CENTRALE
L Y O N

UML-DL : Estimations et Tests

Estimation du coût de développement
Suivi de projet
Tests

CENTRALE
L Y O N

Bertrand DAVID : UML et Développement de Logiciels

Estimation du coût de développement

- **Technique usuelle :**
  - ➔ **Estimation de la taille du logiciel**
  - ➔ **Calcul de la durée de développement en tenant compte de la productivité et d'encadrement**

Technique plus fine :

- ➔ **Décomposition du logiciel lors de la phase de conception préliminaire en unités (modules)**
- ➔ **Estimation de la taille de chaque module**
- ➔ **Estimation de la durée de développement de chaque module en tenant compte de sa complexité**

Coût = Σ C Mi

- ➔ **Il faut prendre en compte tous les facteurs de productivité : expérience, nouveauté, complexité, conditions de travail**

CENTRALE LYON

Bertrand DAVID : UML et Développement de Logiciels

Estimation du coût de développement

- **Il faut établir un coût unitaire spécifique (pour 1000 lignes de code). Ce coût peut intégrer tous les aspects du projet :**
  - **➔Développement**
  - **➔Encadrement**
  - **➔Temps machine**
  - **➔Déplacements**
  - **➔Documentation**
  - **➔…**

TD/UMLDP/ET

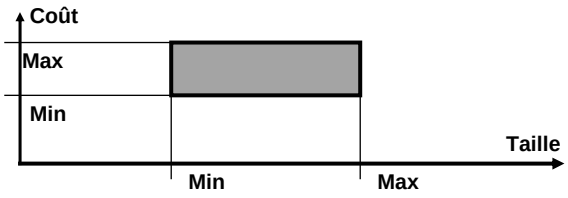
3

CENTRALE LYON

Bertrand DAVID : UML et Développement de Logiciels

Rectangle de probabilité de coût

- **Estimation de la taille par le calcul**
  - **➔ Du cas le plus favorable**
  - **➔ Du cas le plus défavorable**



- **➔ Affinement de l'estimation pendant le déroulement du projet**
- **➔ Elaboration des estimations pour chaque étape du projet :**
  - **Analyse**
  - **Conception**
  - **Test**
  - **…**

TD/UMLDP/ET

4

CENTRALE LYON

Bertrand DAVID : UML et Développement de Logiciels

Suivi du projet

- Etat d'avancement
- Allocation de ressources
- Revues
- Tests
  - ➔ Unitaires
  - ➔ d'intégration
  - ➔ de validation
  - ➔ d'acceptation

TD/UMLDP/ET

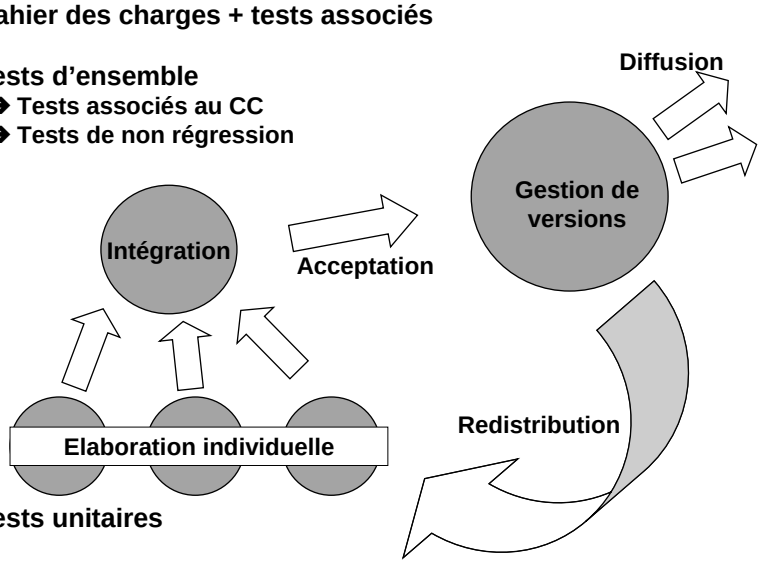
5

CENTRALE LYON

Bertrand DAVID : UML et Développement de Logiciels

Gestion des versions

- Cahier des charges + tests associés
- Tests d'ensemble
  - ➔ Tests associés au CC
  - ➔ Tests de non régression



