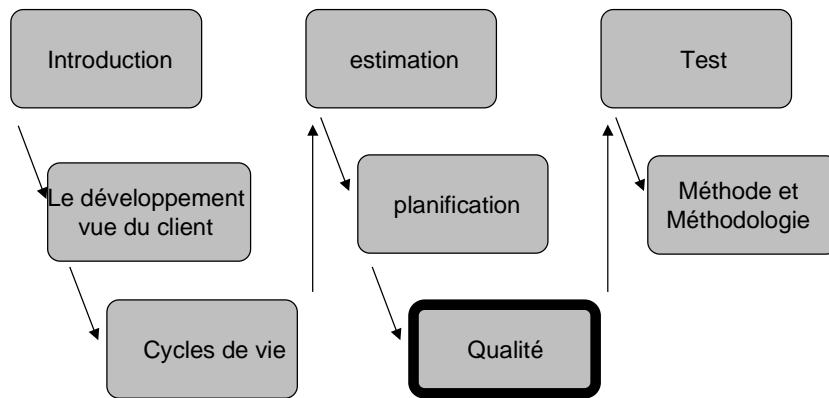


Introduction au génie logiciel # 3

plan



Qualité

-
- **Introduction**
– vocabulaire, normes, processus
- Facteurs, Critères, Métriques
– facteurs AFCIQ, compatibilité, critères, relation Facteur-critère, métrique
- Démarche qualité
– manuel et plan qualité, contrôle, évaluation

Introduction au génie logiciel # 3

Définition

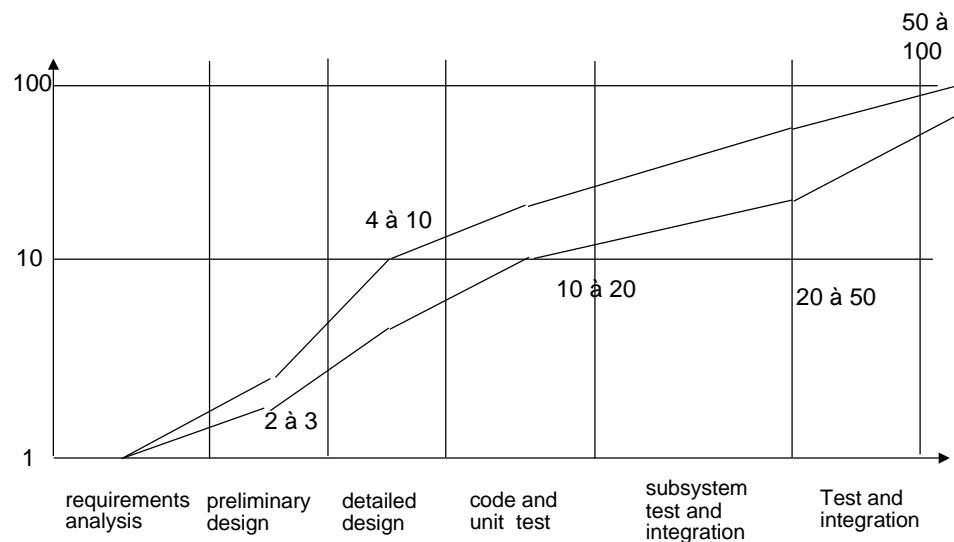
Aptitude d'un produit ou d'un service à satisfaire les besoins des utilisateurs.

En termes de fonctionnalités, délais, coûts.

Non qualité :

- Les défauts apparaissent lors de l'exploitation du logiciel
- coût de correction élevé

Coût de correction des erreurs



Introduction au génie logiciel # 3

Définitions

Assurance qualité : Mise en œuvre d'un ensemble approprié de dispositions préétablies et systématiques destinées à donner confiance en l'obtention d'une qualité requise.

Manuel qualité : Document décrivant les dispositions générales prises par l'entreprise pour obtenir la qualité de ses produits ou de ses services.

Plan qualité logiciel : Document décrivant les dispositions spécifiques prises par une entreprise pour obtenir la qualité du produit ou du service considéré.

Vocabulaire

Clauses qualité : expression des exigences (contractuelles ou non)

Logiciel : Ensemble des programmes, procédés et règles et éventuellement de la documentation, relatifs au fonctionnement d'un ensemble de traitement de l'information (arrêté du 22 décembre 1981).

Produit : Programmes sources et machines, des procédures et des ensembles de données enregistrées.

Plan de développement : Document décrivant pour une réalisation donnée, la décomposition en produits et en fournitures, les moyens à mettre en œuvre, les tâches nécessaires à la réalisation et les délais à respecter.

Client et Fournisseur : Le client commande un logiciel, le fournisseur le réalise.

