

THESE de DOCTORAT de L'UNIVERSITE PARIS 6

Spécialité

INFORMATIQUE

présentée

par Mr Jean-Francois BRETTE

pour obtenir le grade de DOCTEUR DE L'UNIVERSITE PARIS 6

Sujet de la thèse :

PASCAL/V :

**Un environnement pour l'apprentissage de la
programmation par découverte guidée.**

Réification et interprétation contextuelle du rôle des variables.

soutenue le 27 mai 1994
devant le jury composé de :

**Mr Patrick BARRIL
Mr Stefano CERRI
Mr Raymond DURAND
Mr Pascal ESTRAILLIER
Mme Françoise MADAULE
Mr Jean-François PERROT
Mr Harald WERTZ**

Rapporteur

**Directeur
Président
Rapporteur**

Résumé

Cette thèse présente la conception et la réalisation de PASCAL/V, un environnement de programmation permettant d'articuler enseignement et apprentissage et centré sur le rôle des variables. Il comprend un compilateur pédagogique, un outil de visualisation de trace au pas-à-pas, des inspecteurs spécialisés pour les types et une aide contextuelle pour les erreurs d'exécution. Un grand soin a été porté à la fidélité conceptuelle des outils interactifs de découverte. Ils sont souvent le résultat d'un compromis entre besoins didactiques et possibilités techniques des interfaces.

Nous considérons que le rôle d'une variable est une extension de son type et fait le lien entre structure de données et traitements. L'implémentation fait clairement la distinction entre les vérifications de compatibilité de types, effectuées à la compilation et les vérifications de contraintes portant sur les sous-types, effectuées à l'exécution. Ces dernières sont la base de l'aide contextuelle qui fournit des suggestions à l'élève afin de le guider dans son apprentissage. Ces suggestions constituent un élément de la représentation du domaine sous forme de règles PROLOG/V et permettent notamment à l'élève de comprendre le rôle de l'environnement d'exécution dans l'interprétation d'une erreur.

PASCAL/V est écrit en SMALLTALK/V sur MacIntosh et sa réalisation nous a amené à explorer en détail les principes de compilation et de débogage interactif en SMALLTALK/V afin de construire différenciellement un compilateur Pascal, à expérimenter les limites de l'architecture d'interface utilisateur MVC de Smalltalk dont nous faisons une critique accompagnée de propositions nouvelles et à mettre en avant l'importance des coopérations de classes dans la programmation objet, notamment pour l'organisation système et la compréhension de code.

Mots-clés : Environnement d'apprentissage intelligent, programmation objet, environnement de programmation, didactique de la programmation, guidage contextuel, Smalltalk, Pascal.

Abstract

This thesis presents the design and the implementation of PASCAL/V, a programming environment allowing the articulation between teaching and learning and centered on the role of variables. It includes a pedagogical compiler, a step-by-step tracing visualisation tool, specialized inspectors for types and a contextual help for dynamic diagnosis. Great care has been taken over the conceptual fidelity of these interactive discovery tools. They are the result of an inevitable compromise between didactic needs and the technical possibilities of interfaces.

We consider that the role of a variable is an extension of its type and links data structures with treatments. The implementation makes a clear distinction between type compatibility checking, carried out at compilation time and constraints checking concerning subtypes, carried out at running time. The latter is the basis for the contextual help which gives suggestions to the student in order to guide him in his learning process. These suggestions constitute a part of the representation of the domain in the form of PROLOG/V rules and allow the student to understand the role of the running environment in the interpretation of errors.

PASCAL/V is written in SMALLTALK/V on MacIntosh and its development led us to explore in depth the principles of compilation and interactive debugging in SMALLTALK/V in order to build differentially a Pascal compiler ; to test the limits of Smalltalk's MVC user interface of which we offer our criticisms together with new propositions ; and to highlight the significance of class cooperations in object oriented programming, notably for system organization and code understanding.

Keywords : Intelligent learning environment, object oriented programming, programming environment, didactics of programming, contextual coaching, Smalltalk, Pascal.

