Bloc 1

Les cours d'analyse du bloc 1

Introduction

L'*analyse* est le procédé de raisonement qui va de la connaissance desparties à celle du tout. C'est l'opposé de la synthèse. Étude et examen sont des synonymes.

Il s'agit de déterminer les contours et le fonctionnement général de quelque chose avant de le construire. Par exemple à l'aide de dessins, plans, maquettes, etc. Cette description permet de communiquer avec les clients et les concepteurs.

En informatique il s'agit d'identifier les besoins d'un organisme ou d'un ensemble d'utilisateurs. De représenter ces besoins avec des modèles spécifiques et de faciliter la communication avec les différents acteurs (clients, analytes et concepteurs)

Les raisons d'échec d'un projet sont souvent liée à une mauvaise définition des objectifs métiers et de la formulation des besoins.

- Périmètre (ce qu'il faut faire)
- Business rules (les règles métier)
- Data modelling (les données nécessaire)
- UX design (l'expérience utilisateur)

Les règles métier sont les règles du fonctionnement de l'entreprise, comment elle fonctionne, son organisation, etc à fin de la rendre plus efficace.

L'analyste va identifier les besoins en rencontrantles clients, utilisateurs, sponsors, etc. Il va consulter les ressources existantes tel que la documentation, il pose des question et met en évidence les incohérences.

Ensuite l'analyste crée des modèles sur différents aspects du système (données, processus, fonctionalités, UI, etc) pour engager la discussion.

L'analyste va ensuite faire valider ses modèles en discuttant avec l'équipe avec l'aide d'autres documents (cahiers des charges, spécifications fonctionnelles, etc) et peut aussi participer aux aspects budgétaires (évaluation de la charge de travail)

Un modèle en informatique a pour objectif de structurer les informations et activités d'un système (données traitements, flux d'informations). Il a pour but de couvrir les points les plus importants. C'est un outil de communication et d'information sur le projet.

L'analyste estla personne qui sert de pont entre les clients et les concepteurs. Il doit pouvoir faire la "traduction" du client au programmeurs. L'analyste se doit d'être précis et rigoureux pour déceler les trous dans le discours du client. Il doit être un bon négociateur et communiquant.

Sa place dans l'organisation depends de l'entreprise, du projet et de la méthodologie utilisée.

La méthode de l'analyse est de définir les besoins, déterminer la faisabilité et la cohérence de la demande et de structurer les données. Le cours va être plus basé sur la partie identifications des besoins et modélisation de données et interface.

Le but du cours est de décortiquer et interpreter le discour du client, de maitriser les concepts, outils et vocabulaire et de pouvoir modéliser des systèmes complexes en suivant une méthode.

Diagramme de Use Case

- Wikipedia - Unified Modelling Language
- Wikipedia - Use case diagram

Le but d'un diagramme Use Case est de décrire visuellement les besoins d'un organisme (intéraction des aceturs avec le système et ses fonctionalités). Il vient de l'UML (Unified Modelling Language)

On va ainsi définir quelles sont les fonctionalités (dans le cadre de l'analyse, donc seulement les principales) et qui peut y accéder, pour y faire quoi et comment.

Voici les différents éléments du diagramme :

- Le système est représenté par un rectangle délimitant le périmètre.
- Les acteurs externes (éléments externent qui intéragit avec celui ci), sont représenté par leur rôle sous forme de bonhomme baton (stickman). Les acteurs non humains sont définis par un rectangle libéllé
  - Les acteurs primaires qui initient les CUs (cas d'utilisation) sont placés à gauche du système
  - Les acteurs secondaires qui aidentà la réalisation des CUs mais qui n'en sont pasles bénéficiaires à droite.
  - Une flèche peut être tracée d'un acteur à un autre si le premier acteur peut faire tout ce que le deuxième peut. Exempe: Administrateur → éditeur
- Les cas d'utilisations (CU) sont représenté par des ovales libéllé. Le nom contient un verbe d'action à l'infinitif, choisi du point de vue de l'utilisateur, réalise un service du début à la fin. Exemple: « *Lire un message* ». Il sont aussi reliés avec les acteurs avec une ligne

Ce diagramme permet d'avoir une vision globale des fonctionnalités du système, compréhensible par tout le monde, même non initié.

Ensuite il y a la « *Spécification textuelle* » qui décrit avec plus de détails les cas d'utilisations. Le texte doit fournir, pour chaque fonctionnalité:

- Le même nom que dans le diagramme (ex: Lire un message)
- L'identification de l'évènement déclencheur (ex: L'utilisateur veut lire ses nouveau messages ou relire d'anciens messages)
- La description de la fonctionnalité. La description doit être suffisament générale pour ne pas être dépendante sur l'interface.
- L'identification du/des acteurs impliqué(s) (ex: l'utilisateur).
- Les éventuelles pré/post conditions (ex: L'utilisateur doit être authentifié et une fois lu les messages ne doivent plus apparaitre en gras). Les post conditions sont toutes les choses qui doivent avoir été fait pour que l'UC soit considéré comme terminé.

