AFLI

ASSOCIATION FRANCAISE DE GENIE LOGICIEL

DEFINITIONS DE BASE DE L'APPROCHE OBJET

1995 - VERSION 1

avec la participation du Syntec Informatique

ISSN 1258-8474

DEFINITIONS

DE BASE

DE L'APPROCHE OBJET

ADELI - Commission «Orienté Objet»

avec la participation du Syntec Informatique

1995 - VERSION 1

SOMMAIRE

PREAMBULE

Aujourd'hui, dans le monde de la gestion, l'Approche Orientée Objet (A.O.O.) est devenue à la fois un concept et une démarche incontournables pour tous les responsables de projets.

Le développement de logiciel représente un secteur important de notre économie qui s'apprête à passer du monde artisanal au monde industriel. L'évolution des systèmes matériels et logiciels a permis de rendre disponible les technologies Orientées Objet dont les premières études remontent à plus de trente ans. Cette nouvelle façon de concevoir des systèmes informatiques, basée sur la définition de modèles concrets, favorise le dialogue entre les différents acteurs du Système d'Information (Managers, Financiers, Utilisateurs et Développeurs). « L'Objet », après avoir fait ses premières armes et atteint une bonne maturité dans les domaines du développement applicatif (IHM, programmation, bases de données, communication, etc.), investit maintenant tous les niveaux de l'analyse et de la conception.

La nouveauté de l'Objet, le manque de normes et l'absence d'un langage commun n'en facilitent évidemment pas l'accès aux Maîtres d'Ouvrage, aux Maîtres d'oeuvre et à leurs équipes. Pas plus, d'autre part, que chaque publication de la nombreuse littérature, qui y va de son petit lexique plus ou moins personnalisé. Que de temps et d'énergie perdus, bien souvent, à comparer des termes, des définitions ou des concepts figurant dans des ouvrages différents.

Dans ce contexte, le présent document a pour ambition, avant tout, de fournir un répertoire des termes et un dictionnaire des définitions les plus couramment admises.

Il a toujours été hors de notre propos de faire :

- un cours sur la technologie Objet. Il existe suffisamment de publications disponibles sur le marché.
- un inventaire des particularités ou des définitions trop étroitement liées à des démarches ou à des solutions méthodologiques trop spécifiques dans cette technologie émergente.

On dénombre aujourd'hui sur le marché, beaucoup de méthodes de conception orientées objet. Cette diversité tient plus du positionnement concurrentiel des acteurs économiques que d'une très grande divesité de l'offre en matière de méthode objet. S'il existe bien sûr des différences notables entre les méthodes, leurs premières mises en oeuvre, que l'on peut observer dans les entreprises, font apparaître des notions fondamentales communes. Ce sont ces dernières qui constituent le centre d'intérêt du présent document.

Il nous semble désormais possible et utile de dresser un glossaire des DEFINITIONS DE BASE DE L'APPROCHE OBJET appliquée à l'analyse et à la conception des systèmes d'information. Ce premier travail devant constituer le premier jalon d'une nouvelle charte méthodologique.

Ont participé à l'élaboration du présent document :

• Paul-André BRES	CONCIS.
• Michel DEMONFAUCON	AHIMSA.
• Jean-Paul EYBERT	SYNTEC INFORMATIQUE.
• Cuong KHAMPRASONG	
• Daniel J. MARTINS	SG2.
• Jean-Pierre PFISTER	SNCF.

I - PRESENTATION

Ce document, destiné à toutes les personnes intéressées par la technologie Objet, se veut d'un usage pratique, clair et concis. Les différents termes et concepts ont été généralisés pour s'extraire du cadre particulier des méthodes. La démarche Orientée Objet provient à l'origine du monde industriel au travers des langages. Il faut noter que dans le présent document, nous avons circonscrit le domaine aux méthodes de conception des systèmes d'information de gestion uniquement.

I.1 - Objectif

La liste des termes que nous avons retenus constitue le thesaurus minimum que chacun doit posséder pour aborder efficacement les méthodologies Orientées Objet proposées sur le marché.

Son objectif est double :

- permettre aux débutants de se focaliser sur les points essentiels de l'Approche Orientée Objet appliquée aux méthodes de conception de systèmes d'information,

- offrir un vocabulaire de référence aux professionnels expérimentés sur une matière en cours de maturation.

I.2 - Démarche

Les concepts et termes abordés dans ce document ont été identifiés lors de plusieurs séances de travail. Ils sont le résultat d'une démarche commune de réflexion entre des acteurs représentatifs du marché.

Encore une fois, toutes les notions contenues dans les différentes méthodes objet ne sont pas forcément reprises dans le présent document. On constate notamment que la dimension statique est particulièrement approfondie au détriment sans doute des autres dimensions dynamique ou fonctionnelle; où les notions fondamentales ne font pas l'unanimité, loin s'en faut.

I.2.1 - Contexte d'utilisation

Ce document aura atteint son objectif s'il permet au lecteur de comprendre un terme ou un concept qu'il aura découvert dans une revue, dans une méthode ou dans un document technique. Il peut bien sûr susciter aussi des réactions critiques concernant une définition ou la sélection des termes retenus.

Le lecteur souhaitant contribuer à l'évolution du document, est invité à contacter les rédacteurs afin de formuler ses propositions.

I.2.2 - Modalités d'utilisation

Le présent document peut être utilisé de plusieurs manières avec plusieurs points d'entrée :

- les principes objet (chapitre II.1);
- le dictionnaire (chapitre III.1) des définitions les plus couramment admises des termes classés par ordre alphabétique;
- la liste des termes de base (chapitre II.2);
- le tableau de correspondance des synonymies trié par ordre alphabétique (chapitre III.2);
- l'exemple d'illustration (chapitre IV).

Les termes de base de l'Orienté Objet commencent par une majuscule.

II - LES CONCEPTS OBJETS

II.1 - Principes

Dans l'environnement économique extrêmement changeant d'aujourd'hui, la prospérité de nos entreprises dépend de ses informations, dont la qualité doit permettre de prendre les meilleures décisions. Cette prise de décision doit être rapide tout en restant compétitive. La qualité des informations, leur disponibilité et leur précision influeront sur le succès ou sur l'échec des objectifs.

