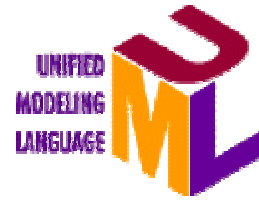


Abedrahim BIBA
Fabien GRABHERR
Eric LARRONDE-LARRETCHE
Yohann RICHARD



Modélisation

avec

UML

UML

Sommaire

I- : Introduction	p. 5
1-1 : UML par rapport au Génie Logiciel	p. 5
1-2 : Origine d'UML	p. 6
1-3 : Démarche d'application d'UML	p. 7
II- : Les diagrammes d'UML	p. 8
2-1 : Diagramme des cas d'utilisation (Use cases)	p. 9
2-2 : Diagramme des classes	p. 10
2-3 : Diagramme d'état-transition	p. 15
2-4 : Diagramme de séquence	p. 18
2-5- : Diagramme de collaboration	p. 20
2-6 : Diagramme des activités	p. 23
2-7 : Diagramme des composants	p. 25
2-8 : Diagramme de déploiement	p. 26
2-9 : Diagramme de paquetage	p. 27
2-10 : Redondance des diagrammes d'interactions (dynamique)	p. 28
2-11 : Résumé des différents diagrammes	p. 29
III- : Conclusion sur la méthode UML	p. 30
IV- : GD Pro en quelques mots	p. 32
V- : Annexe : Projet sur GD Pro	p. 33
Glossaire	p. 43
Bibliographie	p. 44

III- : Conclusion sur la méthode UML

Jusque là, les concepts utilisés pour modéliser les objets du domaine étaient différents de ceux utilisés pour spécifier les objets du système, les méthodes ne pouvaient pas grand chose pour rationaliser la démarche.

La généralisation de l'approche objet à l'ensemble du processus de développement a rendu possible l'unification du cadre conceptuel et UML a permis de supprimer la rupture méthodologique entre l'analyse et la conception.

Cette approche permet de passer du modèle au système de manière lisible et argumentée. Lisible, car, pour faire évoluer le système, il faut pouvoir faire le lien entre les objets du domaine et les objets du système ;

Argumentée car, les choix techniques sont complexes, interdépendants et doivent être périodiquement révisés pour tenir compte de l'évolution des environnements.

Si UML s'est aussi rapidement imposé à tous, c'est parce qu'il ne s'agit que d'un langage, dont l'efficacité n'est liée ni à la nature du problème ni à une démarche méthodologique particulière.

Pour autant, la mise en œuvre d'UML suppose une définition des tâches et des procédures adaptées au contexte, et en particulier de préciser :

→ La nature des problèmes

→ Les technologies et outils de développement utilisés

→ La culture méthodologique des participants, l'expérience des problèmes, les technologies des outils

→ La dimension de l'entreprise et modes d'organisation

UML est aujourd'hui un standard incontournable. Les raisons de son succès sont multiples :

→ Il est le résultat d'un large consensus (industriels, méthodologistes...).

→ Il couvre toutes les phases d'un cycle de développement.

→ Les outils qui supportent UML se multiplient (GDPro, ObjectTeam, Objecteering, OpenTool, Rational Rose, Rhapsody, STP, Visio, Visual Modeler, WithClass...).

UML n'est pas une méthode ou un processus : il a été pensé pour permettre de modéliser les activités de l'entreprise, pas pour les régir (ce n'est pas CMM ou SPICE).

Un processus de développement logiciel universel est une utopie. En effet, il est impossible de prendre en compte toutes les organisations et cultures d'entreprises. Même si un processus constitue un cadre général, il faut l'adapter de manière précise au contexte de l'entreprise.

UML est un langage formel : il est fondé sur un métamodèle, qui définit :

les éléments de modélisation (les concepts manipulés par le langage),

la sémantique de ces éléments (leur définition et le sens de leur utilisation).

Un métamodèle est une description très formelle de tous les concepts d'un langage. Il limite les ambiguïtés et encourage la construction d'outils.

Le métamodèle d'UML permet de classer les concepts du langage (selon leur niveau d'abstraction ou domaine d'application) et expose sa structure.

UML cadre l'analyse objet, en offrant :

- différentes vues (perspectives) complémentaires d'un système, qui guident l'utilisation des concepts objets,
- plusieurs niveaux d'abstraction, qui permettent de mieux contrôler la complexité dans l'expression des solutions objets.

UML est un support de communication

- Sa notation graphique permet d'exprimer visuellement une solution objet.
- L'aspect formel de sa notation limite les ambiguïtés et les incompréhensions.
- Son aspect visuel facilite la comparaison et l'évaluation de solutions.

Son indépendance (par rapport aux langages d'implémentation, domaine d'application, processus...) en font un langage universel.

Les points forts d'UML :

- UML est un langage formel et normalisé, il permet un gain de précision et un gage de stabilité. Ce qui encourage l'utilisation d'outils.
- UML est un support de communication performant, il cadre l'analyse et il facilite la compréhension de représentations abstraites complexes.
- Son caractère polyvalent et sa souplesse en font un langage universel.

Les points faibles d'UML

- La mise en pratique d'UML nécessite un apprentissage et passe par une période d'adaptation.
- UML n'est pas à l'origine des concepts objets, mais en constitue une étape majeure, car il unifie les différentes approches et en donne une définition plus formelle.
- Le processus (non couvert par UML) est une autre clé de la réussite d'un projet. Or, l'intégration d'UML dans un processus n'est pas triviale et améliorer un processus est une tâche complexe et longue.