En pratique

Pour faire le diagrame :

1. Trouver quel est le système de la situation. Il s'agit bien du système informatique. Donc il faut éviter de représenter un élément comme "Bibliothèque" ou "nom de la société" et être plus précis comme "Système de gestion biliothécaire" ou "Site marchant BeGood". Créer un rectangle libéllé représentant le système.
2. Trouver les acteurs primaires et secondaires. Les acteurs sont ceux qui interagissent *directement* avec le système. Les acteurs primaires (qui initient les UC) sont à droite du système, les acteurs secondaires (qui aident à la concrétisation du UC mais qui n'en bénéficie pas) sont mis à droite. Les acteurs humains sont représentés par des stickman tandis que les non humains peuvent être représenté par un petit rectangle libellé. Le nom doit être un *rôle*, et pas un nom ou personne en particulier.
3. Pour chaque acteur noter les actions possible en excluant toutes celles qui sont en dehors du système. Essayer de le synthétiser un maximum, mais tout en incluant toutes les actions dans le système. Les UC doivent commencer par un *verbe à l'infinitif*. (Relire les consignes en se mettant dans la peau de chaque acteur pour voir ce qu'iel pourrait faire). Puis ensuite relire les UC pour vérifier leur portée (mot "Gérer" par exemple)
4. Lier les acteurs avec les UC via une ligne (sans flèche), et lier les acteurs entre eux quand un role peut faire tout ce qu'un autre role peut faire (administrateur --> utilisateur par exemple)

Pour faire la spécification textuelle (à faire par UC) :

1. Définir quel est l'élément déclencheur, cela peut être un souhait ou un désir d'un acteur, ou encore lié au résultat d'un autre UC.
2. La description doit être complète mais pas trop longue et ne doit pas dépendre de l'interface utilisateur (qui peut changer dans le temps).
3. La liste des acteurs impliqués.
4. Les préconditions (tel que l'authentification par exemple), pour que le UC prenne place.
5. Les postconditions, qui définissent l'état du système après le UC. C'est à dire ce qui doit avoir changé pour que le UC soit défini comme terminé.

> Note: Le mot "gérer" fait référence à CRUD (créer, lire, mettre à jour et supprimer) en analyse

Diagramme activité

Le diagramme d'activité sert à donner une vision globale et temporelle d'une partie dynamique d'un système. C'est a dire un enchainement d'activité que l'utilisateur ne voit pas forcément et qui sont opérée par un système.

Cela peut servir à modéliser un algorithme, la dynamique d'un cas d'utilisation, ou un "processus métier".

Fonctionnement du diagramme

Distributeur de billets

Le diagramme est divisé en 2 colonnes (qui sont appellées swimlanes), la colonne de gauche sert à représenter les intéraction avec le système. Tandis que la colonne de droite est la dynamique du système en elle même. Cela permet de modéliser les actions entre le systèmes et les différents acteurs.

Le début et la fin sont symbolisés par un point noir ou un point noir entouré. Pour symboliser la fin d'un seul flux seulement on utilise un rond avec une croix dedans.

Il y a ensuite différents "noeuds" sur notre diagramme :

Symbole	Nom	Description
Ovale	Noeud d'exécution	Fait une action
Losange (1 input, plusieurs output)	Noeud de décision	En fonction d'une condition, elle va continuer dans un ou l'autre direction. Ses conditions sont représentés sur les lignes
Losange (plusieurs input, 1 output)	Noeud de fusion	A l'inverse d'un noeud de décision qui permet de diviser le processus en plusieurs possibilités, le noeud de fusion va recombiner les différentes possibilités.
Barre (1 input, plusieurs output)	Fork	Cela permet de créer des flux parallèles (des actions qui vont se déclencher simultanément)
Barre (plusieurs input, 1 output)	Join	Fait l'inverse de fork en recombinant les flux parallèles en un seul
Un genre de sablier	Evenement temporel	Modélise un évènement qui se déclenche à un moment prédéfinis (par exemple fin du mois)

Symbole	Nom	Description
Un rectangle avec une flèche et avec une "réception de flèche"	Envoi et réception d'un signal	Symbolise une signal asynchrone. C'est a dire que le programme va envoyer un signal vers un autre thread et attendre qu'une tache soit effectué avant de continuer