La nécessité impérative d'une information de meilleure qualité, l'explosion des besoins nouveaux découlant de l'ouverture des Systèmes d'Information (S.I.) ne laissent d'autres choix que de rechercher de nouvelles méthodes, beaucoup plus efficaces, pour la prise en charge des applications. L'Approche Orientee Objet (A.O.O.) est aujourd'hui l'une des pierres angulaires de l'évolution à laquelle l'entreprise est contrainte. Elle conduit tout naturellement à une vision différente de la vision procédurale et plus proche de la réalité des métiers.

L'approche Orientée Objet permet de créer plus rapidement des applications, sous une forme où la maintenance est facilitée. Elle apporte une solution à certaines préoccupations majeures des responsables des systèmes d'information :

- comment parvenir à gérer des systèmes de plus en plus complexes ?
- comment industrialiser la démarche informatique qui envahit tous les secteurs d'activité de l'entreprise ?
- comment améliorer la réactivité du système d'information ?
- comment améliorer les coûts de maintenance dans un contexte très évolutif et de plus en plus concurrentiel ?

La maîtrise de l'Orienté Objet conduit à développer plus rapidement des applications mieux conçues. Son efficacité inhérente améliore la productivité et réduit considérablement les coûts de maintenance. Des économies sont effectivement prévisibles à long terme, la vocation de l'Objet résidant dans sa capacité à s'ajuster aux situations nouvelles.

Les techniques conventionnelles limitent la liberté d'action, notamment en ce qui concerne les procédures et les structures de données. Mais avec cette nouvelle démarche, l'accent est mis sur des Objets : ceux-ci contiennent à la fois des Attributs (données) et des Méthodes (traitements) qui doivent régir ces Attributs. Cette approche nouvelle simplifie la conception et le développement des applications parce qu'elle reflète plus précisément notre façon de penser dans le monde réel.

Nous pensons naturellement en termes d'Objets, à leurs Propriétés et à leur comportement. Un véhicule routier a, par exemple, des Attributs (marque, modèle, âge,) et des Méthodes (accélération, freinage, braquage,...), l'ensemble rendant un Service (conduire de ... à ...).
Nous tendons à Classifier les Objets pour mieux maîtriser la complexité du monde réel : la classification scientifique du mode vivant par exemple.

Il est possible de créer un modèle de classification (Classes) en utilisant une hiérarchisation des différentes catégories d'Objets. Ce dispositif accélère le processus de conception et le rend plus lisible et aussi plus cohérent.

Un Objet ne deviendra-t-il pas un module de codes parfaitement défini possédant son interface propre activée par des Messages auxquels il doit éventuellement répondre. En effet, les Objets communiquent entre eux en s'envoyant des Messages : chaque Objet sait exactement comment répondre à chaque Message, et différentes Classes d'Objets pourront avoir des réponses différentes.

La mise en oeuvre de chaque Objet est transparente au reste du système : Encapsulation des Attributs et des Méthodes. Ceci signifie qu'à partir du moment où un Objet répond correctement aux Messages qu'il reçoit, il est tout à fait possible de modifier le rôle qui lui est attribué sans rien changer par ailleurs (modularité et Polymorphisme)

Il faut toutefois noter que les Objets conservent leur structure et leur comportement à travers la hiérarchie des différentes Classes. Ceci signifie qu'il suffira d'effectuer toute modification en un seul endroit ; elle sera automatiquement propagée : c'est le principe de l'Héritage. Par exemple, si nous affectons le nouvel Attribut propriétaire à la Classe Véhicule, toutes les Sous-Classes de véhicules (trains, avions, bateaux,...) Hériterons de ce nouvel Attribut. Le mécanisme de l'Agrégation le rend plus proche du monde réel lors des opérations de modélisation. L'opérateur d'Héritage offre à l'analyste de riches dispositions pour appréhender le travail de modélisation (i.e. définition des Classes) du problème auquel il est confronté.

En effet, le mécanisme d'Héritage permet, pour un Objet, de séparer les Propriétés qu'il peut considérer comme spécifiques au contexte d'utilisation de l'Objet de celles qui sont plus générales et qui seraient donc susceptibles d'être appliquées à d'autres Objets dans d'autres contextes. L'analyste généralisera ou « les mettra donc en facteur » dans la définition d'une Sur-Classe. En revanche, si l'analyste découvre des spécificités particulières d'un sous-ensemble d'Objets d'une Classe, il sera amené à spécialiser ces caractéristiques dans la définition d'une Sous-classe de manière à éviter une « pollution » de la Classe généralisée.

En conséquence, un Objet qui sera considéré comme une Instance d'une seule Classe, sera également considéré comme élément de plusieurs Classes (à savoir les Sur-Classes de la Classe d'Instance).
Ainsi, un Objet pourra être Classifié dans plusieurs Classes. Par exemple, M. Dupont pourra être Classifié en tant qu'« Homme », en tant qu'« Employé », en tant que « Personne » et en tant que « Responsable »....

Les avantages majeurs de l'Orienté Objet que sont la localisation aisée et la propagation des modifications, rendent plus aisées la maintenance et l'évolution des applications, tout en étant également moins coûteuses.

La modularité des applications Orientées Objet induit la possibilité de réutilisation de Services, de codes, puisque les Objets deviennent des briques de construction à partir desquelles il est possible de faire évoluer des applications existantes ou d'en créer de nouvelles.

Ainsi, au fur et à mesure de l'avancement d'un projet, des Objets ou des portions entières de la hiérarchie des Classes peuvent être réutilisés au sein de la même application, ou au sein d'un autre projet. Cela signifie que les développeurs peuvent partager et échanger des informations, du code dans des proportions jusque là inconnues et même impossibles. Le cannibalisme industriel fait enfin son entrée dans le monde de la gestion.
Il devient ainsi plus aisé de construire et de maîtriser des systèmes complexes, et les développeurs peuvent produire des solutions d'une qualité plus élevée.

La création de nouvelles applications à partir de composants existants générera d'importantes économies dans la production applicative. Les bénéfices sont induits par un raccourcissement du temps de développement, par l'accroissement de la productivité et par la maîtrise d'une meilleure qualité.

