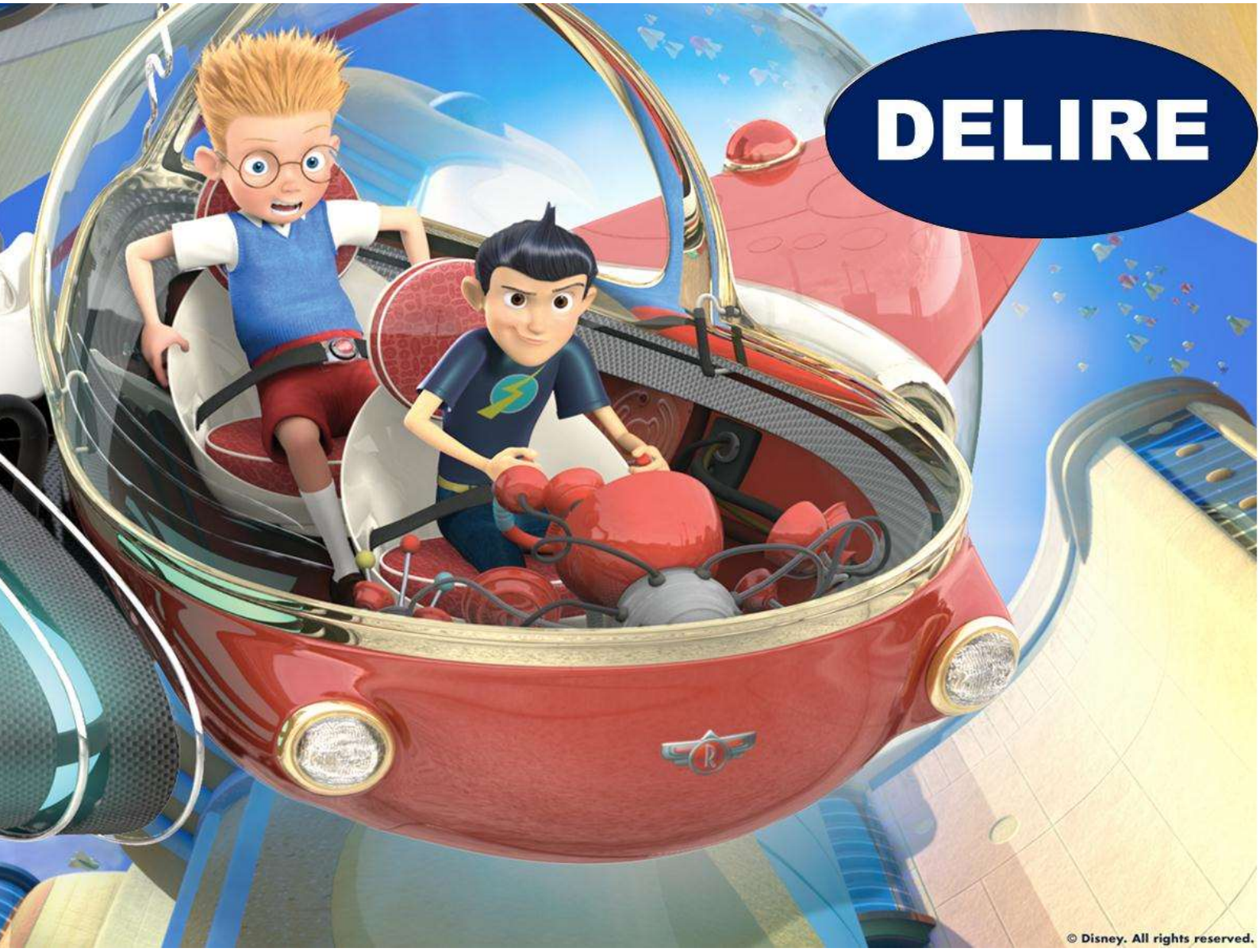

Le projet **DELIRE**

Développement par **E**quipe de **L**ivrables **I**nformatiques et **R**éalisation **E**ncadrée

Comprendre la gestion de projet complexe pour bien démarrer le projet

Introduction
Pour beaucoup d'entre vous, un projet informatique c'est essentiellement se jeter sur son clavier pour pisser du code, secouer le résultat jusqu'à ce qu'il tombe en marche, et considérer alors le travail comme terminé.
Ceci fonctionne bien lorsqu'il s'agit d'un TP sur un sujet technique donné.
Malheureusement, le projet DELIRE a une autre ambition. L'objectif est de vous mettre en situation de développement de projet dans un environnement industriel.