Introduction au génie logiciel # 3

Plan de développement

- la description du logiciel à réaliser en différents niveaux de produits (programmes et documents).
- les moyens matériels et/ou logiciel à mettre à disposition ou à réaliser (Méthodes, Techniques, Outils).
- le découpage du cycle de vie en phases, la définition des tâches à effectuer dans chaque phase et l'identification des responsables associés.
- les supports de suivi de l'avancement (Planning et calendriers).
- les moyens utilisés pour gérer le projet.
- les points clés avec ou sans intervention du client.

Organismes de normalisation

AFNOR

- Recommandation de Plan qualité logiciel Z67-130
- Guide de rédaction de Plan qualité logiciel Z67-130
- Gérer et assurer la qualité : document AFNOR
- Du bricolage à l'industrialisation : la qualité des logiciels, J-P Martin

DGA

- Méthodologie de développement des logiciels intégrés dans les systèmes militaires : GAM-T-17 version 2 (juin 88)

IEEE

- IEEE 730, 732 et 738
- Std 828-1983 : Standard for software test documentation
- Std 829-1983 : Standard for software configuration management plans

Introduction au génie logiciel # 3

Organismes de normalisation

ESA

- PSS : PSS01, PSS05 Assurance qualité logiciel

OTAN

- AQAP 13 et 14 (mai 1984)

DOD

- DoD-STD-2167A : Military standard-Defense system software development (2/88)

AFCIQ

- Recommandation de Plan assurance qualité logiciel (V0 du 23-03-89)
- Recommandation de Plan de développement logiciel (V1 du 17-06-88)

Qualité logiciel

La qualité d'un logiciel n'a pas de mesure objective, ni de définition formelle:

Perceptions différentes (par exemple, en fonction de la position dans l'organisation de l'entreprise)

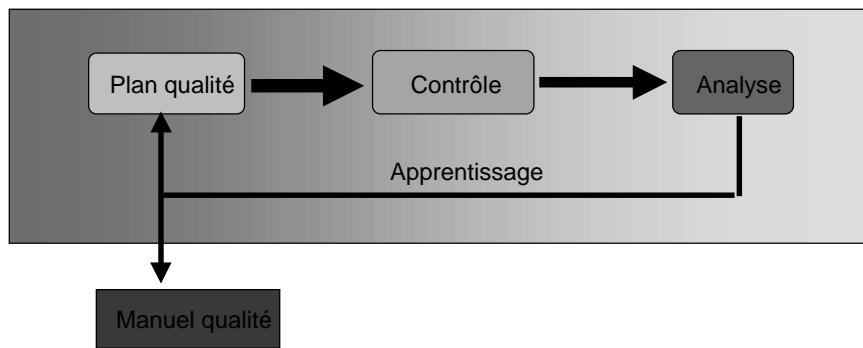
Quelques facteurs de qualité :

d'un Produit	d'un Service
Conformité	Efficacité
Portabilité	Disponibilité
Maintenabilité	Sécurité
Flexibilité	Fiabilité

