Introduction au Génie Logiciel

Sommaire

I) Introduction

II) Approches de développements et Modèles de gestion de projets logiciels

Cours 1 :

I) Introductions

L'objectif principal de cette introduction est de faire comprendre à l'étudiant, les causes de l'apparition du GL et dans quelles circonstances cette discipline a connu sa naissance. Le cours commence par exposer la crise logicielle, ensuite passe à décrire l'apparition ou la naissance du génie logiciel comme une nouvelle discipline dans le monde de l'informatique.

Cette introduction peut être divisée en deux grandes parties : ***Crise Logicielle***, et ***Génie logiciel***.

1 Crise logicielle et première motivation du Génie Logiciel

Deux éléments principaux seront traités ici : une exposition de l'ère des années soixante et leurs effets dans le monde informatique et l'apparition de la crise logiciel: ses caractère, et ses causes et enfin les solutions qui ont été envisagées.

1.1. Ere des années soixante :

Pour un informaticiens, les années soixante sont caractérisées par deux caractères :

a) L'apparition des machines de la troisième génération : ces machines sont en principe équipées par des architectures à base de circuits intégrés et des microprocesseurs. Ces machines sont caractérisées par : une puissance de calcul importante, une capacité de stockage progressive, et des prix en baisse. L'ordinateur n'est plus la machine privée ou inaccessible.

b) L'apparition des langages de programmation de haut niveau : ces langages ont fait une rupture avec le langage assembleur ou binaire, où le programmeur était trop lié et obligé de connaitre des détails compliquées sur l'architecture de la machine. Plusieurs langages, dont quelques un sont toujours exploités où qui représentaient le noyau de ceux utilisé couramment, ont connu leur apparition dans cette période. Les plus connus sont : Fortran (1954), LISP(1958), COBOL (1959), ALGOL (58, 68, vers le PASCAL 71), APL(1962), SIMULA (1962-67), BASIC (64), PL/I(1964), smaltalk (1970), Prolog (1972), ML(1973),

Ces deux circonstances ont aidé l'informatique à avoir une expansion importante et elle a envahi la vie des humains et de leur société. Plusieurs domaines ont été touchés : militaire, sanitaire, logistique, éducatif, scientifique, spatiale, et même sociale… L'informatique est devenue la solution pour plusieurs problèmes. Les programmes et logiciels sont devenus des axes pertinents dans la vie quotidienne, et tout le monde en dépend.

Cependant, et rapidement on s'est rendu compte que les solutions rapporté par l'informatique à cette époque ne sont pas parfaites. On a remarqué rapidement que le matériel de l'informatique connaît une baisse dans son prix et le rend accessible pour tout le monde, alors que les logiciels qui pilotent ce matériel connaissent une augmentation importante dans leurs prix. Les logiciels produit ne posaient pas uniquement un problème dans leurs prix, mais les développeurs n'arrivaient plus ni à contrôler les dates de livraison de leurs logiciels, ni à assurer leurs fiabilités !!!!

On a commencé à parler d'une « **Crise Logicielle** ».

1.2. Crise logicielle : caractéristiques et échecs célèbre :

La crise logicielle est la situation dans laquelle, le matériel informatique connaissait une **baisse** dans son prix qui se heurte par une **augmentation dans les prix des logiciels**, un problème dans leur **fiabilité**, un **délai de livraison**, énormément, **dépassés** et un **coût** difficile à **estimer** et à **contrôler**. Plusieurs échecs ont été enregistrés qui prouvent une telle crise :

Problèmes de fiabilité :

1) 1962 : perte de mariner (première sonde américaine) : erreur de transcription
2) 1971 : perte de ballons durant une expérience de météo en France.
3) 1981 : retard du lancement
4) 1990 : problème d'un **switch** : pas d'électricité dans EU et Canada
5) 1994 le célèbre bug du pentium 470 $ de perte
6) 1996 explosion de Ariane 5 : 37 seconde après son lancement ; erreur de transformation de vecteur de bits
7) 2003 : blocage du système de télécom cote Est EU : mise à jour de l'application, et deadlock
8) 2004 : robot de mars ; bloqué (trop de fichier ouvert !!!). http://fr.wikipedia.org/wiki/Spirit_(rover)

Problèmes de délais et de coût :

1) le système d'exploitation d'IBM-360. Un projet prometteur, mais livré tardivement, un prix énorme, et des erreurs importantes, et trop de mémoire pour le stockage
2) projet PL/I 1968 : un projet de rêve jamais achevé !!!!