Pour mémoire, voici la définition normalisée d'un projet : action spécifique, nouvelle, de durée limitée, qui structure méthodiquement et progressivement une réalité à venir.

Quel sont les difficultés du projet DELIRE

1. Comprendre le besoin du client, qui ne s'est pas exprimé au travers d'un besoin informatique bien structuré, mais qui a lancé un certain nombre de pistes pour exprimer ce dont il pensait avoir besoin
2. Travailler en équipe de 6 ou 7 :
    - Travailler tout seul c'est facile.
    - Travailler en binôme c'est plus difficile.
    - Travailler en équipe c'est vraiment compliqué.

    Et si on ne s'organise pas cela vire rapidement au cauchemar.
3. Travailler sur une longue période. Vous avez l'habitude de travailler à l'arrache au dernier moment pour rendre vos devoirs. Là cela va être compliqué car je vais vous demander chaque semaine un résultat. Vous verrez que dans le projet DELIRE il y a deux rushs, et que démarrer le second début janvier est très difficile.
4. Etre capable de réaliser chacun de son côté un ou N composants logiciels, et que par miracle le produit tombe en marche quand on assemble ces différents composants
5. Arriver en temps et en heure avec un produit qui réponde aux besoins du client, qui soit de suffisamment bonne qualité pour être déployé en environnement de production, et dont le coût de développement n'ait pas explosé le budget alloué pour le réaliser.

Dans la suite de ce document, je vous propose d'analyser ces 5 difficultés et de voir où cette réflexion nous conduit.

Arriver en temps et en heure
DELIRE est un projet qui dure de septembre à fin mars ou début avril de l'année suivante

- Il se termine par une soutenance par équipe, dans laquelle doit être réalisée une démonstration du produit fini.
- Pour votre information, une soutenance intermédiaire, sans démonstration, est positionnée à la fin décembre.

Nous cherchons à obtenir un produit qui réponde aux besoins du client et soit de suffisamment bonne qualité pour pouvoir être déployé en environnement de production. Cela signifie

- Des fonctionnalités conformes aux besoins exprimés initialement par le client
- Un bon niveau de performance

- Un bon niveau de fiabilité
- Un bon niveau de convivialité (un produit facile à utiliser)
- Un bon niveau de maintenabilité (une démarche standard de reporting de problème, une procédure claire d'installation d'update de maintenance ou de fonctionnalité)
- Une documentation qui permette à l'utilisateur de comprendre la logique du produit, et le comportement à adopter en cas de problème
- Une documentation qui permette à l'administrateur de déployer le produit en environnement industriel : dimensionnement et customisation de la base de données, gestion de la sécurité, gestion des sauvegardes…

Pouvoir valider le niveau de qualité (fonctionnalité, performance, fiabilité, convivialité, maintenabilité, et documentation), il est nécessaire de disposer de temps pour assembler les composants logiciels, tester le produit final, le documenter et corriger les bugs trouvées. Cette période de temps est appelée phase de convergence.

Pour optimiser la phase de convergence, il faut que le temps nécessaire pour assembler les composants, et obtenir un produit final qui soit testable, soit très court. Cela suppose que les composants communiquent correctement ensemble, qu'il ne manque pas un composant pour obtenir le produit final, et que chacun travaille pour que le ou les composants qu'il a à développer soit faits dans les temps et avec un bon niveau de qualité. Cette période de temps est appelée phase de réalisation. Durant cette période, chacun travaille dans son coin sans avoir de visibilité importante sur le travail des autres.

Pour que chacun puisse travailler dans son coin, livrer son ou ses composants logiciels au bon moment avec un bon niveau de qualité, et que les composants ainsi réalisés communiquent correctement ensemble et qu'il ne manque pas un composant pour obtenir le produit final, il est nécessaire d'avoir défini, pour chaque développeur et pour chaque composant :

- la liste des tâches à réaliser
- le planning des tâches à réaliser,
- les fonctionnalités à obtenir dans le composant,
- les services qu'on va chercher dans d'autres composants et la façon de les appeler, les objectifs de qualité à obtenir et la façon de les mesurer,
- la façon de réagir dans le cas où le plan ainsi proposé ne se déroule pas bien.