Comment mesurer

Introduction au génie logiciel # 3

Le processus qualité



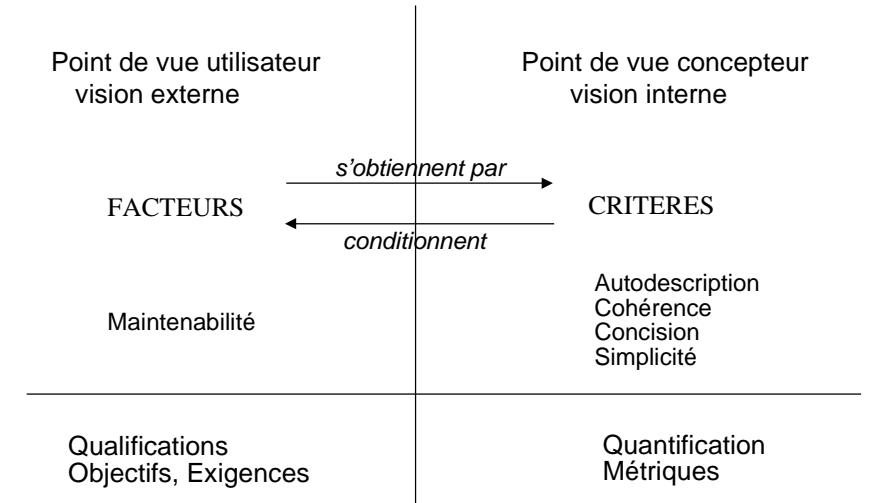
Qualité



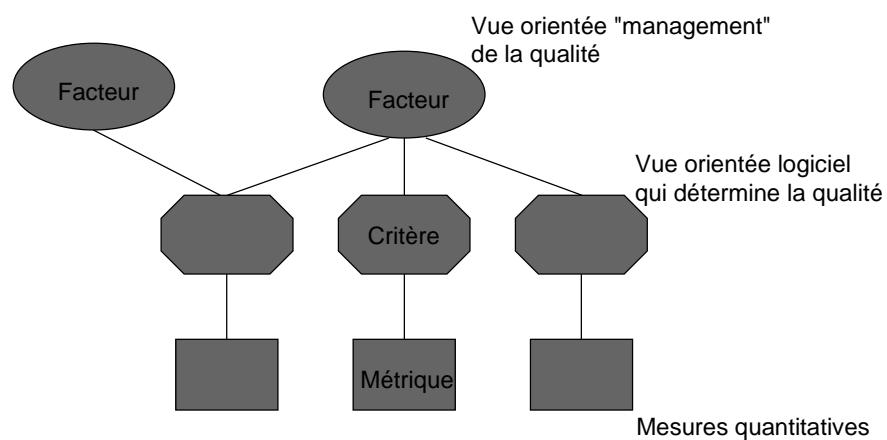
- Introduction
 - vocabulaire, normes, processus
- Facteurs, Critères, Métriques
 - facteurs AFCIQ, compatibilité, critères, relation Facteur-critère, métrique
- Démarche qualité
 - manuel et plan qualité, contrôle, évaluation

Introduction au génie logiciel # 3

Facteurs, critères

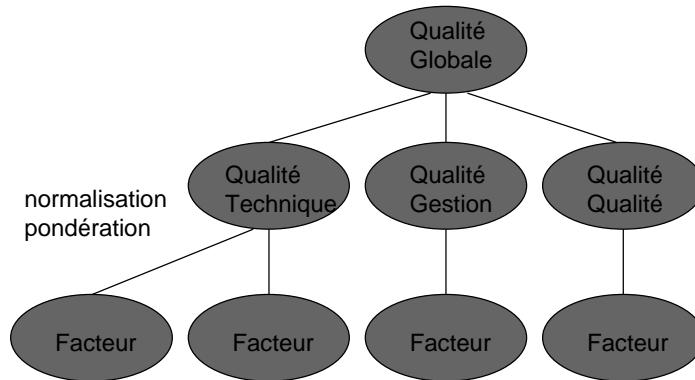


Métriques



Introduction au génie logiciel # 3

Mesure de qualité globale



Facteur qualité

caractéristique du logiciel qui contribue à sa qualité et possède les propriétés suivantes :

- orienté utilisateur
 - être relié à un coût par l'intermédiaire des activités qu'il engendre
- maintenabilité : effort pour localiser et corriger une anomalie

Introduction au génie logiciel # 3

Critère qualité

attribut du logiciel par l'intermédiaire duquel un facteur peut être évalué.

- Il est orienté réalisateur
- peut affecter plusieurs facteurs.

Facteurs (McCall, 1977)

– Correctness	Conformité
– Reliability	Robustesse
– Efficiency	Efficacité
– Usability	Maniabilité
– Integrity	Sécurité
– Maintainability	Maintenabilité
– Flexibility	Adaptabilité
– Testability	Testabilité
– Portability	Portabilité
– Reusability	Réutilisabilité
– Interoperability	Interopérabilité

Introduction au génie logiciel # 3

Définition des facteurs (1)

Facteur de qualité : aptitude du logiciel à	Note
Adaptabilité : minimiser l'effort nécessaire pour le modifier par suite d'évolution des spécifications	0 1 2 3 _____
Conformité : contenir un minimum d'erreurs, à satisfaire aux spécifications et à remplir ses missions dans les situations opérationnelles définies.	
Efficacité : se limiter à l'utilisation des ressources strictement nécessaires à l'accomplissement de ses fonctions.	
Maintenabilité : minimiser l'effort pour localiser et corriger les fautes.	

Définition des facteurs (2)

Facteur de qualité : aptitude du logiciel à	Note
Maniabilité : minimiser l'effort nécessaire pour l'apprentissage, la mise en œuvre des entrées et l'exploitation des sorties.	0 1 2 3 _____
Réutilisabilité : être partiellement ou totalement utilisé dans une autre application.	
Sécurité : surveiller, recenser, protéger et contrôler les accès au code et aux données ou fichiers.	
Robustesse : accomplir sans défaillance l'ensemble des fonctionnalités spécifiées, dans un environnement opérationnel de référence et pour une durée d'utilisation donnée.	

