

Centre de Développement des Technologies Avancées

THESE

présentée en vue de l'obtention

du **Grade de Magister en Cybernétique**

Option : **ROBOTIQUE**

Par

Mme Djamila LEZZAR

**LOOB : un Langage Orienté objet
avec héritage multiple**

Soutenue le 09 Mai 1992, devant le jury composé de :

Mr	H. BESSALAH	Président
Melle	D. HATTAB	Examineur
Mr	H. KHELLALFA	Examineur
Mr	M. BELMIHOUB	Examineur
Mr	C. OUSSALAH	Rapporteur

Remerciements

Je remercie Monsieur H. BESSALAH, Directeur du Centre de Développement des Technologies Avancées, d'avoir bien voulu présider ce jury.

Je remercie chaleureusement Mr C. OUSSALAH d'avoir accepté la lourde tâche de rapporter cette thèse. Celle-ci doit beaucoup à sa lumineuse compétence. J'espère qu'il trouvera ici l'expression de toute ma gratitude.

Je remercie Melle D. HATTAB, Maître-assistante à l'USTHB et Chercheur associé au CDTA/LGL, Mr H. KHELLALFA, Chargé de recherche au CERIST, ainsi que Mr M. BELMIHOUB, Chargé de recherche au CDTA/LGL, d'avoir acceptés de participer à ce jury.

Que toute l'équipe "systèmes experts" du Laboratoire de Génie Logiciel, à sa tête Mr M. Belmioub, soit remerciée pour les fructueuses critiques qu'ils ont constamment portés à ce produit et à leur farouche détermination d'en faire un outil de travail pour leur environnement.

Je remercie l'ensemble du personnel du Laboratoire de Génie Logiciel pour leur aide et encouragement.

Last, but not least, que Mr R. OUIGUINI, Chef du Laboratoire de Robotique, trouve ici mes sincères remerciements pour tous les efforts qu'il a manifesté pour la concrétisation de cette formation.

RESUME

Les langages à objets sont appelés à tenir une place de plus en plus importante dans toutes les activités logicielles. Que ce soit dans le domaine de l'intelligence artificielle, du génie logiciel ou de celui des bases de données, ils constituent la solution la plus prometteuse pour l'analyse, la conception, et la maintenance des systèmes informatiques complexes.

L'objectif de ce mémoire est la définition formelle et l'implantation d'un langage à objets LOOB (Langage Orienté OBjet).

Nous définissons un modèle théorique dans lequel nous décrivons l'objet, les différentes relations entre objets ainsi que l'héritage qui représente le mécanisme de partage de l'information. L'héritage multiple nous amène à définir les différents types de conflit en fournissant un moyen de les résoudre. Nous présenterons, ensuite, les principaux concepts ayant contribué à l'élaboration du langage ainsi que les différentes fonctions qui sont implantées.

MOTS CLES :

Langage à objets, langages hybrides, frame, classe, instance, héritage simple, héritage multiple, message, conflits.

ABSTRACT

The importance of object oriented language and their applications in the real world have been steadily increasing for the last ten years.

This work presents the framework within which LOOB, an object oriented language, has been defined and implemented.

Objects, relationship between objects, and inheritance mechanism are defined. Multiple inheritance conflicts are pointed out and LOOB's solutions extensively reviewed. The concepts used by LOOB and the functions that it implements are presented, too.

KEYWORDS :

Object oriented language, hybrid language, frame, class, instance, simple inheritance, multiple inheritance, message, conflicts.

ملخص

لغات البرمجة المبنية على المواضيع مدعوة لإحتلال مرتبة ذات أكثر أهمية في جميع الأعمال البرمجية. سواء كان هذا في ميدان الذكاء الاصطناعي، هندسة حزم البرامج أو قواعد المعطيات. فإن هذه اللغات تشكل الحل المناسب لتحليل، تصميم و صيانة الأنظمة الإعلامية المعقدة.

هذه الأطروحة يتمثل في تعريف رسمي للغة مبنية على المواضيع ودخولها في حيز التطبيق. نعرف نموذج نظري نصف من خلاله المواضيع، مختلف العلاقات الموجودة بين هذه المواضيع بالإضافة إلى الوراثة التي تمثل ميكانزمات إقتسام المعلومات. الوراثة المتعددة تدفعنا إلى تعريف جميع أنواع النزاعات و سبل حلها. نقم بعد ذلك أهم المفاهيم التي ساهمت في تشييد هذه لغة البرمجة وأهم الوظائف التابعة لها.

