PROJET TUTORE

- **Unité d'enseignement**

  Projet tutoré
- **Année Académique**

  2012/2013
- **Parcours**

  Informatique de Gestion – Licence 3
- **Enseignant**

  Douwé Hallam Vincent (douwevincent@yahoo.fr)

Informations générales

- Nous aurons 90 heures pour vous permettre de réaliser vos différents projets
- Nous aurons des séances en salle de classe pour vous présenter des certains concepts
- Nous aurons aussi des séances en laboratoires qui vont correspondre aux travaux pratiques
- A la fin chaque groupe présentera son projet à un ensemble d'enseignants qui vont les évaluer.

Objectifs du cours

- Présenter les défis et techniques utilisés pour développer des logiciels de qualité.
- Permettre aux étudiants de mettre en pratique les concepts théoriques vu tout au long de leur formation.
- Introduire les étudiants aux problématiques liées au dévéloppement en équipe
- Développement en large
- Gestion et suivi d'un projet de développement
- Maitriser les techniques de POO et le langage java.

Plan

- Introduction
- Cycle de vie du logiciel
- La phase d'analyse et le cahier des charges
- La phase de conception et les modèles UML
- Mise en place de l'environnement de developpement
- Développement de logiciel en couches.
- Le développement dirigé par les tests.

Délivrables

Au long du projet, chaque groupe va produire :

- Un document d'analyse du projet
- Un document de conception
- Un plan de test (pour un autre groupe)
- Implémentation du projet
- Tester un autre projet.

Défis

Ce projet est particulièrement long et :

- Sera difficile mais sera aussi une expérience gratifiante
- Experer apprendre beaucoup de choses et défendre vos points de vue
- Il vous demandera probablement beaucoup de temps. Apprenez donc à très bien vous organiser

Bibliographie

- Tout cours sur le génie logiciel
- Maven ; the definitive guide
- JSF in action
- Jenkins : the definitive guide
- Itext
- ...

Les projets

Technologies

Les technologies suivantes sont imposées par le client :

- Langage de programmation : Java
- Framework Web : JSF
- Servers : Apache tomcat ou
- Système d'exploitation : Linux
- Base de données : MySQL

INTRODUCTION

Le monde du logiciel

Les logiciels sont omniprésents :

- Gouvernement
- Les chaînes de fabrications (usines)
- Les transports et l'éducation
- La santé, la défense, les finances …

Le talon d'achille : le coût, la qualité, le planning, les considérations politiques et opérationnelles (pas seulement techniques).

Le besoin de logiciels de qualité

- Beaucoup de vies et d'entreprises sont dépendants des logiciels de qualité.
- Nous avons besoin de logiciels plus complexes, plus larges et de meilleures qualité et qui peuvent être produit suivant des plannings prédictables.
- Pour atteindre ces objectifs, il faut suivre une méthodologie

Défis

Le développement de beaucoup de logiciels échouent :

- Des fonctionalités non réalistes
- Mauvaise gestion et leadership
- Manque de contrôle (aucun plan personnel des développeurs, connaissances insuffisante de la part du management)
- Qualité insuffisante
- Technologie insuffisante (méthodes, outils et langages)

Et le génie logiciel

La but du génie logiciel est production des logiciels opérationnels de qualité qui satisfont les besoins des utilisateurs.

On fait de la programmation mais on fait aussi beaucoup d'autres choses.

von Clausewitz « Le génie logiciel est la continuité de la programmation à travers d'autres méthodes »

Les composantes du génie logiciels

- Décrire
- Implémenter
- Évaluer
- Gérer
- Opérer

Besoins, Documentation des spécifications

Conception et programmation

Tests et autres techniques de validation et vérification

Plans, Délais, communication, revue ...

Deploiement et Installation

Démarche

Dans le génie logiciel, il existe au moins deux démarche :

Processus. Ici on met l'accent sur les plans, les délais, les documents, les fonctionalités, les spécifications, l'ordre des tâches et des engagements (Cascade, CMMI, RUP).

Agile : On met l'accent sur les itétations courtes, tests (au lieu des spécifications), l'implication continuelle du client, refus de s'engager concernant les délais et les fonctionalités, des pratiques spécifiques (XP, Lean programming).