Introduction au génie logiciel # 3

Définition des facteurs (3)

Facteur de qualité : aptitude du logiciel à	Note
Testabilité : faciliter les procédures de test permettant de s'assurer de l'adéquation des fonctionnalités	0 1 2 3
Interopérabilité : s'interconnecter à d'autres systèmes.	
Portabilité : minimiser l'effort pour se faire transporter dans un autre environnement matériel et/ou logiciel.	

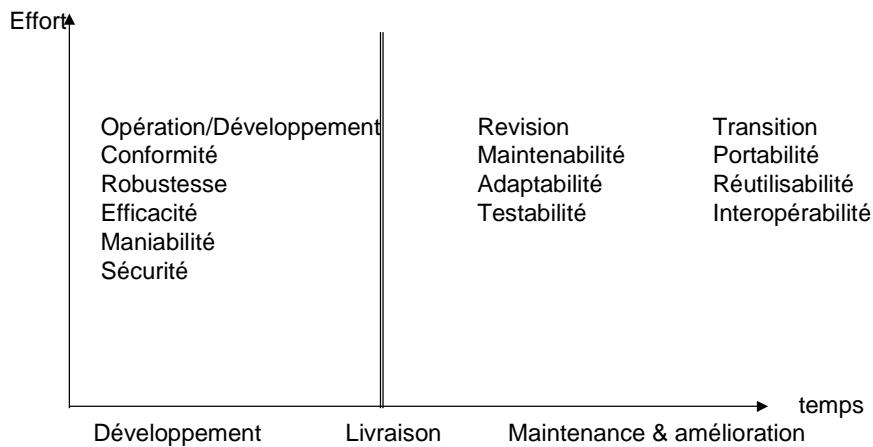
Qualification des facteur

Exigences

Facteur	Sous-rubrique	faible	moyenne	forte
Efficacité	occupation mémoire mémoire auxiliaire occupation lignes charge calcul % avec contr. durée	< 50% < 50% < 50% < 50% < 20%	>50% >50% >50% >50% <50%	>75% >75% >75% >75% >50%
Maniabilité	IHM utilisateur résultats formatés Aide en ligne	Non infor. Non Non	peu imp techn.	imp. public Oui Oui
Robustesse	reprise ap. coupure secteur protec. vs pannes contr. validité données Redondance	Non Non. Non Non	A froid Partielle	A chaud Oui Oui Oui

Introduction au génie logiciel # 3

Facteurs qualité & cycle de vie



Dépendances entre facteurs

	Co	Rb	Ef	Sé	Mn	Mt	Ts	Ad	Pt	Ru	In
Conformité											
Robustesse	+										
Efficacité											
Sécurité			-								
Maniabilité	+	+	-	+							
Maintenabilité	+	+	-		+						
Testabilité	+	+	-		+	+					
Adaptabilité	+	-	-	-	+	+	+				
Portabilité			-		+	+	+				
Réutilisabilité		-	-	-	+	+	+	+			
Interopérabilité			-					+			

Introduction au génie logiciel # 3

Critères et facteurs

- Conformité Traceabilité, consistance, complétude
- Robustesse Tolérance aux fautes, consistance, précision, simplicité
- Efficacité Efficacité d'exécution, de stockage
- Maniabilité Opérabilité, formation, communicativité, volume et taux d'entrées/sorties
- Sécurité Contrôle des accès, audit des accès
- Maintenabilité Consistance, simplicité, concision, modularité, auto-descriptivité

Critères et facteurs

- Adaptabilité Modularité, généralité, "expandability", auto-descriptivité
- Testabilité Simplicité, modularité, instrumentation, auto-descriptivité
- Portabilité Modularité, auto-descriptivité, indépendance matérielle et logicielle
- Réutilisabilité Généralité, modularité, auto-descriptivité, indépendance matérielle et logicielle
- Interopérabilité Modularité, "commonality" des communications et des données

Introduction au génie logiciel # 3

Eléments de mesure

Mesure directe et objective

- comptage de nombre de ligne de code source
- comptage de nombre d'homme-jours
- comptage du nombre d'abort système

Qualité du

produit
processus
service

Métriques obtenues par réponse oui/non (liste de contrôle)

- cohérence de la présentation des écrans
- respect de la procédure de signalisation des incidents
- capacité de raccordement satisfaisante

produit
processus
service

Métriques obtenues par enquête (note de 0 à 5)

- clarté de la présentation des résultats
- apport de l'assurance qualité
- disponibilité du système aux heures de pointe

produit
processus
service

Métriques techniques

- Métriques du code

- lignes de code, nombres d'opérandes, d'opérateurs
- complexité cyclomatique
- taux de commentaires

- Métriques de la spécification

- cohésion et couplage des modules
- taille et fréquence de communication de données