Tout cela nous conduit à réaliser divers documents

- Le Plan Qualité
- Le Plan de Gestion des Risques
- Le Plan Management
- Les Spécifications Techniques Détaillées et les Unit tests
- Le Plan de Tests
- L'Organigramme de Tâches
- Le Budget détaillé des phases de Réalisation et de Convergence
- Le Planning détaillé des phases de Réalisation et de Convergence

Cette période de temps est appelée phase de structuration

Pour que l'on puisse définir le planning des tâches à réaliser, les fonctionnalités à obtenir, les objectifs de qualité et la façon de réagir, il faut avoir préalablement défini

le scope du produit qu'on va développer, la façon dont il va se présenter au client, et comment on va partager le travail à réaliser en composants logiciels.

- le scope du produit et la façon dont il va se présenter au client s'appelle l'architecture fonctionnelle et la maquette d'IHM
- comment on va partager le travail à réaliser en composants logiciels s'appelle l'architecture logique.

Et pour pouvoir définir nos deux architectures (fonctionnelle et logique), il est nécessaire d'avoir défini, à partir des besoins plus ou moins bien exprimés par le client, la liste de fonctionnalités majeures à mettre dans le produit : cela s'appelle les Spécifications Fonctionnelles Générales.
La demande du client risque d'être trop gourmande par rapport au délai qui vous est imparti. C'est pourquoi, on définit généralement dans un projet logiciel des versions de delivery qui vont s'étaler dans le temps : les fonctionnalités les plus importantes dans la première version, les fonctionnalités les plus difficiles dans la dernière version. La façon dont on découpe le projet en version est délicate : si on ne s'y prend pas bien, le travail de ré-architecture du produit pour passer de la version N à la version N+1 peut s'avérer très couteux et mettre en péril notre équilibre financier.
Nous verrons plus tard le problème du budget, mais sachez d'ores et déjà qu'un travail d'élaboration de budget (appelé Business Plan) doit être réalisé en parallèle du choix des Spécifications Fonctionnelles Générales.
Cette période de temps où, à partir du Cahier des Charges, on élabore

- Les Spécifications Fonctionnelles Générales.
- Les Business Plans
- L'architecture Fonctionnelle
- La maquette d'I.H.M.
- L'architecture Logique

est appelée Phase de Spécification

En d'autres termes, le projet DELIRE est composé de 4 phases :

- Phase de Spécification
- Phase de Structuration
- Phase de Réalisation
- Phase de Convergence

Travailler sur une longue période.
Le projet DELIRE s'étale sur globalement 6 mois.
Vous aurez l'impression que c'est très long.

- Vous aurez au début le sentiment que vous avez tout votre temps, ce qui est un piège
- Vous aurez parfois des moments de lassitude, au regard du travail à faire chaque semaine
- Mais vous verrez qu'en fin de projet il vous manquera de toute façon 15 jours pour finir

Je peux vous détailler de grandes théories basées sur Nordon et Rayleight, mais nous allons faire simple

- 1mois ½ pour la phase de Spécification
- 1mois ½ pour la phase de Structuration
- 1mois ½ pour la phase de Réalisation
- 1mois ½ pour la phase de Convergence

<u>Les difficultés de la phase de Spécification</u>
Comprendre ce qu'est une architecture logique est très compliqué, et demande un certain nombre d'itérations
Décomposer le projet en différente version est assez compliqué
Concevoir une maquette d'I.H.M. est facile, concevoir une maquette d'I.H.M. conviviale est difficile.
Comprendre les besoins du client, et les traduire en Spécifications Fonctionnelles Générales est assez compliqué
Mais la vraie difficulté du projet DELIRE, c'est d'obtenir très rapidement un Cahier des Charges : si, dans une phase de 6 semaines, vous mettez 4 semaines pour obtenir ce cahier des Charges, cela va devenir compliqué de tenir les délais.

<u>Les difficultés de la phase de Structuration</u>
Il n'est pas toujours évident de savoir ce qui relève du Plan Qualité, du Plan de Gestion des Risques, du Plan Management et du Plan de Tests
Néanmoins, la difficulté majeure va être de détailler, dans les Spécifications Techniques Détaillées, tout ce qu'il est nécessaire de faire dans un composant logiciel

- Identification claire des interfaces de programmation
- Liste des atomes de réalisation, et parfois documentation de l'architecture interne dudit atome
- Critère chiffré de qualification de chaque atome, et méthodologie pour le mesurer (test unitaire)