كلمات-أساسية : اللغات المبنية على المواضيع، اللغات المختلطة، إطار، القسم، فرد، وراثة فردية، وراثة متعددة، رسائل.

د. رشيد

A V A N T - P R O P O S

Au fur et à mesure de l'amélioration des outils matériels, s'ouvre un domaine de problèmes important concernant le logiciel. En effet, les logiciels développés ne satisfont pas en général les besoins des utilisateurs. Ils sont peu fiables, trop rigides et difficiles à maintenir [Dij72]. De plus, les coûts de ces logiciels ne cessent d'augmenter alors qu'on assiste à une diminution progressive du prix des machines [Boe73]. En d'autres termes, on peut affirmer que les logiciels développés sont incapables de s'adapter à un contexte qui, lui, ne cesse d'évoluer.

Devant la complexité croissante de ces problèmes, il s'avère indispensable de développer des outils de programmation modernes et sophistiqués intégrés dans des environnements appropriés.

Parmi les langages qui se sont développés à cet effet, se détachent les langages à objets dont le représentant principal est Smalltalk-80 [Gol85]. Les langages à objets sont fondamentalement différents des langages traditionnels par le fait que les programmes ne sont pas scindés en procédures et données. Ces programmes sont rendus modulaires par création d'entités cohérentes, indépendantes et facilement manipulables appelés *objets*, qui ne sont en réalité que la transposition informatique du monde réel. Ces langages disposent d'un certain nombre de principes pouvant offrir des qualités pour mettre en oeuvre des méthodes de programmation rigoureuses comme par exemple l'abstraction de données, la modularité et la réutilisabilité des logiciels.

Dans ce mémoire, nous nous proposons de définir formellement et d'implanter un langage à objet LOOB (Langage Orienté OBjet) dont voici le plan :

Why do you propose a new OO language?

Il se divise en quatre chapitres.

CHAPITRE 1. LES LANGAGES A OBJETS

nb3) Ce chapitre présente d'abord les concepts de base de la programmation par objets. Ensuite, nous étudierons successivement les langages de classe et les langages de frame, chacune de ces deux catégories de langages privilégie un point de vue sur la notion d'objet. Nous insisterons sur une troisième catégorie de langages à objets intégrant les caractéristiques empruntées aux deux catégories précédentes : les langages hybrides. Nous terminerons ce chapitre en soulignant les qualités des langages à objets.

actr objects →

CHAPITRE 2. MODELE THEORIQUE DE LOOB

Nous présentons ici le modèle théorique ayant guidé à la définition du langage. Ce modèle est défini par les trois modèles suivants : modèle de définition, modèle de description et modèle d'exécution. Le modèle de définition définit l'objet ainsi que les différentes relations qui lient les objets entre eux.

Le modèle de description décrit l'héritage qui permet le partage ou la factorisation des informations. Dans ce modèle, les différents types de conflits qui peuvent apparaître lors d'un héritage sont détectés.

Enfin, le modèle d'exécution s'intéresse à l'aspect dynamique du langage et propose un moyen de résoudre tous les types de conflits déjà détectés.

CHAPITRE 3. DESCRIPTION DE LOOB

*why is it compared to just
~~the other~~ hybrid languages?
a single*

Les principaux concepts ayant contribué à l'élaboration de LOOB sont exposés dans ce chapitre. Appartenant à la famille des langages hybrides, LOOB est comparé à l'un d'entre eux : le langage LOOPS [Ste86a]. Un petit aperçu des possibilités de ce langage précèdera cette comparaison.

Enfin, nous récapitulons dans un tableau comparatif les caractéristiques des principaux langages hybrides.

CHAPITRE 4. IMPLANTATION

Ce chapitre détaille l'implantation de LOOB. Il est structuré en trois parties définissant chacune des fonctions qui correspondent aux trois modèles décrits au chapitre 2.

Dans le modèle de définition, on définit principalement les fonctions de création de classe et d'instance.

Dans le modèle de description, on définit les fonctions de manipulation d'informations dans les objets. Certaines de ces fonctions utilisent essentiellement l'héritage.

Enfin dans le modèle d'exécution, on définit les fonctions qui permettent de résoudre les conflits d'héritage ainsi que les utilitaires et fonctions d'utilisation.