II.2 - Liste des termes de base

*DEFINITIONS DES TERMES DE BASE
*ABSTRACTION
*AGREGATION
*APPROCHE BASEE OBJET
*APPROCHE ORIENTEE OBJET
*ATTRIBUT
*CAPSULE
*CLASSE
*CLASSIFICATION
*CONTRAINTE
*CORPS
*CYCLE DE VIE DE L'OBJET
*DELEGATION
*DOMAINE
*ENCAPSULATION
*ENSEMBLE D'OBJETS APPLICATIFS
*ETAT D'UN OBJET
*EVENEMENT
*EVENEMENTIEL
*EXTENSIBILITE
*GENERALISATION
*HERITAGE SIMPLE
*HERITAGE MULTIPLE
*HERITAGE ASCENDANT
*HERITAGE DESCENDANT
*HERITAGE EN REFERENCE
*IDENTIFICATEUR D'OBJET
*INSTANCE
*INSTANCIATION
*MESSAGE
*METHODE
*MODELE DYNAMIQUE
*MODELE FONCTIONNEL
*MODELE STATIQUE

*OBJET
*OBJET GENERIQUE
*OBJET METIER
*OBJET TECHNIQUE
*OBJET UTILE
*PERSISTANCE
*POLYMORPHISME
*PRINCIPE DE LOCALITE
*PRIVE
*PROPRIETE
*PUBLIC
*REDEFINITION
*RESOLUTION DYNAMIQUE
*RESOLUTION STATIQUE
*REUTILISATION
*SERVICE
*SIGNATURE
*SOUS-CLASSE
*SUR-CLASSE
*TYPE

III - GLOSSAIRE

III.1 - Dictionnaire

ABSTRACTION

L'Abstraction est une action de l'esprit qui, à partir d'un Objet donné, en extrait une signification utile, et construit une idée ou un concept manipulable intellectuellement. L'Abstraction fait donc ressortir les caractéristiques essentielles d'un Objet (ce qui le distingue des autres) : c'est une mécanique d'interfaçage au niveau de la Classe permettant de traiter la complexité en concentrant les similitudes et en ignorant les différences.
Les différents types d'Abstraction sont :

- Abstraction par simplification (idéation).
- Abstraction par généralisation (conceptualisation).
- Abstraction par sélection (classification).
- Abstraction par schématisation (modélisation).

AGREGATION

L'Agrégation est un opérateur qui permet de regrouper dans un même ensemble (que l'on une appelle une Classe) des éléments divers et variés. Cela peut être une donnée, un traitement, une image, un son ou encore un sous-ensemble constitué lui-même d'éléments. L'Agrégation couvre les aspects multi-média d'un Objet. Il s'agit d'une association de composé à composant (que l'on retrouve par exemple dans les problèmes de nomenclature). L'ensemble composite peut avoir des Propriétés différentes de celles des éléments qui le composent. Le mécanisme d'Agrégation a pour fonction construire des représentations informatiques les plus proches possibles des objets réels manipulés par l'utilisateur (ou plus exactement, de l'image qu'il se fait des objets qu'il manipule).

APPROCHE BASEE OBJET

« L'Approche Basée Objet » concerne tout type de développement, de programmation mettant en oeuvre l'assemblage de modules développés en technique Objet et possédant une dynamique « Objet ». Il n'y a pas de développement Objet à proprement parler, encore moins de modélisation Objet; il s'agit d'un assemblage de codes, de modules existants pour réaliser une fonction spécifique.

APPROCHE ORIENTEE OBJET

L'Approche Orientée Objet est un courant technique et méthodologique qui consiste à utiliser les concepts Objets de Classe, d'Encapsulation, d'Héritage, de Message, de Modularité, de Polymorphisme... et ce dans plusieurs domaines. Elle permet une démarche de type industriel dans la conception et le développement de logiciels, processus défini par ses propriétés majeures

- standardisation des éléments,
- assemblage des éléments en composants (ou Objets),
- réutilisation possible des composants,

d'où, pour les applicatifs développés :

- une grande fiabilité,
- des temps d'assemblage réduits,
- une maintenance facilitée,
- un abaissement des coûts de fabrication à terme.

ATTRIBUT

Caractéristique ou donnée élémentaire de la Classe qui décrit la structure de l'information. C'est une composante élémentaire de la Classe. L'Attribut peut être de Type Classe et invisible de l'extérieur (Privé), ou être visible de l'utilisateur (Public).

CAPSULE

Enrobage d'un programme ou d'une séquence d'instructions traditionnelles sur une structure de données pour lui donner un comportement Objet. C'est une technique utilisée pour faire coopérer des applicatifs anciens (écrits en COBOL par exemple) avec des développements utilisant des environnements Objet.

CLASSE

C'est un ensemble d'Objets ayant des propriétés similaires et des comportements identiques ; ils sont dits d'un même Type. Un Objet est une occurrence de sa Classe (Instance). Une Classe est définie par des Attributs et par des traitements invisibles de l'extérieur (Propriétés Privées), ainsi que par des Attributs et des traitements visibles à l'extérieur sous forme de Services offerts (Méthodes Publiques).

CLASSIFICATION

La Classification permet de définir des classes puis de ranger les Objets dedans. Il s'agit d'un mécanisme d'abstraction de regroupement par généralisation/spécialisation des Objets. Ces classes sont non forcément disjointes, c'est-à-dire qu'un Objet peut « être rangé » dans plusieurs classes. Une Classe peut elle-même être incluse dans des Sur-Classes ; la Classe hérite alors des Propriétés de cette Sur-Classe.

CONTRAINTE

Elles sont de différentes natures :

- les Contraintes de domaine : les Attributs prennent leurs valeurs dans un ensemble défini de valeurs,
- les Contraintes temporelles : un ordre temporel est respecté pour la manipulation des différents Objets suivis dans la base d'information,
- les Contraintes référentielles : elles sont destinées à vérifier l'existence des Objets mis en jeu dans la base d'information,
- les Contraintes sur des agrégats : ici, sont concernés les agrégats d'Attributs.
- les Contraintes entre Attributs : les Attributs doivent respecter différentes contraintes entre eux.