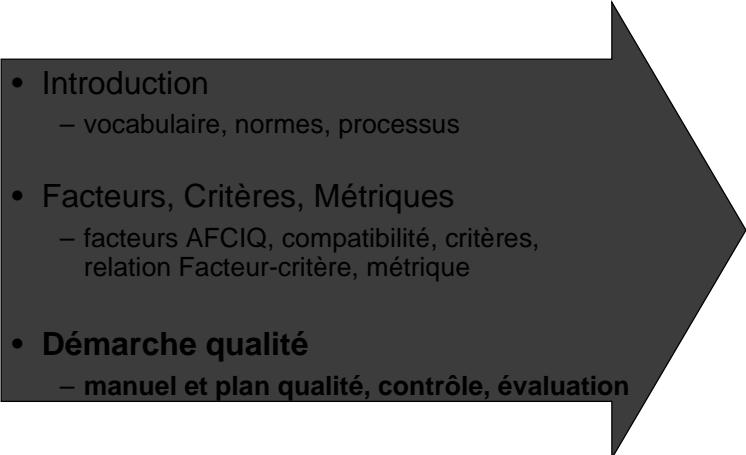
Introduction au génie logiciel # 3

Métriques autres

- Métriques du processus de gestion
 - mesure de la capacité à estimer
 - mesures liées à la documentation (taille, modularité, ...)
- Métriques du processus qualité
 - nombres de revues, d'inspection

Qualité



- 
- Introduction
 - vocabulaire, normes, processus
 - Facteurs, Critères, Métriques
 - facteurs AFCIQ, compatibilité, critères, relation Facteur-critère, métrique
 - Démarche qualité
 - manuel et plan qualité, contrôle, évaluation

Introduction au génie logiciel # 3

Activités de contrôle

Objectif mise en évidence de *non conformités*

Gestion

- Projet
- Modification
- Configuration



Contrôle Processus

- respect des modalités de déroulement (organisation et résultats) :
- des lectures croisées,
 - des tests
 - des activités de gestion
 - de la qualité

Techniques

- Analyse
- Conception
- Réalisation
- Test



Contrôle technique

- lecture simple ou croisée
- inspection
- test

Contrôle technique

Portée

- document de spécification
- code source

Modalités

- lecture simple, croisée, inspection
- test

Contrôle de fond

- contradiction, silence, omission, ambiguïté, ajout fonctionnel

Contrôle de forme

- redondance, bruit, sur-détail, normes non respectées

Introduction au génie logiciel # 3

Contrôle de processus

Portée

- procédure de gestion
- démarche technique

Modalités

- revue, audit

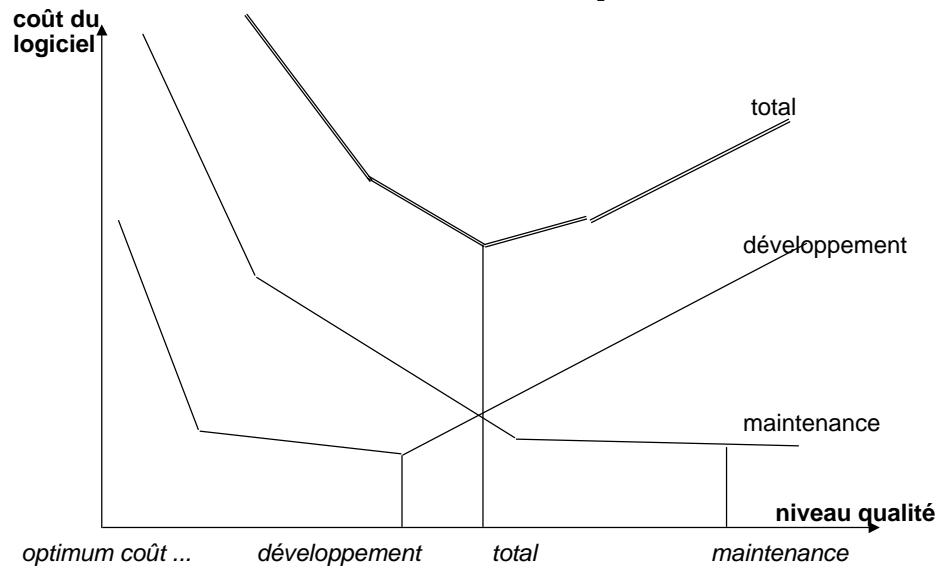
Contrôle de fond

- existence des processus, respect de la procédure, pertinence des tests

Contrôle de forme

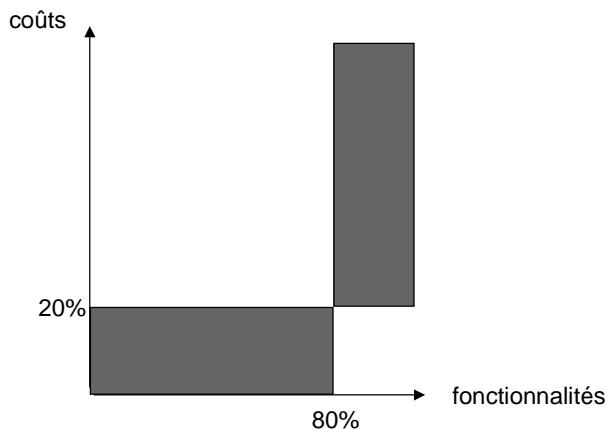
- conformité des contenus, conformité des circuits de validation