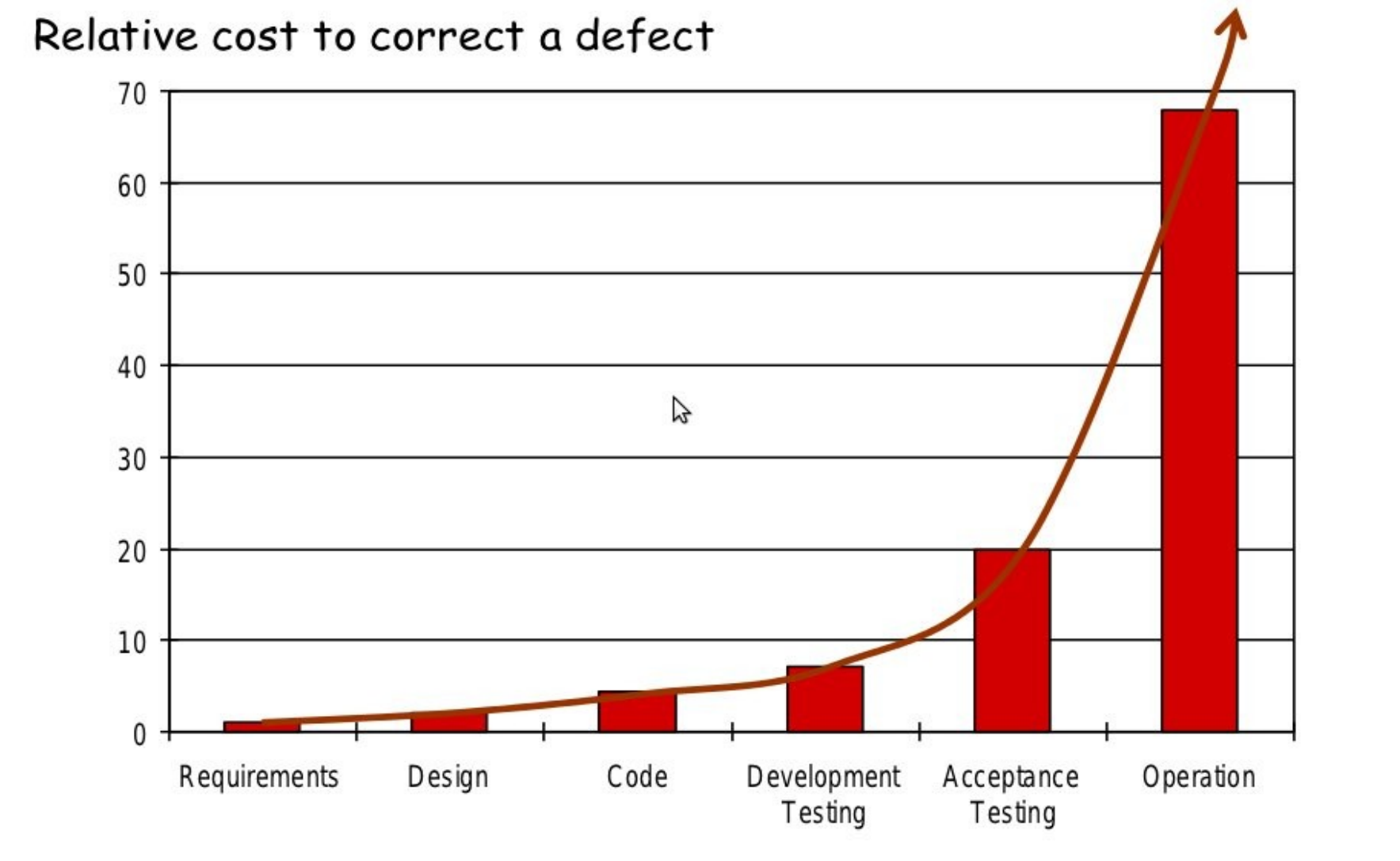
LA SPÉCIIFICATION DES BESOINS

Introduction

La phase de recueil des besoins est une phase crucial dans le développement du logiciel.