Elles sont supportées par des Méthodes.

CORPS

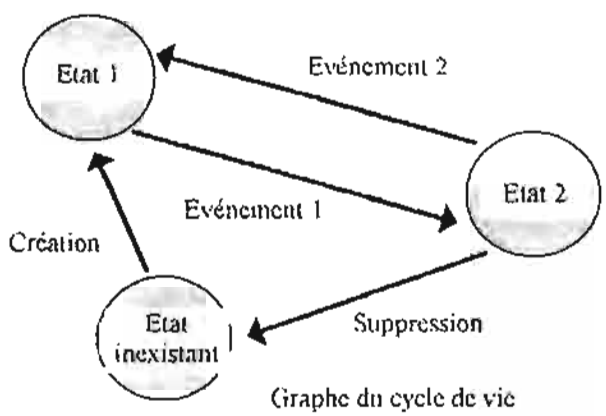
Le Corps de la Méthode correspond à l'ensemble des instructions qui traduisent cette Méthode et permettent de faire exécuter le traitement correspondant.

CYCLE DE VIE DE L'OBJET

C'est l'ensemble des Etats provoqués par des Evénements que peut prendre un Objet dans le temps. Un Evénement est une action définie sur un Objet qui engendre un changement d'Etat de l'Objet.

Le graphe du Cycle de vie reprend :

- l'ensemble des Etats de l'Objet,
- l'ensemble des Evénements qui provoquent des changements d'Etat,
- l'ensemble des Contraintes qui réglementent les changements d'Etat intervenus à la suite d'Evénements.



Graphe du cycle de vie

DELEGATION

Possibilité d'adresser directement un Objet à un autre Objet sans passer par l'Héritage. La Délégation se substitue à l'Héritage multiple.

DOMAINE

Regroupement de Classes appartenant à un même ensemble logique du point de vue métier ou technique.

ENCAPSULATION

C'est, à l'intérieur d'un même concept Objet, la suppression de la dualité entre les données (Attributs) et les traitements locaux (Méthodes), tout en les masquant à l'utilisateur (Propriétés Privées). Elle agit comme une boîte noire avec des Messages, mais sans tuple, sans Attribut, sans Méthode.

ENSEMBLE D'OBJETS APPLICATIFS (« Frameworks » ou sous-ensemble applicatif)

Ensemble ou sous-ensemble d'Objets, parfaitement documentés, proposés à la vente par des éditeurs, SSII, sociétés, etc... sur le marché. Ces ensembles ou sous-ensembles sont de véritables composants complets permettant, de plus en plus souvent, de réutiliser des fonctions déjà réalisées par d'autres concepteurs. Il s'agit d'Objets techniques ou métiers, véritables pièces de « LEGO » autorisant des gains de temps importants dans la conception d'applicatifs.

ETAT D'UN OBJET

L'Etat d'un Objet est caractérisé par le contenu de ses données (Attributs) et de ses références à un moment donné.

EVENEMENT

C'est un stimulus d'un Objet vers un autre et un fait survenu sur un Objet qui engendre un changement d'Etat de l'Objet.
Il sert à caractériser le comportement des Objets, il est complémentaire des Attributs. Dans cette perspective un Objet est décrit et représenté par des valeurs d'Attributs et des Occurrences d'Evénements. L'approche des modèles sémantiques a permis de donner aux Objets une dimension dans l'espace : un Objet peut être composé d'autres Objets. La notion d'événement permet de donner aux Objets une dimension temporelle.

EVENEMENTIEL

Logique Objet réalisée dans les applications, où l'action est associée à un événement. Le seul type d'événement qui peut se produire, est l'envoi de Message d'un acteur Objet vers un autre dont on attend un Service.

EXTENSIBILITE

Possibilité d'étendre à volonté l'ensemble des Classes de base fournies ou les Types prédéfinis (dates, caractères, lignes, surfaces,...). Toute Classe est elle-même un Objet qui appartient à une Classe générique plus générale appelée Métaclasse.

GENERALISATION

Propriété entre les Types de deux Classes différentes transmettant à une Classe, les Propriétés d'une autre Classe. C'est une des caractéristiques de l'Héritage.

HERITAGE SIMPLE

L'Héritage, notion fondamentale de l'Orienté Objet, permet à un Objet d'hériter des données (Attributs) et des traitements (Méthodes) d'un autre Objet. Il ouvre la possibilité à des Objets de structures différentes de partager des opérations attachées à leur partie commune. Il autorise également la définition de nouvelles Classes à partir des Classes existantes en mettant en facteur ces parties communes (Attributs et Méthodes), d'où la création de Sur et de Sous-Classes. Il est possible d'enrichir un concept par l'adjonction d'Attributs et de Méthodes nouveaux.
L'Héritage est dit Simple quand il n'hérite que d'une seule Classe. C'est un enrichissement progressif du modèle d'Objet où les extensions des Classes sont emboîtées les unes dans les autres.

HERITAGE MULTIPLE

L'Héritage est dit multiple quand une Classe hérite de plusieurs Classes simultanément. Ce type d'Héritage est plus délicat à gérer et peut être une source d'ambiguïtés et de dérives dans les raisonnements. Il augmente néanmoins le pouvoir de réutilisation.

HERITAGE ASCENDANT

mécanisme de « remontée » des Propriétés des Sous-Types dans le Sur-Type, avec éventuellement un nouvel Attribut d'identification du Sous-Type.

HERITAGE DESCENDANT

Mécanisme de «descente » des Propriétés du Sur-Type dans les Sous-Types.

HERITAGE EN RELATION

Mécanisme de considération séparée des propriétés suivant le découpage Sous-Types/Sur-Type ; le lien sémantique n'étant implanté que par référence (pointeur ou clé étrangère).

IDENTIFICATEUR D'OBJET

L'Identificateur donne l'identité d'un Objet en le désignant de façon unique. L'Identificateur permet de relier deux ou plusieurs Objets indépendamment de leurs valeurs. Deux Objets sont égaux s'ils ont la même valeur, et ils sont identiques s'il s'agit du même Objet. Cette identité facilite la notion de partage en utilisant des références de différents endroits vers le même Objet, au lieu de générer des copies, comme dans certains S.G.B.D. non Orientés Objet.

