MODELISATION UML2

ETUDE DE CAS DIRIGÉE

PROCESSUS DE MODÉLISATION SIMPLIFIE PAR UML2 D'UNE APPLICATION WEB

Table des matières

Introduction

Le développement de site Web est souvent le royaume où règne la loi du « vite fait, mal fait ».

Néanmoins si cette approche convient tot à fait aux sites simples, elle pose de gros problème de cohérence, de maintenance, de gestion de projet et de performances pour les applications de plus grande ampleur.

Pourquoi Modéliser

La modélisation est une activité fondamentale dans le développement logiciel.

Il permet d'anticiper les résultats de codage.

C'est une représentation abstraite du système à réaliser.

Il permet de communiquer avec les différent acteurs du projet.

Un modèle sert des objectifs différents suivant l'activité de développement.

Les activités de spécification des exigences

Le système à réaliser est considéré comme une boite noire. On étudie sa place dans le système métier de l'entreprise.

On modélise le contexte afin de délimiter précisément les frontières fonctionnelles du système.

Les activités d'analyse

Les objets découvert lors de ces activité représentent des abstraction des concepts manipulés par les utilisateurs du système.

Deux visions sont développées dans le modèle d'analyse, la structure statique et le comportement dynamique.

Les activités de conception

On modélise ici les concepts informatique manipulé par les frameworks, les langages ou les plates formes de développement.

Le modèle sert à étudier, documenter, communiquer et anticiper les solutions.

Le modèle représente aussi pour le déploiement, les matériels et les logiciels à inter-connecter.

Le modèle

On dit que A est un bon modèle de B si A permet de répondre de façon satisfaisante à des questions pré-définies sur B.

Un bon modèle doit être construit :

- au bon niveau de detail,
- selon le bon point de vue.

Pensez à l'analogie de la carte routière. pour circuler dans Anglet, la carte de France serait un peu inutile. En revanche la carte d'Anglet ne suffit pas pour aller de Paris à Anglet.

A chaque voyage correspond la « bonne carte ».

Aujourd'hui le standard industriel de modélisation objet est UML. Il est sous la responsabilité de l'OMG.

www.omg.org

Les documents sur UML élaborés dans le cadre de l'OMG sont publics et disponibles sur le site:

www.uml.org

Choix du processus de développement

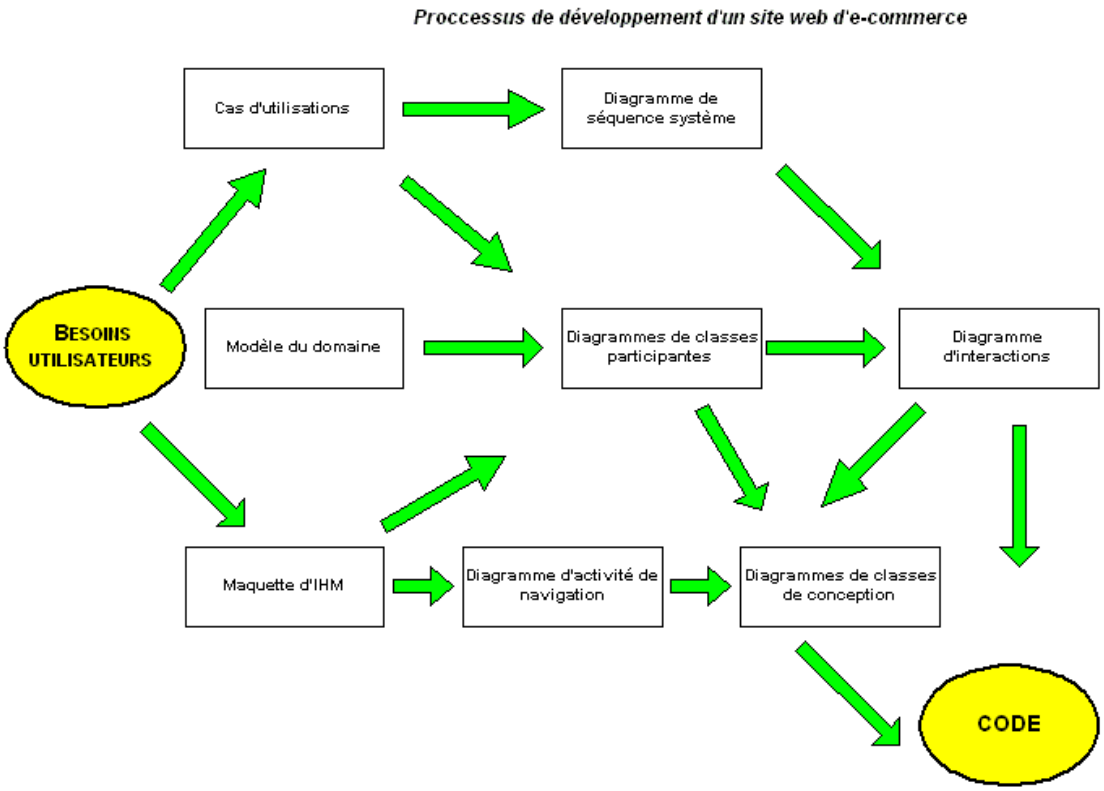
Le processus que nous allons suivre pour le développement d'une application web se situe à mi-chemin entre UP (unified process) et XP (unified process).