L'objectif est que les Spécifications Techniques Détaillées soient suffisamment détaillées pour pouvoir confier la réalisation à une autre personne du groupe, voire même à une personne extérieure au groupe, et que le résultat obtenu soit utilisable.
La raison qui fait que définir les Spécifications Techniques Détaillées est très compliqué est souvent que l'Architecture Logique du projet est insuffisamment travaillée et détaillée : de ce fait le but et les limitations de chaque composant ne sont pas bien explicités et tout le monde travaille dans le brouillard.
Ensuite, l'autre difficulté va être d'identifier toutes les taches à réaliser dans l'Organigramme des Tâches, d'affecter une durée et une responsabilité pour chaque

tâche, et à partir de ces informations de construire le Budget et le Planning détaillés des phases de Réalisation et de Convergence en vérifiant quatre critères importants :

- La cohérence du planning obtenu (une tache A ne peut pas finir après le début d'une tâche B si le résultat de A est nécessaire pour démarrer B
- Une même personne ne peut pas accomplir simultanément plusieurs tâches
- La répartition du travail doit être équitable entre tous les membres du projet
- Le budget et le Planning global obtenu sont cohérents avec le Budget et le Planning posés en début de projet

En général, le résultat obtenu en termes de Budget et de Planning va très au-delà de ce qui avait été initialement défini : ceci va nécessiter une itération sur le contenu de chaque version, pouvant impacter fortement la valeur ajoutée de la version, et demandant parfois un re-travail important en ce qui concerne l'architecture logique

Les difficultés de la phase de Réalisation

La plus grande difficulté de la phase de Réalisation est de démarrer
Il y a trois raisons à cette difficulté

- Il est difficile de se remettre au travail après les vacances de Noël, sachant qu'on a terminé la phase de Structuration sur les chapeaux de roue pour tenir la présentation intermédiaire de fin décembre
- Les Spécifications Techniques Détaillées étant floues voire inexistantes, ce qui a conduit à ne pas pouvoir définir le planning détaillé, personne ne sait ce qu'il a exactement à faire ni quand ;
- Les Spécifications Techniques Détaillées étant floues voire inexistantes, personne ne sait quels services il peut trouver dans les autres composants, ni la façon exact de les appeler.

La conséquence est que vers la mi-février, au moment où il faudrait finaliser la réalisation, et parce que le professeur responsable de l'enseignement vous met une pression un peu forte, vous allez devoir redéfinir l'Architecture Logique du projet, pisser les Spécification Techniques Détaillées en urgence, réaliser les composants à l'arrache, devoir faire des coupes sombres dans le projet ce qui remet en cause l'intérêt du produit, et vous engueuler copieusement.
Ne rêvez pas, vous n'y échapperez pas. Mais si vous l'avez anticipé, les conséquences seront sans doute moins dramatiques.
Au contraire, une réalisation qui prend la suite d'une phase de Structuration bien déroulée et accomplie, se passe en général très bien, et l'heureuse surprise est que le code est réalisé beaucoup plus rapidement qu'anticipé, avec un niveau de qualité bien supérieur. Mais sachez-le, cette configuration est rarissime même dans les entreprises éditrices de logiciel. On parle d'une industrie du logiciel, mais dans la réalité, cela reste un artisanat, voire un bricolage qui confine parfois à la pétaudière.
Et voici pourquoi votre code est malade

Les difficultés de la phase de Convergence

La plus grande difficulté de la phase de Convergence est de démarrer
Tout d'abord parce que la phase de Réalisation est loin d'être terminée. D'autre part, parce que chacun pense que nous sommes proches de la fin, que tout va bien se passer, et qu'il n'est pas nécessaire de paniquer.
Dans la réalité, cela ne va pas tout fait se passer comme cela

- La première difficulté va être de disposer de l'ensemble des composants
- La seconde difficulté va être d'avoir un ensemble complet, et pas un gros trou dans la raquette.

- La troisième difficulté va être de savoir comment tester le produit, ce qu'il faut mesurer et les valeurs attendues
- La quatrième difficulté va être de corriger les erreurs sans introduire de nouvelles régressions. Pour mémoire, cette phase s'appelle phase de Convergence, et non de Divergence.

Mais la phase de Convergence ne se limite pas à la simple convergence du produit.

- Il est également nécessaire de rédiger les documentations : documentation d'utilisation, documentation d'installation et d'administration, documentation de maintenance
- D'autre part, il est temps de faire le point sur le projet, et donc de reprendre tous les documents (appelés livrables) élaborés en phases de Spécification et de Structuration pour mettre en évidence les divergences en phases de Réalisation et de Convergence, en expliquer les raisons et en identifier les conséquences.
- Enfin, il restera à chacun à tirer les conclusions de ce projet, ce qu'il a appris, les difficultés rencontrées et les enseignement tirés : ce qu'on appelle le Post Mortem.