INSTANCE

On appelle Instance une occurrence d'une Classe. Un Objet est donc une instance de Classe. Cette Instance est désignée par son Identificateur. L'Objet identifié est seul et unique.

INSTANCIATION

Mécanisme de constitution des instances d'un Objet. Par extension, ensemble des instances d'une Classe.

MESSAGE

Mode de communication ou de dialogue entre des Objets de Classes différentes. A l'aide d'un Message un Objet demandeur d'une Classe sollicite les Services d'un Objet receveur d'une autre Classe. Le Message est la caractéristique du comportement de l'Objet. Il fournit la formule d'invocation des traitements qui autorise notamment le polymorphisme.

METHODE

C'est un traitement ou une fonction qui manipule des Attributs et c'est la composante dynamique de l'Encapsulation des Objets dans une Classe. La Méthode est un moyen d'activer une des fonctionnalités attachée à un Objet donné ; de l'extérieur, elle est soit visible (Service Public), soit invisible (Service Privé).
Les Méthodes sont divisées en deux parties :

- l'Entête, ou Signature est aussi appelé Spécification, seule partie visible par l'utilisateur.
- le Corps ou Implantation.

MODELE DYNAMIQUE

La conception/programmation Orientée Objet repose sur 3 axes. Le Modèle Dynamique est le « QUAND », l'axe TEMPOREL des Evénements. Il décrit les contrôles d'un système avec leur séquencement et les Evenements temporels des interactions entre Objets.

MODELE FONCTIONNEL

La conception/programmation Orientée Objet repose sur 3 axes. Le Modèle Fonctionnel est le « QUOI » de la programmation Objet. Il décrit les aspects du système qui transforme les valeurs en utilisant des fonctions, des représentations, des contraintes et des dépendances fonctionnelles.

MODELE STATIQUE

La conception/programmation Orientée Objet repose sur 3 axes. Le Modèle Statique est le « DANS QUOI » de la conception Objet. Il décrit la structure des Objets, l'Identité, les opérateurs ainsi que les Attributs et les Méthodes qui sont appliquées. C'est l'expression de l'arbre de Classe et de leurs structures

OBJET

Un Objet est un élément ayant une signification précise et complète. Un Objet a une identité, un contour net significatif dans le contexte où il existe ; il est capable de comportements spécifiques et peut interagir avec son environnement en fonction des sollicitations qu'il reçoit et des informations qu'il contient. C'est l'Identificateur qui le différencie des autres Objets.

OBJET GENERIQUE

Objet remarquable permettant de mettre en facteur des propriétés techniques ou transverses pouvant et devant être réutilisées par les concepteurs et les développeurs sur plusieurs projets. L'Objet générique n'est pas instanciable. Ses Propriétés ne sont exploitées qu'au travers d'un héritage.

OBJET METIER

Objet remarquable d'un métier de l'entreprise, porteur d'une sémantique, hors solution informatique, contribuant à décrire les invariants de ce métier dans un système d'information (S.I.), tant sur le plan de l'urbanisme (domaines d'activités, découpage du S.I., ...), que sur le plan de l'administration (données, traitements, composants métiers...).

OBJET TECHNIQUE

Objets de l'architecture technique contribuant à developper :

- l'urbanisme entre les logiciels des différents niveaux des configurations,
- l'administration des composants, des logiciels, des bases de données, des matériels,...
- le développement, la maintenance, l'intégration des composants logiciels applicatifs fonctionnels et techniques,...
- les choix et la mise en oeuvre des composants matériels et logiciels de base.

OBJET UTILE

Propriétés d'un Objet liées à un contexte d'utilisation; c'est-à-dire suivant la perspective d'un ensemble d'utilisateurs (et d'informaticiens) qui partagent des besoins homogènes en matière de système d'information.

PERSISTANCE

La Persistance permet de faire survivre à travers le temps et l'espace les données à l'exécution d'un programme afin de les utiliser ultérieurement :

- pour le temps : les S.G.B.D.O.O.,
- pour l'espace : les déplacements dans les systèmes distribués.

Cette notion doit être implicite et transparente à l'utilisateur des données. Elle doit aussi être orthogonale aux Types, c'est-à-dire que tout Objet doit être conservé, sans la moindre conversion et sans tenir compte de son type. La Persistance doit faire L'Objet d'une Classe contenant tous les Services de traitements, écran ...

POLYMORPHISME

Redéfinition de fonctions similaires, mais présentant des particularités. Le Polymorphisme s'applique à des Services de même nom, mais rattachés à des Classes différentes.

PRINCIPE DE LOCALITE

Principe qui oblige à penser Encapsulation en demandant aux Objets d'accomplir eux-mêmes leurs actions, et non à chercher à manipuler directement ces Objets au loin par l'intermédiaire de procédures globales. Ce principe dote les Classes des propriétés d'autonomie.

PRIVE

Dans une Classe d'Objets, cette notion s'applique à (Encapsule) l'ensemble des données (Attributs) et des traitements (Méthodes) non visible de l'extérieur (utilisateur). L'accès depuis l'extérieur est réservé au seul administrateur.

PROPRIETE

Les Classes d'Objets sont dotées de Propriétés :

- les données (Attributs) et les traitements (Méthodes). Il existe deux catégories de Propriétés :
  - les propriétés privées,
  - les propriétés publiques.

PUBLIC

Dans une Classe d'Objets, ce terme s'applique à l'ensemble (Services offerts) des données (Attributs) et des traitements (Méthodes) visibles de l'extérieur (utilisateur).

REDEFINITION

Il est possible d'avoir dans un même système plusieurs Méthodes de même nom ayant des comportements totalement différents. Cette fonction a pour objet de simplifier l'écriture et la maintenance des programmes, et d'en préserver la lisibilité. Elle a cependant des répercussions sur le mode de liaison des procédures et de leur implantation : elle fait appel aux procédures de liaison Dynamique qui ne sont, en général, pas gratuites.
Ainsi, l'Héritage permet de redéfinir dans les Sous-Classes des Méthodes définies dans une Sur-Classe. Lorsqu'une telle Méthode possède le même nom, mais a une implémentation différente, on l'appelle Surcharge sémantique.