Le processus que nous allons présenter est:

- conduit par les cas d'utilisation, comme UP, mais beaucoup plus simple;
- relativement léger et restreint, comme XP, mais sans négliger les activités de modélisation en analyse et conception;
- fondé sur l'utilisation d'un sous ensemble nécessaire et suffisant du langage UML.

Dans un premier temps, les besoins vont être modélisés au moyen des cas d'utilisation UML. Il vont être représentés de façon plus concrète par une maquette d'IHM (interface homme machine) destinée à faire réagir les utilisateurs.

LA PROBLEMATIQUE : COMMENT PASSER DES BESOINS AU CODE



Expression initiale des besoins

(Phase UP inception ; synthèse du cahiers des charges)

Vision du projet

Collecter analyser, analyser et définir les besoins de haut niveau et les caractéristiques de la futur application web.

Positionnement

Pourquoi se projet.

Exigences fonctionnelles

Description par l'utilisateur (reprise du cahiers des charge) des contraintes fonctionnes.

Exigences non fonctionnelles

Exigences de qualité:
- Ergonomie
- Sur les formulaires
- Aide en ligne

Exigence de Performances:
- Volumétrie de la base
- Heure d'utilisation de l'application web

Exigence de sécurité:
- Confidentialité
- Intégrité
- Disponibilité

Contraintes de conception:
- Mise à jour des données de références:
- Mise à jour depuis les formulaires du site:
- Panier:
- Paiement sécurisé: La saisie du numéro de carte de crédit par le client doit s'effectuer de manière sécurisé, en cryptant le transfert HTTP, via le protocole SSL. La commande et le numéro de carte de crédit sont stockés dans la base, jusqu'au traitement de la commande. La banque concernée validera la transaction. A cette étape, le numéro de la carte de crédit ets supprimé de la base de données.

Réalisation d'une maquette des écrans du système:

Spécification des exigences d'après les cas d'utilisation (L'Etude Préliminaire)

L'expression préliminaire des besoins donnent lieu à une modélisation des cas d'utilisation et à une maquette d'IHM. Cette maquette est très importante, car elle va nous permettre d'identifier pendant l'analyse du domaine les concepts ou entités métiers. De plus à partir de cette maquette nous réaliserons des diagrammes d'activités UML pour modéliser la navigation.
Nous allons décrire ici comment modéliser les cas d'utilisation.
Nous allons nous intéresser à ce qu'il y a derrière la maquette:

- quelles sont les informations à montrer (modèle du domaine, concepts);
- à qui (les acteurs);
- pour quoi faire(les workflow, enchaînements).

Nous allons :

- identifier les acteurs;
- identification des cas d'utilisation;
- structurer les cas d'utilisation en packages
- ajouter les relations entre les cas d'utilisation;
- finaliser un ou plusieurs diagrammes de cas d'utilisation par package.

Le modèle ainsi obtenu permet d'établir les priorités entre les cas d'utilisation afin d'aider le chef de projet à planifier ses intégrations.

Identification des acteurs

Un acteur représente un rôle joué par une entité externe (utilisateur humain, dispositif matériel ou autre système) qui interagit directement avec le système étudié.
Un acteur peut consulter et/ou modifier directement l'état du système, en émettant et/ou en recevant des messages pouvant transporter des données.

Identification des cas d'utilisations

Un cas d'utilisation représente un ensemble de séquences d'action qui sont réalisées par le système et qui produise un résultat observable intéressant pour un acteur particulier.
Un cas d'utilisation modélise un service rendu par le système. Il exprime les interactions acteurs/système et apporte une valeur ajoutée notable à l'acteur concerné.
Une erreur fréquente concernant les cas d'utilisation consiste à vouloir descendre trop bas en granularité. En effet un cas d'utilisation représente un objectif métier de l'acteur.