Travailler en équipe de 6 ou 7

Bien entendu, en phase de Structuration, vous serez tous futurs développeurs : chacun réalisera les Spécifications Techniques Détaillées du ou des composants logiciels qui lui auront été dévolus par le responsable de l'Architecture Logique

Bien entendu, en phase de Réalisation, vous serez tous développeurs. Mais chacun réalisera le ou les composants logiciels qui lui auront été dévolus par le responsable de l'Architecture Logique

Bien entendu, en phase de Convergence, chacun d'entre vous serez testeurs. Mais le travail de tests devra être réparti et coordonné dans l'équipe

Bien entendu, en phase de Convergence, vous serez tous développeurs : chacun corrigera les erreurs identifiées dans son ou ses composants.

Et bien entendu, en phase de Convergence, chacun reprendra les différents documents qu'il a élaborés en phase de Spécification ou de Structuration.

Mais par contre, durant les phases de Spécifications et de Structuration, il va falloir se partager les rôles qui sont plus ou moins apparus dans les chapitres qui précèdent. Ils sont grosso modo au nombre de 8 :

- Le MOA : Maitre d'OuvrAge
- Le MOE : Maitre d'OEuvre.
- Le Designer
- L'architecte
- Le responsable du Plan Qualité
- Le responsable du Plan de Risques
- Le responsable du Plan de Tests
- Le responsable du Budget et du Planning

Et bien entendu, en phases de Structuration, de Réalisation et de Convergence, vous garderez les rôles que vous avez eus en phases de Spécification ou de Structuration

- Ainsi le responsable du Plan Qualité suivra la qualité du produit jusqu'à la fin de la phase de Convergence
- L'architecte validera la cohérence et la complétude des composants logiciels jusqu'à la fin de la phase de Convergence…
- Et le MOE pilotera l'équipe jusqu'à la fin du projet.

Il y a plus de rôles que de membres de l'équipe, et chacun pourra être amené à tenir plusieurs rôles. Néanmoins, pour des raisons qui seront explicitées dans la suite du projet DELIRE, la tenue de certains rôles est incompatible avec la tenue de certains autres rôles.

- On ne peut pas être MOA et MOE
- On ne peut pas être MOA et Designer
- On ne peut pas être Designer et Architecte
- Le rôle du MOE est très chronophage et est exclusif des autres rôles chronophages de Designer, d'Architecte ou de responsable Budget et Planning

Dernier point :

- La personne qui est responsable d'un rôle particulier n'est pas obligée de faire seul le travail, elle peut avoir un certain nombre d'adjoints
- Mais ultimement, c'est elle qui est responsable des documents produits et c'est à elle de prendre une décision en cas de divergence au sein de l'équipe

Les responsabilités de chaque rôle

Le MOA : Maitre d'OuvrAge

Il représente les intérêts du client

Il a la responsabilité de rédiger :

- Le Cahier des Charges, incluant niveau de qualité requis, délais attendus et budget alloué pour le projet
- Le Business Model MOA : ce que le produit va lui rapporter, ce qu'il est prêt à payer pour le faire développer

Il a la responsabilité de valider les Spécifications Fonctionnelles Générales et le découpage en versions du projet

Il a la responsabilité de conduire les tests de recette en fin de projet pour accepter (ou refuser) le produit développé.

Le MOE : Maitre d'OEuvre.

Il représente les intérêts de l'équipe des fournisseurs, grosso modo l'équipe de projet Dans la suite de ce document en particulier et du projet DELIRE en général, nous l'appellerons le Chef de Projet

Le rôle principal du MOE n'est pas d'être un chef autoritaire, mais au contraire de faciliter le travail de collaboration de son Designer, de son Architecte, de ses responsables de Plan Qualité, de Plan de Risques, de Plan de Tests et de Budget et Planning. Son rôle essentiel est de donner envie à l'équipe d'experts, avec qui il travaille, de faire et de réussir le projet.

C'est lui qui constitue l'équipe du projet en fonction des savoir-faire et des appétences de chacun. Grosso modo vous avez tous un bagage équivalent et

chacun d'entre vous peut tenir chacun des rôles. Comme le projet DELIRE est un projet d'enseignement, si quelqu'un souhaite ardemment un rôle, donnez le lui : il vaut mieux une personne motivée et souhaitant apprendre qu'une personne un peu plus compétente mais non motivée par le poste.
Le MOE a la responsabilité de rédiger le Plan Management :

- Identifier comment motiver les troupes
- Identifier les problèmes humains qui pourront survenir durant le projet et préparer à l'avance la façon de les résoudre.

