y accèdent à la même donnée globale

– Couplage de contenu (R5) : x modifie des données ou des instructions à l'intérieur de y

Graphe de couplage



Mesure de couplage

i est la pire relation de couplage et n le nombre d'interconnexions entre x et y

Exemple

$$c(M_2, M_4) = 5 + \frac{3}{3+1} = 5{,}75$$

Mesure de couplage

– c(x,y) est de type ordinal

– Soit un système S comportant les modules M1, …, Mm, comment définir une mesure globale du couplage C(S) ?

C (S) est la valeur médiane de l'ensembl e

$$\{c(M_i, M_j) : 1 \le i \le j \le m\}$$

– D'autres mesures de couplage pour un module x existent.

Par exemple : le nombre maximum, le nombre moyen et le nombre total d'interconnexions

6. Cohésion

– La cohésion d'un module est le degré de participation des composants à la même tâche. Nous distinguons plusieurs types de cohésion (du plus désirable au moins désirable)

- **Fonctionnelle** : le module effectue une seule fonction bien définie.
- **Séquentielle**: le module effectue plus d'une fonction, mais dans l'ordre défini par la spécification

- **Communicationnelle:** le module effectue plusieurs fonctions toutes sur le même ensemble de données
- **Procédurale :** le module effectue plusieurs fonctions appartenant au même processus
- **Temporelle:** le module effectue plusieurs fonctions intervenant toutes dans le même laps de temps.
- **Logique**: le module effectue plusieurs fonctions liées logiquement
- **Cohésion par coïncidence:** le module effectue plusieurs fonctions sans aucune liaison entre elles

Un module peut avoir plusieurs types de cohésion
_ On caractérise un module par sa moins désirable cohésion
_ La mesure de cohésion est de type ordinal

_ La cohésion d'un système peut être mesurée comme suit

$$\text{Ratio de cohésion} = \frac{\text{nb de modules ayant une cohésion fonctionnelle}}{\text{nb total de module}}$$