Structuration en package

Pour améliorer le modèle, nous allons organiser les cas d'utilisation et les regrouper en ensembles fonctionnels cohérents. Nous allons utiliser le concept UML de « package ».
Le pattern le plus courant est de structurer en trois packages:

Acteurs;
UC des acteurs externe de l'entreprise;
UC des acteurs interne de l'entreprise.

Relation entre les cas d'utilisations

La relation entre un acteur et un cas d'utilisation et du type « association » (défini par UML).
Les relations entre cas d'utilisation sont:

« incluse », relation d'inclusion : le cas d'utilisation de base incorpore explicitement un autre, de façon obligatoire.(ex: la connexion au système)

« extend », relation d'extension : le cas d'utilisation de base incorpore implicitement un autre cas d'utilisation, de façon optionnelle. (ex: la connexion au système)

« généralisation/spécialisation », relation d'héritage : le cas d'utilisation descendant hérite de la description de leur parent commun.

Classement des cas d'utilisations

Après tout ce travail d'identification des cas d'utilisation, nous pouvons maintenant les hiérarchiser en tenant compte de deux facteurs suivants :

La priorité fonctionnelle, déterminée par la maîtrise d'ouvrage.
Le risque technique, déterminé par le chef de projet.

Nous pouvons fixer les niveaux de priorité à :

Haute;
Moyenne;
Basse;

Planification du projet en itérations

L'un des bon principe du Processus Unifier et de lever les risques majeurs au plus tôt. Le chef de projet doit donc prendre en comptes de façon combinée la priorité fonctionnelle et l'estimation du risque :

Si la priorité est haute et le risque également, il faut plannifier le cas d'utilisation dans une des toute premier itérations.

Si la priorité est basse et le risque également, on peut reporter le cas

d'utilisation dans les dernière itérations.
Traiter un cas d'utilisation risqué mais peut prioritaire, au lieu d'un cas d'utilisation plus prioritaire mais ne comportant aucun risques.

Analyse du domaine

(les objets métier)

Identification des concepts du domaine

Ajout des associations et des attributs

Généralisation

Structuration en packages de classes

Spécification détaillée des exigences (Début du cycle itératif)

Spécification détaillée des cas d'utilisations.

Mise à jour des diagrammes de cas d'utilisation

Diagrammes de séquence système

Réalisation des cas d'utilisation (Les Classes d'analyse)

Classes d'analyse participantes des cas d'utilisation

Typologie des classes d'analyse du pattern (MVC)

Les classes d'analyse se répartissent en 3 catégories:
Celles qui permettent les interactions entre l'application web et les utilisateur sont qualifiées de ***dialogue***. Ces classes proviennent directement des maquettes
Les classes qui contiennent la cinématique de l'application sont appelées ***contrôles***. Elles font la transition entre les dialogues et les concepts metier. Elles contiennes les règles de l'application.
Celles qui représentent les concepts métier sont qualifiées ***d'entités***. C'est celles que nous avons commencées à identifier dans les cas d'utilisation en classes du domaine.

Diagramme de classes participantes

C'est le point névralgique de notre démarche. Nous allons l'appeler diagramme de classes participantes.
C'est un diagramme de classes UML qui décrit, cas d'utilisation par cas d'utilisation, les trois principales classes d'analyse et leur relations.
Un avantage important de cette technique pour le chef de projet est de découper le travail de ses analystes suivant les cas d'utilisation.

- Les entités vont seulement posséder des attributs. Ces attributs représentent en général des informations persistantes de l'application.
- Les contrôles vont seulement posséder des opérations. Ces opération représentent la logique de l'application.
- Les dialogues possèdent des attributs et des opérations. Les attributs représentent des champs de saisies ou des résultats (attribut dérivé). Les opérations représentent les actions de l'utilisateur sur l'IHM.