IV- : GD Pro en quelques mots

Gdpro dans la presse :



"Notation unifiée, AGL en rivalité" Décembre 1999 n°16 p.6

"GDPro est l'un des sérieux challengers UML, créé par ASTI et distribué en France par Software & Process. Déjà plébiscité dans nos précédents dossiers, cet AGL pousse à fond les possibilités d'UML(...). GDPro produit du code Java, C++ ou IDL (applications CORBA) et l'AGL est également un outil de rétro-ingénierie très performant(...). On apprécie enfin la possibilité de travail d'équipe avancé, avec mise à jour des modifications par le Web facilitant le développement coopératif de plusieurs équipes géographiquement séparées".



"La rétro-ingénierie sans peine" 10 Septembre 1999 n°1555 p.54

"Il rivalise déjà avec les ténors du marché de l'orienté objet, tel Rational et son atelier Rose (...). GDPro possède toutes les fonctions d'un AGL haut de gamme (...). GDPro se distingue surtout par ses fonctions de rétro-ingénierie, permettant de s'assurer aisément de la cohérence du code et des modèles UML".



"Software & Process commercialise GDPro" 9 juillet 1999 n°818 p.47

"Software & Process annonce la commercialisation de GDPro, le logiciel de modélisation et conception UML d'ASTI. Le code source est généré automatiquement en C++, Java ou IDL à partir d'un modèle UML (...). GDPro extrait aussi les diagrammes à partir d'un code source, même de plusieurs millions de lignes".



"GDPro 3.1 : A Clear Contender Among Visual Modelers" Octobre 1999

"For all its potential, the Unified Modeling Language (UML) needs sophisticated tool support to enable next-generation design. A promising tool for UML enablement is GDPro 3.1 - a visual modeling tool offered by ASTI to support software design, development, redesign and maintenance for small- and large-sized teams alike (...). GDPro is the first and only visual modeling tool that support all eight diagrams and even manages to extend the language. This is something that even Rational Software cannot claim about its Rose 98 offering (...). Strong contender ASTI has done application developers a great service with GDPro".

V- : Annexe : Projet sur GD Pro

Plan de l'annexe :

Présentation

1- : Diagramme des cas d'utilisations :

2-1 : Diagramme d'activité :

2-2 : Diagramme d'activité (simplifié) du service adhérent :

3- : Diagramme des classes :

4-1 : Diagramme de séquence 1 (Scénario n°1) :

4-2 : Diagramme de séquence n°2 :

4-3 : Diagramme de séquence n°3 :

5- : Diagramme d'état de la classe serveur :

6- : Diagramme de collaboration :

Bilan

Présentation :

Nous avons décidé de modéliser le fonctionnement d'un moteur de recherche offrant un certain nombre de services (e mail, comparatifs de prix par produit et par localité, recherches personnalisées, ...).

Ce moteur de recherche fonctionne à partir d'une base de données «*Einstein*» qui contient des informations sur les adhérents et d'autres informations nécessaires au fonctionnement des services.

Nous n'avons pas traité tous les diagrammes en détail mais, pour chaque diagramme de la méthode, nous nous sommes intéressés à une partie spécifique du moteur de recherche.

Le principe est le suivant :

- Un utilisateur se connecte et on lui offre la possibilité d'adhérer à nos services.
- L'adhérent a alors accès à nos services (e mail, page web, ...) ainsi qu'à une recherche personnalisée en fonction de ses préférences et de ses recherches les plus fréquentes. Ainsi, il peut être prévenu par mail, à la suite d'une recherche, des nouveaux résultats pertinents.
- Les entreprises désirant participer à nos services (comparatifs de prix, publicité) accèdent également à la base Einstein pour y inscrire des données.

Glossaire

OMG : Object Management Group

AGL: Atelier de Génie Logiciel

Abstraction : Démarche qui consiste à ne considérer que certains éléments, pour des raisons de pertinence et/ou de dépendance.

Classe : Description d'un ensemble d'objets. Au sens strict, une classe décrit (totalement ou partiellement) une implémentation. On parlera de type pour une description limitée à l'interface (attribut et opération), indépendante de toute implémentation.

Etat : Représente la situation d'une activité ou d'un objet. Les états (de l'une ou de l'autre) doivent être significatifs par rapport à un comportement ou à une collaboration. Leur nombre doit être défini.

Message : Mécanisme par lequel un objet communique avec un autre. Un message est supposé provoquer l'exécution d'une opération par l'objet destinataire. Il faut distinguer :

- le message et l'opération : un même message peut déclencher des opérations différentes.
- Le message et la réponse : un événement est associé à la réception d'un message. Si la réponse est instantanée, il n'y a pas de message associé à la réponse.

Méthode : Implémentation d'une opération.

Node (node) : Ressource physique supportant l'exécution, la persistance ou la transmission des composants.

Objet : Elément identifiable caractérisé par les états qu'il peut prendre et par les opérations qu'il peut réaliser. En termes d'implémentation, la notion d'objet couvre l'ensemble des éléments physiques identifiables par le système, ce qui peut inclure les classes.

OMT : Object Modelling Techniques

Processus (process) : Ensemble des activités générées par un événement externe et capables de s'exécuter indépendamment, c'est à dire sur leurs seules ressources. Correspond à l'exécution d'une collaboration.

Template : Mécanisme qui permet de définir une méthode sans avoir à définir le type de ses arguments. Uniquement implémenté par le langage C++.

Bibliographie

<u>UML Principes de modélisation</u>	Rémy Fannader, Hervé Leroux	ED. DUNOD
<u>MERISE vers OMT et UML</u>	Joseph Gabay	InterEditions
<u>Penser objet avec UML et Java</u>	Michel Lai	InterEditions
<u>Conception et modélisation des systèmes temps réel</u>	Bui Minh Duc	ED. Eyrolles

Liens :

www.softteam.fr (Objecteering)

www.rational.com (Rose)

www.advancedsw.com (GD Pro)