« *The hardest single part of building a software system is deciding precisely what to build. No other part of the conceptual work is as difficult as establishing the detailed technical requirements, including all the interfaces to people, to machines, and to other software systems. No other part of the work so cripples the resulting system if done wrong. No other part is more difficult to rectify later.* » Brooks, 87.

Introduction



Source : Boehm 81

Introduction

- 80% of interface fault and 20% of implementation faults due to requirements (Perry & Stieg, 1993)
- 48% to 67% of safety-related faults in NASA software systems due to misunderstood hardware interface specifications, of which 2/3rds are due to requirements (Lutz, 1993)
- 85% of defects due to requirements, of which: incorrect assumptions 49%, omitted requirements 29%, inconsistent requirements 13% (Young, 2001).

Définition

- Un besoin est une spécification qui décrit un comportement voulu pour un système.
- L'ensemble des besoins doivent être contenu dans un document appelé cahier des charges.

Les objectifs

- Comprendre les problèmes qui doivent être resolu par le logiciel
- Poser les questions pertinentes concernant le problème ou le système.
- Fournir une base pour répondre aux questions (surtout les spécificités)
- Définir les propriétés du problème et du système
- Décider ce que doit faire le système
- Décider ce que le système ne doit pas faire.
- Astreindre que le système va satisfaire les besoins des parties prenantes.
- Fournir une base pour le développement du système.
- Fournir une base pour la vérification et la validation du système.

Les produits

Après la phase d'analyser les besoins, les documents suivants doivent être produits :

- Le cahier des charges qui recapitulent tous les besoins
- Le plan de développement
- Le plan de test et validation.

Conseils pratiques

- Le document de validation et de vérification doit être produit avant le développement du produit
- Il faut identifier toutes les parties prenantes du du projet le plus tot possible.
- Il ne faut par sous-estimer le potentiel des mathématiques
- Préferer un langage précis et non falsiable à de banales généramités.
- Distinguer les spécifications de la machine des propriétés du domaines
- Utiliser la structure recommandé par le standard IEEE
- Écrire un glossaire
- Appliquer les cas d'utilisation pour dériver le plan de test et non les besoins.

Les types de besoins

- Fonctionnel
- Procédural
- Informel

vs

- Non fonctionnel
- Orienté objet
- Formel

- Textuel
- Exécutable

- Graphique
- Non exécutable

Les composantes

- Le domaine des propriétés
- Les besoins fonctionnels
- Les besoins non fonctionnels (fiabilité, sécurité,
- La précision des resultats, la performance concernant le temps et l'espace mémoire, la portabilité)
- Les besoins concernant le processus de développement et d'évolution du produit.

Les différents objectifs

- Justified
- Correct
- Complete
- Consistent
- Unambiguous
- Feasible
- Abstract
- Traceable
- Delimited
- Interfaced
- Readable
- Modifiable
- Verifiable
- Prioritized*
- Endorsed

Les difficultés

- Les langages naturels et leurs imprécisions
- Les techniques formelles et leurs abstraction
- Les utilisateurs et leur tendance à décrire vaguement leurs besoins.
- Les clients et leurs demandes qui changent toutes les fois
- Le reste du monde et sa complexité.

Deux travers constants

- Se retrouver entrain de prendre des décisions d'implémentation lors de la phase d'analyse. (overspecification)
- Oublier certaines parties du problème (underspecification)

Vérifiable vs non vérifiable

- Non vérifiable :
  - Le système doit fonctionner correctement
  - L'interface doit être conviviale
  - Le système doit répondre en temps réel
- Vérifiable:
  - La sortie doit être produite en moins de 30 secondes. Elles doivent être produites en moins de 1à secondes dans 80% des cas
  - Les apprennants doivent atteindre le niveau 1 de précision après deux jours d'apprentissage

Complete

Déduit du standard IEEE

« An SRS is complete if, and only if, it includes the following elements: All significant requirements, whether relating to functionality, performance, design constraints, attributes, or external interfaces. In particular any external requirements imposed by a system specification should be acknowledged and treated. Definition of the responses of the software to all realizable classes of input data in all realizable classes of situations. Note that it is important to specify the responses to both valid and invalid input values. Full labels and references to all figures, tables, and diagrams in the SRS and definition of all terms and units of measure. »

Les standards

Les standards dans le domaine de l'ingénierie servent à:

- Définir des pratiques communes
- De guide aux nouveaux ingénieurs.
- Rendre les processus de développment des logiciels comparables.
- Rendre possible la certification.