RESOLUTION DYNAMIQUE

Lorsqu'un Type de données n'est connu qu'à l'exécution (cas du Polymorphisme), l'édition des liens ne peut être totalement réalisée à la compilation. Ce que l'on attend de lui correspond à ce qui lui a été attribué effectivement à sa création. Le lien entre un Message et son implantation ne se fait qu'à l'exécution : elle est aussi appelée Résolution tardive.

RESOLUTION STATIQUE

Le Type statique d'un Objet définit ce que l'on attend de lui a priori.

REUTILISATION

Finalité et conséquence logiques de l'Orienté Objet qui découpe l'application en composants logiciels réutilisables afin d'optimiser la production de logiciels, d'améliorer la qualité et de rendre plus fiable cette production.

SERVICE

C'est une finalité fonctionnelle de une ou plusieurs Classes, rendue par un ou plusieurs Objets.

SIGNATURE

Les traitements (Méthodes) d'une Classe d'Objets sont divisés en deux parties :

- l'entête, ou Signature est aussi appelé Spécification, seule partie visible par l'utilisateur.
- le corps ou implantation.

La Signature d'une Méthode est représentée par la définition des Types de chacun de ses arguments, ainsi que de la valeur de retour. Cette Signature se décompose donc en :

- un receveur appelé destinataire,
- un sélecteur ou un nom de procédure (Méthodes).

SOUS-CLASSE

Position relative d'une Classe par rapport à une autre Classe. En l'occurrence, position immédiatement intérieure par rapport à une autre Classe (qualifiée Sur-Classe). Incorporation de propriétés appartenant à une Sur-Classe avec ajouts de Propriétés propres, uniques.

SUR-CLASSE

Position relative d'une Classe par rapport à une autre Classe. En l'occurrence, position immédiatement supérieure par rapport à une autre Classe (qualifiée Sous-classe). Factorisation de propriétés appartenant à plusieurs Sous-Classes.

TYPE

Il définit les Objets de mêmes caractéristiques et de mêmes comportements dans une Classe. On dit que ces Objets sont Typés. Il existe des Classes de Types différents :

- le Type Réel représenté par des Objets existants,
- le Type Abstrait sans Objet, mais décomposable en Classes à Types Réels.

III.2 - Tableau des synonymes

SYNONYMES	*DEFINITIONS DES TERMES DE BASE
COMMUNICATION, DIALOGUE ENTRE OBJETS	*MESSAGE
COMPOSITION	*AGREGATION
DONNEE	*ATTRIBUT
ENVELOPPE	*ENCAPSULATION
CARACTERISTIQUE	*PROPRIETE
FRAMEWORK	*ENSEMBLE D'OBJETS APPLICATIFS
LIEN DE GENERALISATION-SPECIALISATION	*HERITAGE SIMPLE
OCCURRENCE	*INSTANCE
RESOLUTION TARDIVE	*RESOLUTION DYNAMIQUE
SOURCE	*CORPS
SOUS-TYPE	*SOUS-CLASSE
SUR-TYPE	*SUR-CLASSE
SURCHARGE	*REDEFINITION
TRAITEMENT, FONCTION	*METHODE
TYPAGE, GENERALISATION/SPECIFICATION	*CLASSIFICATION
	*ABSTRACTION
	*APPROCHE BASEE OBJET
	*APPROCHE ORIENTEE OBJET
	*CAPSULE
	*CLASSE
	*CONTRAINTE
	*CYCLE DE VIE DE L'OBJET
	*DELEGATION
	*DOMAINE
	*ETAT D'UN OBJET
	*EVENEMENT
	*EVENEMENTIEL
	*EXTENSIBILITE
	*GENERALISATION
	*HERITAGE MULTIPLE
	*HERITAGE ASCENDANT
	*HERITAGE DESCENDANT
	*HERITAGE EN REFERENCE
	*IDENTIFICATEUR D'OBJET

	*INSTANCIATION
	*MODELE DYNAMIQUE
	*MODELE FONCTIONNEL
	*MODELE STATIQUE
	*OBJET
	*OBJET GENERIQUE
	*OBJET METIER
	*OBJET TECHNIQUE
	*OBJET UTILE
	*PERSISTANCE
	*POLYMORPHISME
	*PRINCIPE DE LOCALITE
	*PRIVE
	*PUBLIC
	*RESOLUTION STATIQUE
	*REUTILISATION
	*SERVICE
	*SIGNATURE
	*TYPE

IV - ILLUSTRATION

Pour illustrer l'approche Objet, un simple exemple permet de mettre en évidence quelques-uns des principaux mécanismes caractérisant l'approche Objet.

IV.1 - Etude du Tour Opérator : « Séjour Village-Vacances »

L'application « Séjour Village-Vacances » ci-après est la description d'un contexte que chacun connaît bien : celui d'une petite agence de voyage qui vend différents produits touristiques. En l'occurrence, nous nous intéresserons à l'agent chargé de vendre des séjours village-vacances avec les vols aller-retour associés.

Le mécanisme d'achat d'un séjour dans un village-club est tout à fait simple. L'acquisition d'un séjour se déroule ainsi : la personne, « le client » intéressé par un séjour se rend à l'agence de voyage et s'adresse à un interlocuteur, « l'agent de voyage », chargé de ce type de séjour. Le client potentiel formule sa demande auprès de l'agent de voyage qui lui demande l'ensemble des informations nécessaires à l'établissement de son séjour :

- nombre de personnes concernées par le séjour,
- village-vacances envisagé pour le séjour,
- type de règlement choisi par le client