Dernier détail, le chef de projet ne porte pas sur ses seules épaules la responsabilité d'un projet : un projet est quelque chose qui se fait et qui se gagne en équipe

Le Designer

Le Designer est celui qui analyse le Cahier des Charges du MOA et qui en déduit, en accord avec l'Architecte, les fonctionnalités clé du projet (les Spécifications Fonctionnelles Générales), dont il met en évidence les éléments différentiateurs de son offre
C'est lui qui identifie les types d'utilisateurs du produit : chef de projet, contributeur, reviewer, administrateur… Dans le cas de DELIRE : médecin, infirmier, secrétaire…
C'est lui qui décompose, en accord avec l'Architecte, ces Spécifications Fonctionnelles Générales en fonction des versions. Pour ce faire, il a la charge de rédiger le ou les scénarios fonctionnels du produit pour chaque version et pour chaque rôle.
C'est lui qui élabore, en accord avec l'Architecte, l'Architecture Fonctionnelle de chaque version, c'est-à-dire l'arborescence des fonctions et des commandes disponibles en fonction des types d'utilisateurs du produit.
C'est enfin lui qui dessine et propose la maquette d'I.H.M., en accord avec les standards définis par le responsable du Plan Qualité.

L'Architecte

L'Architecte est responsable de l'identification des composants logiciels, de la définition claire de la finalité de chacun des composants, et de la spécification des interfaces des services proposés par chaque composant, services qui peuvent être appelés soit par l'utilisateur via une commande soit par un autre composant. .

Un composant logiciel est un atome de réalisation. Un composant logiciel est :

- Fortement cohérent : il a une et une seule finalité clairement identifiée
- Faiblement couplé : il a un minimum de liens avec les autres composants

On distingue plusieurs types de composants :

- Des composants de dialogue et de présentation : ils sont souvent structurés par les types d'utilisateurs du produit identifiés par le Designer
- Des composants de Business Unit : ils sont structurés par les savoir-faire métier
- Des composants de gestion de données : ils sont structurés par le Data Model de l'application, qui est de la responsabilité de l'Architecte

La finalité de l'Architecture Logique, c'est-à-dire la définition des composants logiciels est triple

- Supporter l'exhaustivité des fonctions attendues pour la version en cours

- Gérer la maintenabilité et l'évolutivité du produit, par une anticipation claire des besoins, une bonne encapsulation des internes des composants, mais également par un système défensif de programmation qui permette un diagnostic simple et rapide des causes d'erreurs
- Permettre la répartition du travail de façon équilibrée entre les équipes de développement en fonction des charges de travail et des savoir-faire.

Une des activités essentielles de l'Architecte est d'identifier la difficulté d'un développement, compte tenu des technologies qui ont été retenues pour le projet
C'est dans ce cadre qu'il partage la responsabilité avec le Designer d'identifier les fonctions à livrer dans chaque version :

- Le Designer a une bonne vision de ce qui apporte de la valeur ajoutée d'un développement pour l'utilisateur
- L'Architecte a une bonne vision de la complexité et, par ce faire, du coût dudit développement

A eux de trouver, dans une approche win-win, un plan de déploiement du produit qui les satisfasse tous deux.

Un autre rôle de l'Architecte est d'anticiper les développements qui seront nécessaires dans les versions futures : il peut être intéressant d'anticiper un développement si cette anticipation réduit de façon importante le coût de développement d'une version postérieure
A titre d'exemple, le fait de pouvoir choisir la langue d'utilisation du produit doit être pris en compte dans la première version, même si la fonctionnalité n'est proposée sur le marché qu'en 3ème version.
.
Attention, le rôle d'Architecte est complexe et chronophage. Et la réussite du projet est souvent liée à la qualité de son travail :

- Une Architecture Logique indigente conduit à ne pas pouvoir rédiger les Spécifications techniques Détaillées, à ne pas pouvoir élaborer le planning, et à mettre l'équipe dans le corner soit en phase de Réalisation soit lors de l'assemblage des composants logiciels en phase de Convergence
- Au contraire, une Architecture Logique simple et robuste permet de bien partager le travail en phase de réalisation et de garantir un assemblage des composants logiciels aisé en phase de Convergence.