- Tests unitaires

TD/UMLDP/ET

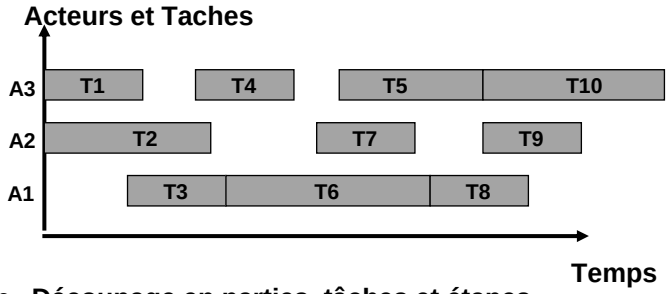
6

CENTRALE LYON

Bertrand DAVID : UML et Développement de Logiciels

Planning du projet

- Découpage du projet en étapes et tâches correspondantes, définition des jalons



- Découpage en parties, tâches et étapes
- Définition de dates de terminaison
- Définition de dépendances entre tâches
- Allocation de ressources

TD/UMLDP/ET

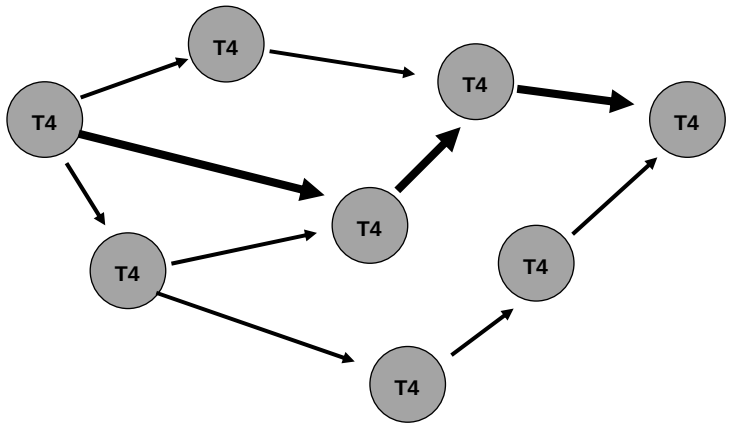
7

CENTRALE LYON

Bertrand DAVID : UML et Développement de Logiciels

Planning du projet

- PERT (chemin critique, date au plus tôt, au plus tard)
- GANTT (utilisation de ressources)



TD/UMLDP/ET

8

CENTRALE LYON

Bertrand DAVID : UML et Développement de Logiciels

Activités de test

- **Démarche globale**
  - **Préparation**
  - **Planification**
  - **Exécution**
  - **Analyse**
- **Situations**
  - **Environnements centralisés**
  - **Cycle en V pour des grands projets**
  - **Applications orientées objet**
  - **Applications client-serveur : Prototypage incrémental (RAD Rapid Application Development)**
  - **Applications Web - Intranet**
  - **Progiciels de gestion (ERP)**
  - **Applications multimédia**

TD/UMLDP/ET

9

CENTRALE LYON

Bertrand DAVID : UML et Développement de Logiciels

Catégories de tests

- **Boite blanche (développement et validation de performances)**
- **Boite noire (recette d'application)**
- **Boite grise (conversions)**

TD/UMLDP/ET

10

CENTRALE
L Y O N

Bertrand DAVID : UML et Développement de Logiciels

Boite blanche
(développement et validation de performances)

- **Créer des jeux d'essais validant le code de l'application**
  - **Tous les chemins ont été parcourus au moins une fois**
  - **Les conditions vrai/faux ont été validées**
  - **Les boucles sont exécutées correctement**
  - **Les valeurs limites sont correctement définies**
  - **Il n'y a pas de fuite mémoire**

TD/UMLDP/ET

11

CENTRALE
L Y O N

Bertrand DAVID : UML et Développement de Logiciels

Boite noire (recette d'application)

- **Tests boite noire et non-régression en phase de recette**
  - **Il s'agit de tests fonctionnels**
  - **Le respect des exigences**
  - **Détection de fonctions incorrectes ou manquantes**
  - **Les incohérences au niveau de l'interface**
  - **Les erreurs d 'accès aux données**
  - **Les problèmes de performance de l'application**
  - **Les conditions initiales ou finales d'une fonction**