Règles sur les associations entre les classes d'analyse

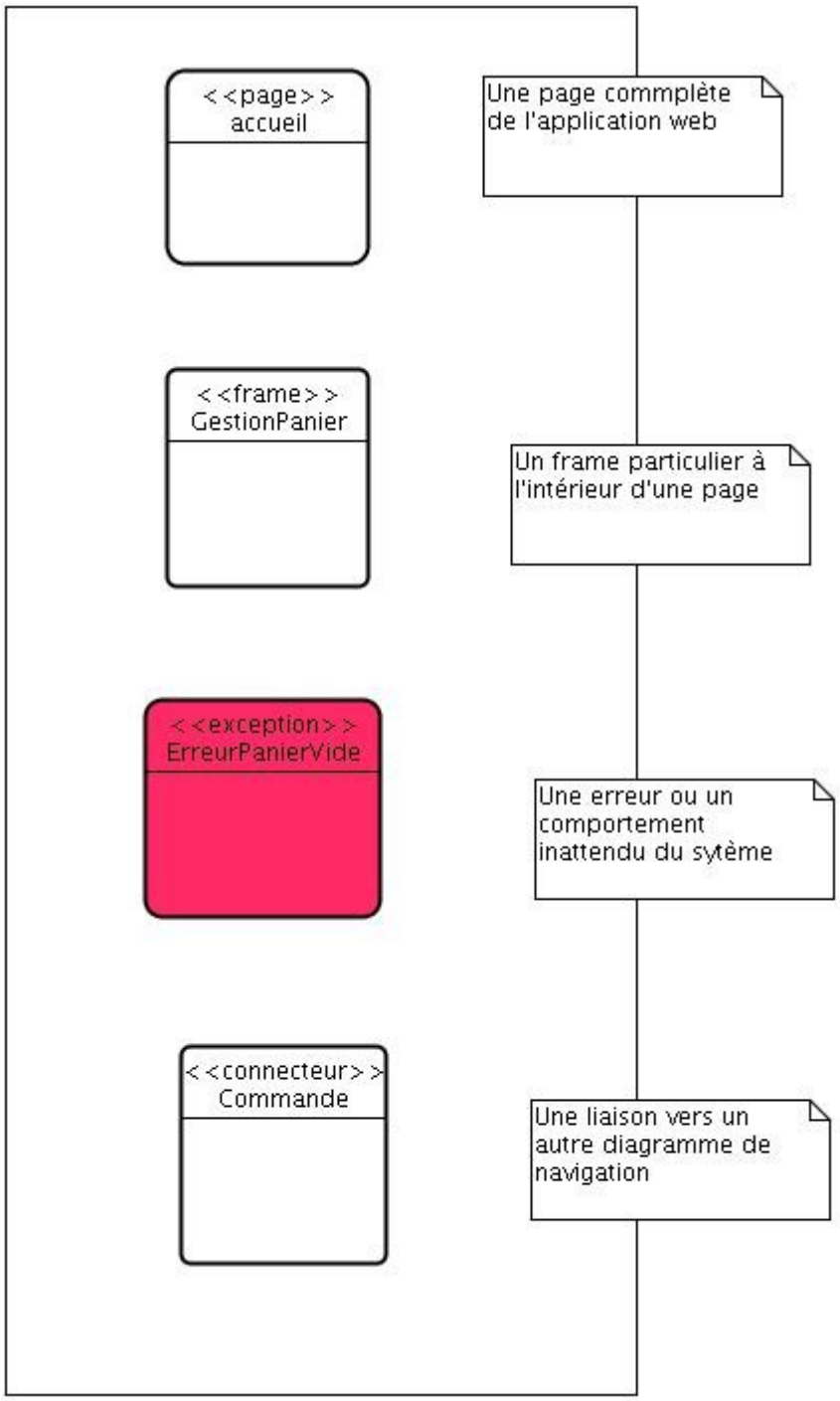
- Les dialogues ne peuvent être associés qu'aux contrôles ou à d'autre dialogues, mais pas directement aux entités.
- Les entités ne peuvent être associées qu'aux contrôles ou à d'autres entités.
- Les contrôles sont associés à tous type de classes, y compris d'autres contrôles.

On rajoutera les acteurs qui ne peuvent être liés qu'à un dialogues

Lorsque cela est nécessaire on peut décrire le comportement d'une entité par un diagramme d'état (ex : classe commande).

Modélisation de la navigation

Navigation dans l'application web peut être décrite soit par un diagramme d'activité soit par un diagramme d'état. Le choix dépend de l'outil utilisé ou de l'analyste.



Conception objet préliminaire

A présent nous allons ajouter les responsabilités précises de comportement aux classes d'analyse identifiées précédemment.
Nous allons utiliser le diagramme de séquences UML.

Classe de conception préliminaire

A partir des diagrammes de séquences de conception préliminaire, nous allons affiner les classes d'analyse pour donner les classes de conception préliminaires.

Architecture de l'application

(structuration en packages)

Pour structurer notre modèle, nous allons organiser les classes et les regrouper en ensemble cohérent. Pour cela nous allons utiliser les packages UML.

Nous allons classer nos classes dans trois couches:

- Une couche de présentation, rassemblant toutes les classes dialogues ;
- Une couche Logique Applicative, rassemblant toutes les classes de contrôles ;
- Une couche logique Métier, rassemblant toutes les classes métier ;

Conception objet détaillée

C'est la dernière étape avant l'implémentation.
On ajoute les classes techniques provenant des framework que l'on va utiliser sur le projet.