Coût de la démarche qualité



Introduction au génie logiciel # 3

La règle 80-20



Mise en place de la démarche qualité

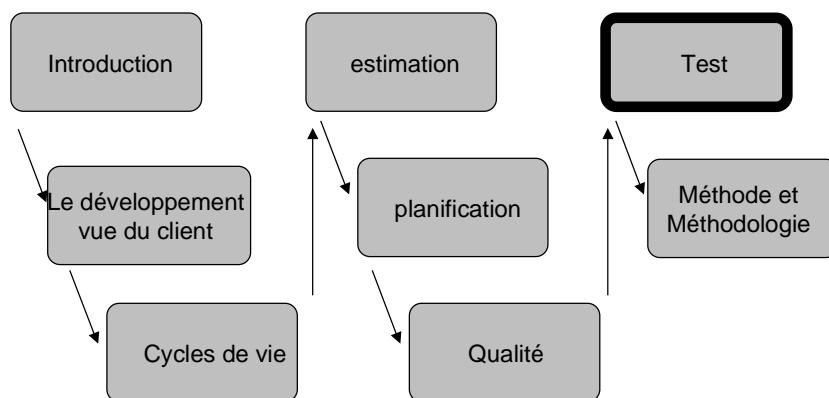
- Utilisation de techniques : Génie logiciel, contrôle
- Mise en place de méthodes : prototypage, ...
- Utilisation d'outils : spécification, simulation, gestion de projet
- Etablir, mettre à jour diffuser des références
- Formaliser la chaîne de production
- Définir des métriques adaptées à chaque activité ou produit
- Essayer la démarche sur des projets
- Contrôler la qualité et comparer
- Evaluer la démarche

Introduction au génie logiciel # 3

Bibliographie

- J.A. McCall, *Quality factors*, in Encyclopædia of Software Engineering, Vol 1, pp 958-969, John Wiley & Sons, 1994
- T. Forse, *Qualimétrie des systèmes complexes, mesure de la qualité du logiciel*, Les éditions d'organisation

plan



Introduction au génie logiciel # 3

Les catégories de test

Les tests peuvent être classés en fonction des critères suivants :

- les objectifs des tests
- la source de l'ensemble de tests
- le moment où les tests sont effectués

Objectifs des test

- détecter les déviations par rapport aux spécifications
- détecter des erreurs
- augmenter la confiance dans le programme
- déterminer un niveau de fiabilité dans le logiciel
- évaluer les performances
- évaluer le comportement en charge

Introduction au génie logiciel # 3

Source de l'ensemble de tests

- Spécification
 - Black box testing, tests fonctionnels
- Implantation
 - White box testing, tests structuraux
(couverture d'instructions, de branchements, de chemins)
- Faute
 - Essai de provoquer des fautes détectées dans des versions ou expériences précédentes (système, méthode, langage,etc)
- Utilisation
 - Jeu de tests réels

Moment où les tests sont effectués

- Test unitaires
(développement d'environnement de tests)
- Test de modules
- Test d'intégration
(détecter les problèmes d'interface)
- Test du système
 - alpha : système final prêt, utilisation interne
 - beta : utilisation chez des utilisateurs externes avertis
 - final

Introduction au génie logiciel # 3

Exemple

Retourne l'inverse de la racine carrée d'un nombre pour tout positif, sinon le nombre...