➢ **L'intérêt principal des tests « boite noire » est qu'ils valident les fonctionnalités du logiciel plus que la qualité du développement. Ils sont donc plus proches des préoccupations quotidiennes des utilisateurs.**

TD/UMLDP/ET

12

CENTRALE LYON

Bertrand DAVID : UML et Développement de Logiciels

Boite grise (conversions)

- **Combinaison des tests « boite blanche » et « boite noire » dans le cas des conversions (an 2000, euro, …)**

Deux étapes :

- **Phase d'analyse : boite blanche - pour comprendre**
- **Phase de réalisation : boite noire - pour valider**

TD/UMLDP/ET

13

CENTRALE LYON

Bertrand DAVID : UML et Développement de Logiciels

Tests unitaires et tests d'intégration

- **Tests unitaires : composants individuels**
  - **Concepts spécifiques**
  - **Driver de test : logiciel spécifique définissant le cadre de test**
  - **Bouchon (stub) : module remplaçant les modules dépendant de l'unité qui doit être testée**
- **Trois approches :**
  - **Approche descendante (top/down)**
  - **Approche ascendante (bottom/up)**
  - **Approche modules isolés**

TD/UMLDP/ET

14

CENTRALE LYON

Bertrand DAVID : UML et Développement de Logiciels

Approche descendante (top/down)

- L'approche top/down consiste à tester chaque unité indépendamment de chaque unité appelante, en partant du haut de la hiérarchie. Chaque module appelé est remplacé par un bouchon.
- Avantages :
  - On effectue simultanément des tests unitaires et des tests d 'intégration
  - On valide dès le début les fonctions les plus importantes de l 'application, car ces tests sont basés sur les besoins fonctionnels
  - La structure de ces tests pourra être réutilisée dans les tests de non-régression
- Inconvénient
  - Utilisation de bouchons, dont la mise en œuvre peut être complexe.

TD/UMLDP/ET

15

CENTRALE LYON

Bertrand DAVID : UML et Développement de Logiciels

Approche ascendante (bottom/up)

- Les tests bottom/up sont basés sur l'approche inverse. On commence à tester les modules élémentaires et on remonte dans la hiérarchie
- On utilise pour cette méthode les drivers de test et non plus les bouchons
- Avantage :
  - Utilisation de fonctionnalités détaillées et utilisation des résultats des tests dans les niveaux supérieurs
- Inconvénients :
  - Les modules unitaires sont souvent développés plus tard que les niveaux supérieurs
  - Le contrôle sur les tests devient plus complexe
  - Les relations entre les modules peuvent devenir complexes dans des applications graphiques ou web.

TD/UMLDP/ET

16

CENTRALE LYON

Bertrand DAVID : UML et Développement de Logiciels

Approche modules isolés

- Le module est testé indépendamment des autres et on combine les deux approches top/down et bottom/up.
- On remplace ainsi chaque module appelant par un driver de test et chaque module appelé par un stub.
- Il est ainsi facile de tester chaque module, quels que soient la complexité de l'application et son impact sur les autres modules.
- En revanche, on perd la dynamique de l'application, et les tests d 'intégration devront être conçus et réalisés indépendamment des tests unitaires

TD/UMLDP/ET

17

CENTRALE LYON

Bertrand DAVID : UML et Développement de Logiciels

Tests de performance

- Temps de réponse dans les conditions normales
- Temps de réponse dans les conditions extrêmes
  - Volume
  - Conditions extrêmes d 'exploitation (charge utilisateur, charge réseau, manque de périphériques, …)
- Outils :
  - Simuler les utilisateurs virtuels
  - Enregistrement des scripts de simulation
  - Exécution des scénarios de simulation des utilisateurs

TD/UMLDP/ET

18

CENTRALE
L Y O N

Bertrand DAVID : UML et Développement de Logiciels

Ordonnancement des tests

- Planification des tests
- Estimation des ressources
- Approche modules isolés
- Tests unitaires
- Tests d ’intégration
- Tests système et tests fonctionnels
- Tests d installation
- Tests de recette
- Bêta tests

TD/UMLDP/ET

19