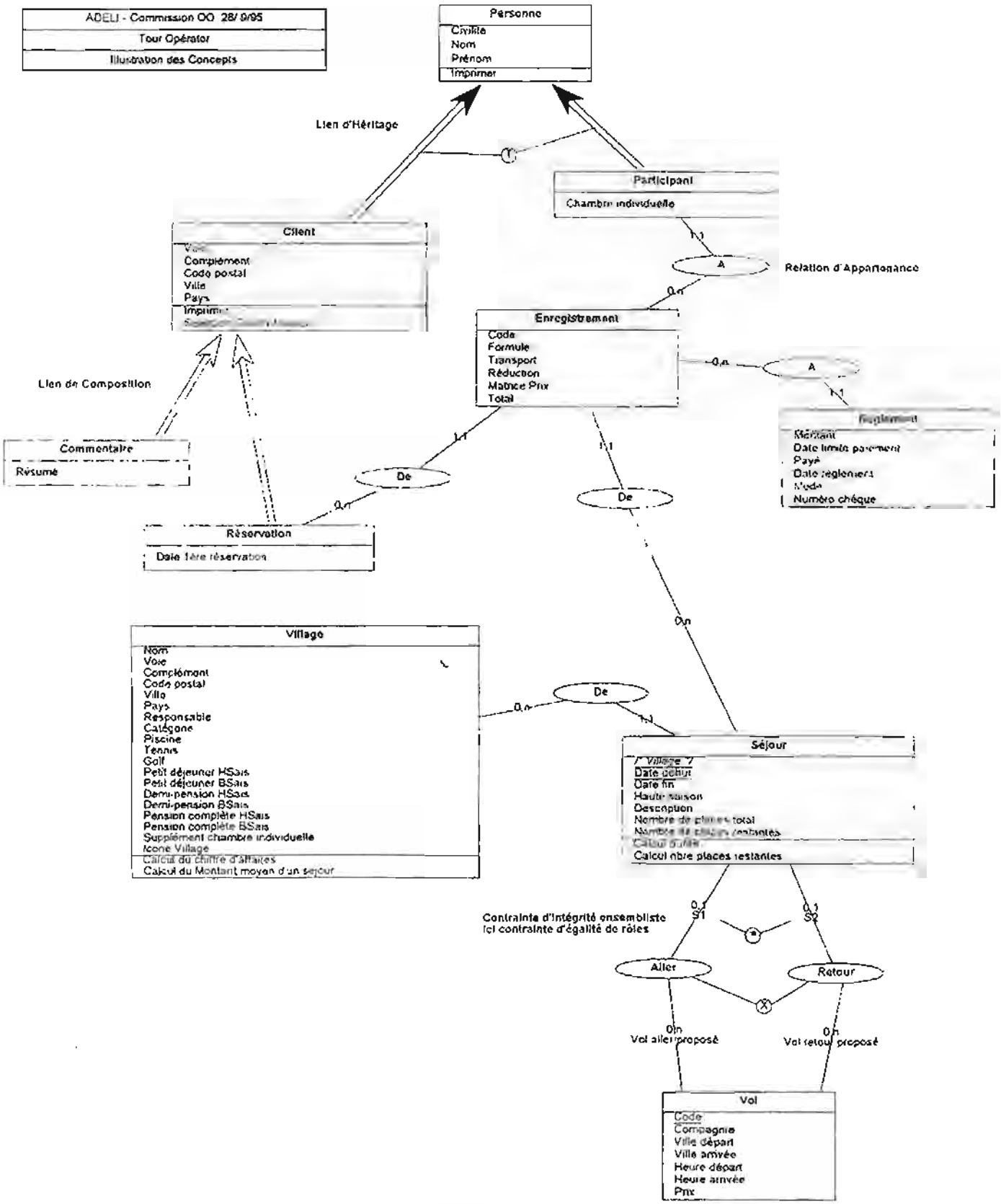
A l'issu de ce premier échange, l'agent de voyage est en mesure de vérifier si le client est un client fidèle ou s'il s'adresse à l'agence pour la première fois : une bonification est accordée aux clients fidèles.
Ensuite, l'agent de voyage vérifie la disponibilité des places dans le village-vacances demandé ainsi que l'adéquation parfaite du vol aller et du vol retour correspondant. Si tous les critères requis sont réunis, alors les réservations séjour village-vacances, vol-aller et vol-retour sont effectuées et le voyage est alors enregistré.

Conceptuellement, les composants requis pour réaliser cette application informatique en approche Objet apparaissent naturellement :

- nous avons besoin de décrire la personne qui va venir commander le séjour et la ou les personnes qui bénéficieront de ce séjour,
- pour savoir si l'acquéreur du séjour est un bon client, nous avons besoin d'accéder à l'historique des clients pour vérifier si son nom figure au fichier,
- pour saisir les caractéristiques de son séjour, il faut pouvoir enregistrer le type d'hébergement choisi, le village-vacances sélectionné ainsi que le vol-aller et le vol-retour correspondant,
- enfin, pour le voyagiste, il est important de tenir à jour un ensemble de statistiques et d'éléments comptables lui permettant de piloter son agence.

IV.2 - Conception Orientée Objet du projet « Séjour Village-Vacances »

Le schéma ci-après groupe l'ensemble des Objets nécessaire à l'enregistrement effectif d'un séjour en village-vacances.



IV.3 - Détails composant l'application de réservation « Séjour Village-Vacances »

L'analyse pour réaliser cet applicatif illustre plusieurs concepts de l'Orienté Objet.

Concept de CLASSE, concept d'ATTRIBUT, concept de METHODE :

Huit Classes ont été créées, contenant chacune leurs Attributs (zones) et leurs Méthodes (algorithmique de traitement) pour couvrir les besoins propres de l'applicatif :

- La Classe « Personne », trois Attributs, une Méthode.
- la Classe « Client », cinq Attributs, deux Méthodes,
- la Classe « Participant », un Attribut,
- la Classe « Enregistrement », six Attributs.
- la Classe « Règlement », six Attributs,
- la Classe « Village », dix-neuf Attributs, deux Méthodes.
- la Classe « Séjour », sept Attributs, deux Méthodes.

Concept de CONTRAINTE D'INTEGRITE ENSEMBLISTE :

Pour garantir l'association vol-aller et vol-retour, qui correspond à une obligation, un lien d'intégrité ensembliste attache vol-aller et vol-retour, un autre lien impose la règle suivante : obligation de réserver vol-aller et vol-retour sur deux vols différents.