Trois conseils pour notre Architecte

- Ne faites pas une Architecture Logique pour vous faire plaisir et vous lancer un défi : n'oubliez pas que vous n'êtes pas le seul à devoir réaliser le projet, et que certains n'ont pas le même niveau de compétence informatique que vous.
- En permanence privilégiez la simplicité. Et faites la chasse aux développements inutiles.
  Un développement est utile :
  - Quand il est nécessaire pour la version en cours
  - Ou quand il est présent (même s'il est caché à l'utilisateur) pour faciliter la migration de la version en cours vers les versions futures : il en par exemple ainsi des structures de données de la base de données qui peuvent être en avance sur le niveau fonctionnel de la version en cours.

- Tant que vous n'êtes pas satisfait de votre travail, reprenez-le : concevoir une architecture Logique est un métier qui s'apprend avec le temps, mais à un moment on sent qu'on tient quelque chose de robuste et d'élégant.

Le responsable du Plan Qualité

Un projet c'est :

- Un produit à réaliser pour le compte d'un client
- Un processus à exécuter pour réaliser le produit

Dans un projet il y a donc deux domaines de qualité

- La qualité du produit final
- La qualité d'exécution du processus

La qualité du produit final doit respecter un contrat passé avec le client. On ne peut pas se contenter de dire : un produit performant, un produit fiable, un produit convivial… Le jour de la livraison du produit, il risque d'y avoir conflit avec le MOA, qui derrière les mots abstraits de performance, de fiabilité ou de convivialité mettait des chiffres très différents de ce que vous livrez. Les objectifs de qualité du produit doivent donc être chiffrés (on définit la valeur minimum à atteindre) et mesurables (on définit comment on obtient la valeur).
Exemple : le temps de chargement d'une page de 100 résultats à partir de la soumission d'une requête dans une base de données de 10,000 valeurs est inférieur à 1 seconde sur une machine Lenovo W541 avec 100 gigas de disponible sur le disque C, sur laquelle tout est installé en local, et sur laquelle aucune autre application ne tourne.

La qualité d'un processus est difficile à mesurer. La solution est donc de définir des standards à respecter, et de demander à chacun des membres de l'équipe de les respecter.
Voici 4 exemples de standards que vous pouvez définir

- Un standard (Polices de caractère, couleur des titres, nomenclature…) commun pour tous les livrables du projet : Architecture Logique, Plan de Gestion des Risques, Organigramme des Tâches…
- Un standard de présentation : numérotation des pages, logo du projet, espace de titre…
- Un standard de Spécifications Techniques Détaillées : fonctionnalité, API d'appel, algorithme, cas d'erreur, test unitaire, performance attendue…,
- Un standard de Programmation : entête, indentation, règle sur les commentaires, nombre maximum de lignes…

Respecter un standard est toujours compliqué lorsqu'il n'y a pas d'outil pour le supporter. Je vous recommande de fournir aux membres de l'équipe un template de livrable, un template de présentation, un template de Spécification technique Détaillée, un ou N templates de squelette de programmation.
Vous verrez que la rédaction des livrables et les présentations arrivent très tôt dans le projet DELIRE. Attelez-vous vite à ces deux templates.
Dans la suite du projet vous serez amenés à définir un standard de « Fièvre du samedi Soir » et un standard de fiche de livraison, mais nous aurons l'occasion d'en reparler.

Le responsable du Plan de Risques

Un projet c'est une suite de problèmes qui apparaissent lors des phases de Réalisation et de Convergence, et qu'il va falloir traiter quand ils seront là.
Bien sûr, une solution peut être de faire l'autruche, de se mettre la tête dans le sable et d'attendre que les problèmes arrivent. Et le jour où ils arrivent, ne pas savoir comment réagir, et découvrir un peu tard que le problème met en péril votre projet.
Mais la plupart des problèmes peuvent s'anticiper.