Le standard IEEE 830

- Il s'agit des pratiques recommandées par IEEE pour la spécification des besoins.
- Approuvé le 25 Juin 1998
- Décrit le contenu et la qualité d'une bonn spécification des besoins.
- But: “La spécification des besoins doit être correct, non ambiguë, complète, consistent, stable, vérifiable, modifiable et traçable

Le standard IEEE 830

- Contract: A legally binding document agreed upon by the customer and supplier. This includes the technical and organizational requirements, cost, and schedule for a
- product. A contract may also contain informal but useful information such as the commitments or expectations of the parties involved.
- Customer: The person, or persons, who pay for the product and usually (but not necessarily) decide the requirements. In the context of this recommended practice the customer and the supplier may be members of the same organization.

Le standard IEEE

- Supplier: The person, or persons, who produce a product for a customer. In the context of this recommended practice, the customer and the supplier may be members of the same organization.
- User: The person, or persons, who operate or interact directly with the product. Th user(s) and the customer(s) are often not the same person(s).

Le standard IEEE

Quelques problèmes à résoudre lors de la spécification des besoins

- Les fonctionalités
- Les interfaces externes
- La performance
- Les attributes
- Les constraintes de conception imposées par une implémentation

Le standard IEEE

La structure du document recommandé :

- 1. Introduction
  - 1.1 Purpose
  - 1.2 Scope
  - 1.3 Definitions, acronyms, and abbreviations (glossaire)
  - 1.4 References
  - 1.5 Overview
- 2. Overall description
  - 2.1 Product perspective
  - 2.2 Product functions
  - 2.3 User characteristics
  - 2.4 Constraints
  - 2.5 Assumptions and dependencies
- 3. Specific requirements
- Appendixes
- Index

Exemple de scope

Identifie le produit logiciel qui sera produit par sont nom (e.g., Host DBMS, Report Generator, etc.)

- Expliquer ce que fera et ne fera pas le produit*
- Decrire l'application du logiciel: buts et avantages
- Établir une relation avec des besoins plus généraux s'il y en a

Product perspectives

Décrire la relation avec les autres produits s'elle existe.

Exemples:

- Les interfaces du systèmes
- Les interfaces utilisateurs
- Les interfaces matérielles
- Les interfaces logicielles
- Les interfaces de communication
- Mémoire
- Opérations
- Les bésoins concernant le site d'implémentation

Contraintes

Décrire les propriétés qui peuvent limiter les options du dévéloppeur.

Exemples:

- Les dispositions légalesRegulatory policies
- Les limitations matérielles
- Les interfaces aux autres applications
- Les opérations parallèles
- Les fonctions d'audits
- Les fonctions de contrôle
- Les contraintes de fiabilité
- Criticité de l'application
- Les considérations de sécurité

Les besoins spécifiques

Dans cette section, l'on détaille les différentes fonctionalités pour les rendre utilisables par les concepteurs et les testers

Exemples:

- Détails concernant les interfaces externes
- Spécification précise de chaque fonction
- Répondre aux situations anormales
- Détailler les besoins en performance
- Les besoins en base de données
- Les contraintes de conception
- Les attributs spécifiques tels la fiabilité, la disponibilité, la sécurité et la portabilité.

Structure

3. Specific requirements

- 3.1 External interfaces
  - 3.1.1 User interfaces
  - 3.1.2 Hardware interfaces
  - 3.1.3 Software interfaces
  - 3.1.4 Communication interfaces
- 3.2 Functional requirements
- 3.3 Performance requirements
- 3.4 Design constraints
- 3.5 Quality requirements
- 3.6 Other requirements

LA CONCEPTION ET L'ARCHITECTURE

LA CONCEPTION ET L'ARCHITECTURE D'UN SYSTÈME

MISE EN PLACE DE L'ENVIRONNEMENT DE DEVELOPPEMENT

DEVELOPPEMENT DES LOGICIELS EN COUCHES

LE DEVELOPPEMENT DIRIGÉ PAR LES TESTS