Quel jeu de test ?

```
function klouk(x: float): float
begin
    if x > 0 then
        return 1/sq(x);
    else
        return x;
    endif
end;
```

Le jeu de test {1, 0, -1} est falatieux ;
il fait croire que cette fonction est correcte

Fiabilité et tests aléatoires

- Jeux de tests aléatoires : N points
- Seuil de confiance : $1 - e = 95\%$ par exemple
- $\theta < 1 - (1-e)^{1/N}$
- $MTBF = 1/\theta$

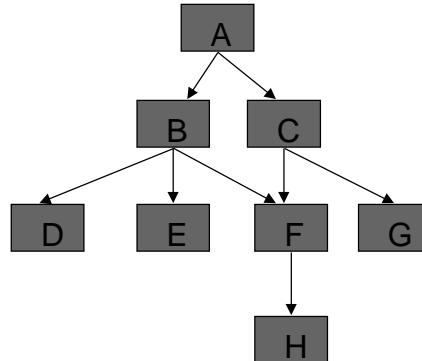
⇒ 1.000.000 points de tests pour 95% de confiance

⇒ MTBF de 220.000 exécutions

Introduction au génie logiciel # 3

Stratégies de test

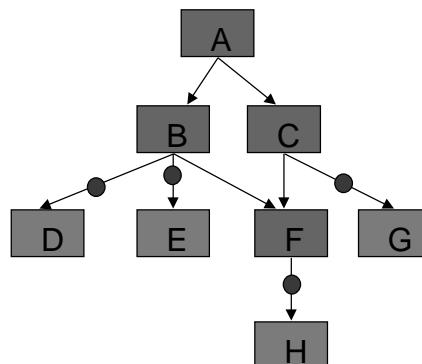
- Bottom-up testing
- Bottom-up testing
- Sandwich testing
- Build testing



Graphe d'appel

Bottom-up testing

- D, driver(D)
- E, driver(E)
- H, driver(H)
- G, driver(G)

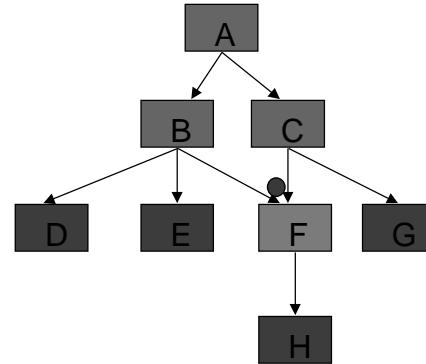


Graphe d'appel

Introduction au génie logiciel # 3

Bottom-up testing

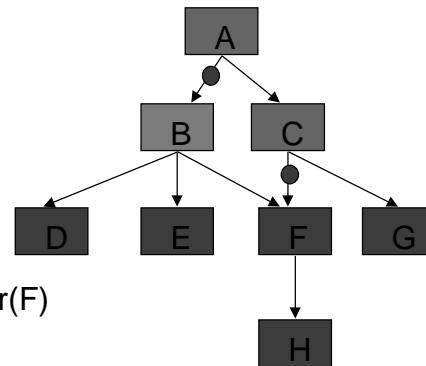
- D, driver(D)
- E, driver(E)
- H, driver(H)
- G, driver(G)
- H, F, driver(F)



Graphe d'appel

Bottom-up testing

- D, driver(D)
- E, driver(E)
- H, driver(H)
- G, driver(G)
- H, F, driver(F)
- D, E, F, B, driver(B), driver(F)

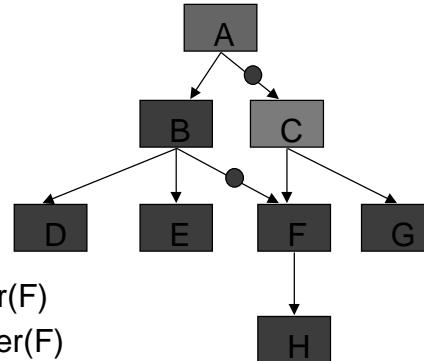


Graphe d'appel

Introduction au génie logiciel # 3

Bottom-up testing

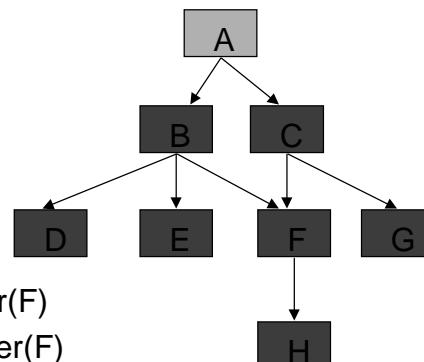
- D, driver(D)
- E, driver(E)
- H, driver(H)
- G, driver(G)
- H, F, driver(F)
- D, E, F, B, driver(B), driver(F)
- H, F, G, C, driver(C), driver(F)



Graphe d'appel

Bottom-up testing

- D, driver(D)
- E, driver(E)
- H, driver(H)
- G, driver(G)
- H, F, driver(F)
- D, E, F, B, driver(B), driver(F)
- H, F, G, C, driver(C), driver(F)
- D, E, H, G, F, B, C, A