1.3. Crise logicielle : raison possibles :

Les raisons de cette crise peuvent être classées en deux types :

a) Des raisons dues aux développeurs : les développeurs souffrent de deux problèmes
   - Ils considèrent que la réalisation d'un logiciel est une tâche purement technique (programmation). Et donc, ils ne passent pas le temps nécessaire dans la phase d'analyse et de compréhension. Principe de programmation : Programmer →traquer les erreurs.
   - Les développeurs sot d'origine n'ont pas des compétences d'interaction, de communication, de pouvoir écouter, comprendre et convaincre les clients et les utilisateurs de leurs produits

b) Des raisons dues aux caractéristiques du produit logiciel : le logiciel est un produit spécifique qui lui rajoute d'autre problèmes :
   - **Imprévisible** : on ne peut pas facilement avoir une prévision de ce logiciel avant la fin de sa réalisation ;
   - **Flexible** : Un produit trop **souple**, d'où des modifications mineures dans le code source peuvent altérer radicalement son comportement !!!!
   - **Se propage par simple copie** : le **clonage** et la **duplication** de ce produit se fait par simple copie et non pas par une recréation.

1.4. Crise logicielle : solutions envisagées :

Les solutions proposées doivent, bien sur, traiter les deux sources du problème :

a) **Pour les développeurs** : on doit revoir la formation de ces développeurs. Ils doivent avoir conscience que développer un logiciel ce n'est pas une simple programmation, et donc ils doivent prendre leurs temps d'analyse, de compréhension, et de retarder le plus possible cette programmation. Les développeurs doivent avoir aussi d'autre compétences leurs permettent de communiquer et de comprendre les clients et leurs besoins.
b) **Pour le produit logiciel** : en général, l'intervention la plus importante peut prendre place sur la prévision du logiciel. L'idée est d'avoir des visions préalables, partielles sur ce logiciel avant même de sa réalisation. On parle de **prototype**, de **maquette**, de **modèle** … qui sont des représentations intermédiaires permettent de prédire la forme et le fonctionnement du futur logiciel.

   Ces propositions ont été prises en considération de manière plus claire, en proposant toute une nouvelle science dont l'objectif est de développer des logiciels, c'est le Génie Logiciel.

2 Naissance du Génie logiciel et ses objectifs primaires

2.1. Naissance

Le GL a connu sa naissance sous le nom de « «software engineering » dans la période **7-11/10/1968** durant une réunion d'un ensemble de spécialiste dans le développement et l'exploitation des logiciels, sous le parrainage de l'OTAN. Dans la ville de **Garmisch-Partenkirchen** (Allemagne). Une nouvelle science pour assurer la fiabilité des logiciels produits, contrôler leurs coûts et respecter leurs dates de livraison.

2.1. Définition du logiciel

Ensemble de programmes, procédures et des données manipulées par ces programmes pour faire fonctionner un ordinateur, ainsi que les documentations nécessaire pour le développement et la maintenance de ces programmes.

Logiciel=programmes+données+documentations

2.2. Définition du GL

Définition 1. L'art de spécifier, concevoir, réaliser et faire évoluer, dans des délais raisonnables et avec des moyens, des programmes des données et de documentations de qualités.

Définition 2. La science dont l'objectif est de mettre en place : des **outils**, **langages**, **méthodes**, **approches** favorisant la production de logiciels de qualité.

2.3. Production des Logiciels selon le GL : Processus Logiciel

Le GL considère que le logiciel est un produit de « manufacturing», qui ne se réalise pas d'un seul coup, mais qui se réalise en plusieurs étapes, et qui exige plusieurs personnes (une équipe). On parle

d'un **processus logiciel**, où le logiciel nait, évolue, et meurt (être rejeté) à une certaine phase. Par cela, l'objectif du GL se résumera encore dans deux points :

- La définition de ce processus logiciel
- La gestion de ce processus logiciel

Dans ces objectifs le GL propose ce qui est connue par **Approches de développement de logiciel**, et **Modèle de Gestion de projet Logiciel**.

a) Les approches de développement : une approche de développement est une démarche ou une politique (théorique) permettant de définir les étapes à suivre pour développer un logiciel. Il y'a une variété d'approches :
   - L'approche de la cascade
   - L'approche de prototypage
   - L'approche de la programmation exploratoire
   - L'approche de la réutilisabilité ;
   - L'approche des transformations formelles

b) Les modèles de gestion de projet logiciel : ici on met l'accent, en plus de la définition des activités, sur leurs gestion surtout. Par modèle, on veut dire une image abstraite d'un système ou d'une situation. Donc un modèle de gestion est une image qui décrit l'évolution de la construction d'un projet logiciel. On trouve par exemple :
   - Modèle de la cascade
   - Modèle des incréments
   - Modèle de la Spirale :

3 Conclusion

Le GL est la science proposée pour résoudre la crise logicielle, et assurer le développement de logiciels de qualité. Cette discipline doit établir un ensemble d'approches de développement, et de modèle de gestion gérant le processus logiciel. Du côté qualités requise, on peut citer les plus exigées comme suit :

- Fiabilité ;
- Efficacité : le logiciel doit utiliser rationnellement ses ressource : ne doit pas consommer trop de mémoire, ni gaspiller ses cycles d'horloge ;
- Interface appropriée : l'interface doit être facile à utiliser, et à apprendre ;
- Maintenabilité