Gestion de Projet S3

Waterfall

Objectif : organiser de manière scientifique la démarche permettant de produire des programmes. Beaucoup de paperasse, et pas mal de vérification en cours de développement. Cela se faisait en produisant énormément de rapports très détaillés. En plus de cela, il fallait énormémment planifier à l'avance le développement.
Enorme production de paperasse !! $\rightarrow$ beaucoup de temps perdu

Les désavantages

- Bureaucratie : Beaucoup de rapports à produire.
- Démotivation : trop de paperasses, les développeurs finissaient par se démotiver en cours de route à force de devoir rédiger des tonnes de paperasses.
- Retards : Comme les développeurs étaient démotivés, ils finissent toujours à accumuler du retard, et le projet finissait toujours en retard.
- Peu de valeur pour le client : Au final le produit fini ne convenait pas aux attentes du client.

Raisonnement

On receuille l'avis du client, on construit alors l'architecture sur ces avis, et puis une fois que l'architecture et la nature du programme sont fixés, on n'en rediscute plus.

On réfléchit alors sur cela à une solution pour coder le programme résolvant les besoins du client.

Une fois cela fait, on n'a plus qu'à implémenter tout ce qu'on a défini à l'avance. Cela permet de juste écrire tout ce qu'on a planifié avant sur le papier.

Et on fini par ne tester qu'une fois toutes les fonctionnalités. Pas de débbugage avant la toute fin du développement.

Conclusion

On ne va pas répéter les 4 étapes ci-dessus pleins de fois durant le projet. On va juste faire une fois chaque étape. Ce sera donc des grosses étapes, mais qui ne seront fait qu'une fois dans un soucis de gains de temps. Le plus gros avantage est qu'on va donc pouvoir spécialisé des équipes pour chaque partie du développement(Une équie d'analyste, de designer, de développeurs et une dernière équipe de vérification).

Conclusion du jeu de dés

cas où on jette juste les dés : Productivité de certains gaspillé

cas du bottelneck : l'ensemble de la chaîne à la productivité de la partie la moins productive

cas où l'on peut aller travailler là où c'est nécessaire : On ne gaspille pas son effort, mais nécessité de communiquer.

Cycle en V

Concept inventé pour justifier le fait que pour chaque phase de travaille, il y aura une phase de tests. Cela permet de vérifier le bon fonctionnement de la production des différentes phases de travail.

Test d'acceptance : on vérifie avec l'utilisateur que le produit convient bien à ses attentes.

Coût d'un bug

Le coût de l'apparition d'un bug augmente avec l'avancement du travail : un bug (une macompréhension) qui apparaît dans la phase récoltage des données, sera rapidemment corrigé. Alors qu'un bug qui apparaît en production sera beaucoup plus problématique à corriger.
$\rightarrow$ plus tôt le bug est découvert mieux c'est.

Les types de diagramme

- **GRANTT**
- **PERT :** le chemin critique, c'est la suite des tâches réellement importante pour la livraison du projet. Pour optimiser le temps de livraison, il faut obligatoirement

optimiser les tâches présentes dans le chemin critique.

Métaphore de la construction

On essaye souvent de comparer le développement informatique avec le domaine de la construction de maison. Mais malheureusement, les deux n'ont pas grand chose en commun parce qu'en développement informatique, tout est plus compliqué, et le domaine des possibles est largement plus élevés. De plus, en informatique, on peut changer le résultat d'une application après la production d'un produit, alors qu'une maison, qu'en c'est fait, c'est fait, plus moyen de changer.

l'implémentation d'un code est une activité de création, ce qui prend un temps difficile à définir !