TABLE DES MATIERES

CHAPITRE 1. LES LANGAGES A OBJETS.	8
Introduction	9
1. Concept de base de la programmation par objets	10
Introduction	10
1.1 L'objet	11
1.2 La classe	11
1.3 L'envoi de message	13
1.4 L'héritage	14
2. Les différents langages à objets	16
2.1 Les langages de classes	16
2.1.1 Historique	16
2.1.2 Entité de base	16
2.1.3 Hiérarchie et héritage	17
2.1.4 Communication	18
2.1.5 Exemples	19
2.1.6 Avantages et inconvénients des classes . .	22
2.2 Les langages de frames	24
2.2.1 Historique	24
2.2.2 Entité de base	24
2.2.3 Attributs et facettes	25
2.2.4 Hiérarchie et héritage	26
2.2.5 Accès aux attributs et héritage	27
2.2.6 Cohérence de la hiérarchie d'objets . . .	28
2.2.7 Communication	29
2.2.8 Exemples	31
2.2.9 Avantages et inconvénients des frames . .	32
2.3 Différences entre langages de classes et langages de frames	34
2.4 Les langages hybrides	35
2.4.1 Notion de langages hybrides	35
2.4.2 Caractéristiques d'un langage hybride . .	35

2.4.3	Lisp et les objets	36
2.4.4	Prolog et les objets	39
2.4.5	Exemples	40
3.	Caractéristiques des objets	42
3.1	Abstraction de données et encapsulation	42
3.2	Le typage	43
3.3	Le polymorphisme	44
3.4	Généricité	45
3.5	L'envoi de message	46
3.6	Modularité	46
3.7	Parallélisme	46
4.	Conclusion	48
 CHAPITRE 2. MODELE THEORIQUE DE LOOB		 49
Introduction		50
1.	Le modèle de définition	51
1.1	Définition hors contexte	51
1.1.1	Définition de l'objet	51
1.1.2	La classe et l'instance	52
1.2	Définitions en contexte.	52
1.2.1	Relations "spécifie" et "généralise"	53
1.2.2	Relations "instance de" et "sous-classe de".	54
2.	Le modèle de description	56
2.1	La relation d'héritage	56
2.2	Le graphe d'héritage.	58
2.3	L'héritage simple et l'héritage multiple	59
2.3.1	L'héritage simple.	59
2.3.2	L'héritage multiple	60
2.4	Notions de conflit	60
2.4.1	Les conflits de l'héritage simple	61
2.4.2	Les conflits de l'héritage multiple	61

3.	Le modèle d'exécution	65
3.1	Résolution de conflit de l'héritage simple	65
3.2	Résolution de conflit de l'héritage multiple	66
3.2.1	Les différents mécanismes de l'héritage multiple	66
3.2.2	Résolution de conflit	70
3.2.3	Choix du concepteur	72
3.3	Description de l'algorithme P1	73
3.3.1	Critères de choix	73
3.3.2	Différents algorithmes de linéarisation	78
3.4	Conclusion	83
4.	Conclusion	84
CHAPITRE 3. DESCRIPTION DE LOOB		85
Introduction		86
1.	Description de LOOB	87
1.1	L'objet	88
1.2	La classe	89
1.3	L'envoi de message	89
1.4	L'héritage	90
1.4.1	Héritage des attributs et des méthodes	91
1.4.2	L'ordonnancement des relations d'héritage	92
1.5	L'environnement de programmation	93
2.	LOOB et les autres langages	94
2.1	Caractéristiques de LOOPS	94
2.2	Comparaison de LOOB et de LOOPS	97
2.3	Tableau comparatif	98
3.	Conclusion	100

CHAPITRE 4. IMPLANTATION	101
INTRODUCTION	102
1. Modèle de définition	103
1.1 L'objet	103
1.1.1 Attributs et facettes prédéfinis	103
1.1.2 Implantation physique	107
1.2 Fonctions du modèle de définition	109
2. Modèle de description	111
2.1 Relation d'héritage	111
2.2 Héritage simple	112
2.3 Héritage multiple	113
2.4 Procédures de détection de conflit de l'héritage multiple	113
2.5 Fonctions du modèle de description	116
2.5.1 Fonctions de manipulation d'objets	116
2.5.2 Fonctions de manipulation d'attributs	118
A/ Fonctions n'utilisant pas l'héritage.	118
B/ Fonctions d'héritage	120
3. Modèle d'exécution	124
4. Utilitaires et fonctions d'utilisation	126
CONCLUSION	130
ANNEXE	133
BIBLIOGRAPHIE	137
INDEX	143