Dans le Plan de Gestion des Risques d'un projet, on segmente souvent les risques en 4 catégories

1. Economiques : évaluer et tenir les prévisions budgétaires
2. Organisationnels : équilibrer délais et ressources, actualiser en permanence les estimations de charges
3. Fonctionnels : travailler en commun, valider progressivement
4. Technologiques : maîtriser la nouveauté

L'objectif de la gestion des risques est de réduire, par des méthodes préventives, opérationnelles ou correctives, les différents risques identifiés et retenus dans chacune des 4 catégories.
Pour chacun de ces risques on va donc définir un plan de levée (plan de résolution)
Il y a deux types de plans :

- **Les plans qui définissent des actions de réduction de la probabilité du risque et de l'impact du risque sur le projet : on appelle ceci un plan de levée préventif.**
- **Les plans qui définissent les actions à effectuer si le risque advient : on appelle ceci un plan de levée correctif.**

Le responsable du Plan de Tests

Malgré toute la rigueur que vous mettrez dans la Spécification, la Structuration puis la Réalisation de votre projet, il y aura nécessairement des erreurs dans le produit obtenu après codage. Ces erreurs peuvent être :

- Des oublis vis-à-vis des Spécifications Fonctionnelles Générales
- Des réponses non conformes aux spécifications
- Des problèmes de convivialité rendant le produit difficilement utilisable
- Des bugs de fiabilité ou de performance…

Depuis que l'informatique existe, la solution a toujours été et reste de chercher à découvrir un maximum d'erreurs et à les corriger avant la mise du produit sur le marché, en minimisant le nombre de régression introduites par ces corrections.
Cette activité de correction est assez chronophage, il est donc nécessaire de la structurer. On exécute pour ce faire 4 types de tests successifs

- Les Unit Tests : ils concernent un composant logiciel donné, ils sont écrits par le développeur dudit composant durant la phase de Structuration, et sont réalisés par le même développeur durant la phase de Réalisation
- Les Integration Tests : ils ont pour objectif de vérifier la cohérence des appels entre les composants logiciels, lors de l'assemblage de ceux-ci en début de phase de Convergence
- Les Function Tests : ils ont pour objectif de tester que le produit obtenu est conforme aux Spécifications Fonctionnelles Générales. En toute logique, ils ont été rédigés par le Designer durant la phase de Réalisation

- Les System Test : Ils ont pour objectif de valider le comportement sain (pas de carton, messages d'erreurs compréhensibles) du produit en utilisation industrielle : undo, entrée erronée, le serveur tombe, le client tombe, 5000 utilisateurs simultanés…

En règle générale, le MOA et le MOE se mettent d'accord en début de projet, une fois les Spécifications Fonctionnelles Générales validées, sur la définition exacte des tests de recette (qui sont de la responsabilité du MOA). Les tests de recette doivent être inclus dans les Function Tests ou les System Tests

Le responsable du Budget et du Planning
Rappelons la définition normalisée d'un projet : action spécifique, nouvelle, de durée limitée, qui structure méthodiquement et progressivement une réalité à venir.

Au début du projet, on n'a qu'une vague idée du coût et du délai qui seront nécessaires pour réaliser le produit.
On va juste poser :

- Des dates de fin de chacune des phases : Spécification, Structuration, Réalisation et Convergence, en fonction de la date finale imposée par le client
- Un nombre d'heures de travail, en fonction du budget proposé par le client, et de la disponibilité de l'équipe projet.

On va ensuite définir les actions à réaliser en phase de Spécification et de Structuration et les positionner dans le calendrier. Prenons un exemple. Si la phase de spécification doit durer 6 semaines, il serait bien d'avoir :

- Le Cahier des Charges et les Spécifications Fonctionnelles Générales au bout de 2 semaines
- Une première version de l'Architecture Fonctionnelle et de l'Architecture Logique au bout de 4 semaines
- La version définitive de l'Architecture Fonctionnelle et de l'Architecture Logique, segmentées par versions, la maquette d'I.H.M. et les deux Business Plans au bout de 6 semaines

Au fur et à mesure de l'avancée du projet, on a de moins en moins d'incertitudes, les choix structurant se font, on connait l'expertise et la productivité des équipes métier, on peut donc être beaucoup plus précis sur la date et le coût de chaque tâche.
L'objectif est d'avoir un planning et un budget détaillé des phases de Réalisation et de Convergence à la fin de la phase de Structuration.
Pour votre information, dans les deux plus gros projets de concurrent Engineering, le Boeing 787 et l'Airbus A380, qui ont impliqués chacun plus de 10,000 ingénieurs simultanément en phase de Detailed Design, les activités de chacun était définies au jour le jour. Ce qui n'a pas empêché les deux projets de prendre également chacun plusieurs années de retard et d'exploser leurs budgets.

Dernier point, mais d'importance : le rôle du responsable du Budget et du Planning est non seulement de définir le budget et le planning, mais également de vérifier que ceux-ci sont suivis, d'identifier les divergences et de recaler régulièrement budget et planning

Le projet DELIRE c'est