Concept d'AGREGATION

Exemple n° 1 :
En plus de ses propriétés, la Classe « Client » agrège les informations historiques contenues dans la Classe du client fidèle « Réservation » et les informations résumées contenues dans la Classe « Commentaire ».

Exemple n° 2 :
Pour assurer la cohérence, un lien d'appartenance attache la Classe « Enregistrement » à la Classe « Participant » et la Classe « Enregistrement » à la Classe « Règlement ».

Concept d'HERITAGE :

Tant pour la Classe « Client » que pour la Classe « Participant », la description de chaque individu est nécessaire ; il apparaît superflus de décrire plusieurs fois les Attributs civilité, nom et prénom par exemple. Plutôt que de répéter les mêmes Attributs, il est plus facile de décrire ces informations éminemment redondantes dans une seule Classe qui servira à toutes celles qui en auront besoin. C'est le concept d'Héritage qui permet ensuite l'utilisation de ces informations. Tous les clients,.. et tous les participants à un séjour ont une « civilité », un « nom » et un « prénom ». Ils sont décrits une fois pour toutes dans la Classe « Personne ». D'un point de vue perspective, la Classe « Personne » précède logiquement les Classes « Client » et « Participant ». Ce positionnement logique fait que la Classe « Personne » est vue comme la Sur-Classe des Classes « Client » et « Participant ». Les Classes « Client » et « Participant » utilisent la Classe « Personne » pour constituer un Objet complet, ils sont vus comme Sous-Classes par rapport à la Classe « Personne ».

Ainsi, logiquement, dans les arbres d'Héritage, une « Sur-Classe » peut être la « Sous-Classe » d'un autre « Sur-Classe » ; une « Sous-Classe » peut être la « Sur-Classe » d'une autre « Sous-Classe ». Tout est affaire de positionnement à un instant donné.

L'écran CLIENT ci-après (généré par un outil upper-case Orienté Objet) illustre complètement le concept d'Héritage : en effet, on retrouve parfaitement dans sa composition les Attributs provenants de la Classe « Personne » concaténés à ceux de la Classe « Client ». En pratique, l'attachement du lien d'Héritage dans la Classe « Client » vers la Classe « Personne » génère cette concaténation et concrétise l'Objet Client.

Concept de SURCHARGE :

Dans le schéma conceptuel, les Classes «Personne» et «Client» comportent un exemple de «surcharge». Dans le cas de la Classe «Personne», compte tenu du contenu visible par la Classe «Personne», la Méthode «Imprimer» imprimera un spectre d'informations beaucoup plus restreint que celui qu'elle imprimera à son invocation dans le corps de la Classe «Client» enrichie des Propriétés de cette Classe.

V - BIBLIOGRAPHIE

Bibliographie de la commission Objet

Ces documents sont des documents de base utilisés ou connus des membres de la commission et ne peuvent représenter l'intégralité des documents existants. Les ouvrages et documents cités s'inscrivent dans le contexte choisi de l'étude, c'est-à-dire la conception de systèmes d'informations de gestion orientée objet. Mais la publication d'ouvrages directement liés à ce domaine est rare. Les expériences et réalisations effectuées par les administrations, entreprises, sociétés de services et consultants sont perçues comme des savoir-faire maison à ne pas divulguer ou seulement sous une forme édulcorée. Un effort de transposition et d'extrapolation du lecteur sera donc nécessaire pour prendre en compte les savoirs publiés et l'appliquer à son environnement.

Documents :

BRES, P.A., *Les apports de l'approche objet et la méthode des objets naturels.* Argenteuil : Concis. 6 avril 1993. Conférence AGL93. 18p.

CHEREL C. *Objets de réflexion.* La Défense : IMPROVE S.A. 13 septembre 1994 10p.

PFISTER, J.-P. *Note sur le passage à la technologie objet.* Ermont : S.N.C.F. GF-PCSI/T. Mars 1995 10 p.

PFISTER, J.-P. *Concepts de base et définition de l'orienté Objet.* 13 p. Extrait de *l'orienté objet.* Ermont : S.N.C.F. IG/SC. Juin 1994. 350 p.

ROCHFELD, A.E. *Evolution de Merise vers les objets : la méthode OOM.* Paris : AR consultant. 1994. 32 p.

Ouvrages :

AFCET, *Actes des journées de synthèse. Méthodes d'analyse et de conception orientées objet des systèmes d'information 22-23 novembre 1993.* AFCET Novembre 1993. 450 p.

BOOCH, G. *Object oriented designs with applications.* Menlo Park, Californie : Benjamin-Cummings. 1991.

CASTELLANI, X. *L'ingénierie des besoins.* Paris : Masson. 1993. 432 p. Série MCO, méthodologie générale d'analyse et de conception des systèmes objets. T.1.

COAD, P., YOURDON, E. *Object Oriented Analysis.* Englewood Clifffs, NJ : Prentice Hall. 1990. Col. Yourdon Press computing series.

COAD, P., YOURDON,E. *Object Oriented Design.* Englewood Clifffs, NJ: Prentice Hall. 1991. Col. Yourdon Press computing series.

DEFRAY, P. *Ingénierie objet - Approche classe-relation, application à C++*. Paris : Masson. 1992. 244 p.

MARTIN, J. *Principal of object oriented. Analysis and design.*. Prenticed Hall. 1993.

MOREJON, J. Merise. *Vers une modélisation orientée objet*. Paris. Les éditions d'organisation. 1994. 256 p. Coll. Ingénierie des systèmes d'informations.

PANET. G., LETOUCHE.R. *MERISE/2 Modèles et techniques MERISE avancées*.Paris, Les éditions d'organisation. 1994. 360p. Coll. Ingénierie des systèmes d'informations.

RUMBAUGH. J. et al. *OMT Modélisation et conception orientées objet*. Paris : Masson. 1995. 515 p.

SHLAER, S., MELLOR, S. *Object-oriented systemes analysis : Modeling the world in data*. Englewood Cliffs. NJ : Yourdon Press. 1988.

VAUQUIER, D. *Développement orienté objet. Principes, processus, procédés*. Eyrolles. 1993. 310 p.