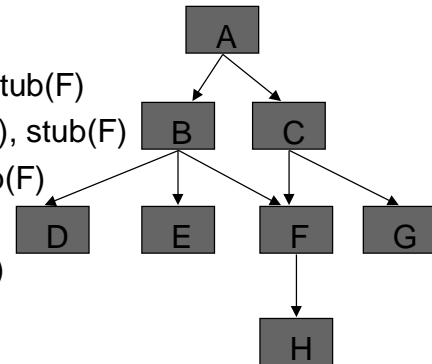


Graphe d'appel

Introduction au génie logiciel # 3

Top-down testing

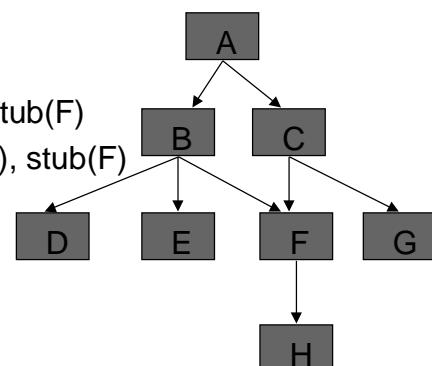
- A, stub(B), stub(C)
- A, B, stub(C), stub(D), stub(E), stub(F)
- A, B, C, stub(G), stub(D), stub(E), stub(F)
- A, B, C, D, stub(G), stub(E), stub(F)
- A, B, C, D, E, stub(G), stub(F)
- A, B, C, D, E, F, stub(G), stub(H)
- A, B, C, D, E, F, G, stub(H)
- A, B, C, D, E, F, G, H



Graphe d'appel

Sandwich testing

- A, stub(B), stub(C)
- A, B, stub(C), stub(D), stub(E), stub(F)
- A, B, C, stub(G), stub(D), stub(E), stub(F)
- D, driver(D)
- E, driver(E)
- H, driver(H)
- G, driver(G)
- H, F, driver(F)
- A, B, C, D, E, F, G, H

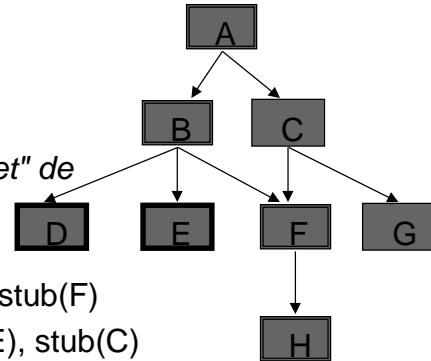


Graphe d'appel

Introduction au génie logiciel # 3

Build testing

- *Intégration top down par "paquet" de modules de criticité croissante.*
- A, stub(B), stub(C)
- A, B, stub(C), stub(D), stub(E), stub(F)
- A, B, F, stub(H), stub(D), stub(E), stub(C)
- A, B, F, H, stub(D), stub(E), stub(C)
- A, B, F, H, D, stub(E), stub(C)
- A, B, F, H, D, E, stub(C)
- A, B, F, H, D, E, C, stub(G)
- A, B, F, H, D, E, C, G



Graphe d'appel

Classes autodocumentées

- On exploite la caractéristique d 'encapsulation des classes :
cohérence entre données et services
- Chaque classe dispose de son jeu de tests...qui éventuellement lance un jeu de tests d 'autres classes

Appliqué à java, on peut définir un main() pour chaque classe.

Introduction au génie logiciel # 3

Bibliographie

- B. Beizer, *Software Testing Techniques*, 2nd ed, Van Nostrand Reinhold, New York, 1990
- R.A. DeMillo, R.J Lipton et F.G Sayward, *Hints on Test Data Selection: Help for Practicing Programmer*, Computer 11(4), 34-41, Apr 1978
- G. Myers, *The Art of Program Testing*, John Wiley & Sons, Inc, New York, 1979
- I. Sommerville, *Software Engineering*, 4th ed, Addison-Wesley Reading, Mass., 1992
- T.J. Ostrand, *Categories of Testing*, Encyclopædia of Software Engineering, J.J. Marciniak ed, John Wiley & Sons, Inc, New York, 1994
- R. Hamlet, *Test du logiciel & confiance*, Génie logiciel et systèmes experts, 30, mars 1993
- L.J. White et H.K.N. Leung, *Integration Testing*, Encyclopædia of Software Engineering, J.J. Marciniak ed, John Wiley & Sons, Inc, New York, 1994

Quelques liens

- Top 10 des « bugs » :
<http://www.cnet.com/Content/Features/Dlife/Bugs/ss05.html>
- Rapport de l'accident « Ariane V »
http://www.cnes.fr/actualites/news/rapport_501.html
- Histoires amusantes ou parfois inquiétantes...
<http://www.ozemail.com.au/~sphampel/Fun/Computer/famous.bug.txt>
ou
<http://www2.southwind.net/~rwweeks/bugs.html>