Chapitre 1 : Introduction	1
1. Problématique.....	1
2. Une approche multidisciplinaire.....	3
3. Présentation de notre travail	4
4. Organisation de ce mémoire.....	6
Chapitre 2 : Contexte général du travail	7
1. Introduction.....	7
2. Apprentissage et formation.....	8
2.1. Introduction.....	8
2.2. Conditionnement (ou comportementalisme).....	9
2.3. Constructivisme	10
2.4. Apprentissage et milieu social.....	11
2.4.1. Théorie sociale de l'apprentissage	11
2.4.2. Apprentissage coopératif.....	12
2.5. Conclusion.....	12
3. Psychologie cognitive.....	13
3.1. Introduction.....	13
3.2. Le modèle computo-symbolique du cognitivisme classique	13
3.3. Le modèle connexionniste	15
3.4. Conclusion.....	16
4. Le domaine de l'EIAO : des tuteurs et des environnements d'apprentissage.....	17
4.1. GUIDON : tutoriels intelligents et systèmes experts	17
4.1.1. GUIDON.....	17
4.1.2. NEOMYCIN	18
4.1.3. GUIDON2.....	18
4.1.4. Conclusion	19
4.2. Diagnostic de programmes	19
4.2.1. Introduction.....	19
4.2.2. Diagnostic d'exercice.....	20
4.2.2. Les environnements de programmation intelligents.....	22
4.2.3. Conclusion	23
4.3. WHY et le dialogue socratique	23
4.4. ACTP, Lisp-Tutor et la psychologie cognitive	25
4.4.1. Introduction.....	25
4.4.2. ACT*	25
4.4.3. Lisp-Tutor.....	26

4.4.4. Conclusion	27
4.5. SHERLOCK et l'apprentissage libre guidé ("coach")	28
4.5.1. Introduction.....	28
4.5.2. Des précurseurs : West et Wusor.....	28
4.5.3. MHO et Sherlock.....	29
4.5.4. Memolab-Etoile	32
4.6. Logo, Arcade : micro-mondes et environnements d'apprentissage	33
4.6.1. LOGO.....	33
4.6.2. ARCADE et les visualisations de programmes.	34
4.6.3. CABRI-GEOMETRE et le respect de contraintes.....	37
4.6.4. Conclusion	37
5. Modèle élève et modèle du domaine.....	39
5.1. Les techniques employées.....	39
5.2. Modèles de compétence et modèles qualitatifs	41
5.3. Compétence/performance	42
5.4. La connaissance en programmation.....	43
5.4.1. Les niveaux de savoir	43
5.4.2. Modéliser la connaissance de programmation.....	45
5.5. Modèle élève exécutable et compagnon d'apprentissage	46
5.6. Conclusion.....	48
6. Conclusion.....	50
Chapitre 3 : Le projet FORCE.....	52
1. Projet FORCE.....	52
1.1. Introduction.....	52
1.2. Objectifs.....	52
1.3. Un langage homogène et réflexif.....	53
1.4. Conclusion.....	53
2. DELTA.....	55
2.1. Introduction.....	55
2.2. Phase de spécification	55
2.3. Phase de programmation	56
2.4. Phase d'évaluation	56
2.5. Conclusion.....	56
3. DIPLOMAT.....	58
3.1. Introduction.....	58
3.2. Représentation des connaissances et architectures.....	58
3.3. Intégrer un environnement de programmation au tutoriel.....	59

3.4. Epistémologie et compilation.....	59
3.5. Conclusion.....	60
4. SINTONIA.....	61
4.1. Introduction.....	61
4.2. Description du système.....	61
4.3. Conclusion.....	62
5. GEODE.....	63
5.1. Introduction.....	63
5.2. Description du système.....	63
5.3. Conclusion.....	64
Chapitre 4 : Présentation de l'environnement Pascal/V	65
1. Objectifs et proposition.....	65
1.1. Introduction.....	65
1.2. Micro-monde	68
1.2.1. Le constructivisme pour l'enseignement conceptuel.....	68
1.2.2. Une approche pragmatique	69
1.3. Interventions de l'auteur.....	70
1.4. Interprétation dynamique "intelligente"	70
1.5. Une contrainte : l'implémentation.	71
1.6. Conclusion.....	73
2. Le langage Pascal/V.....	74
2.1. Introduction.....	74
2.2. Variables non initialisées.....	75
2.3. Portée des compteurs de boucles for.....	76
2.4. Paramètres de procédures.....	76
2.5. Conclusion.....	78
3. Le micro-monde	79
3.1. Introduction.....	79
3.2. Une présentation générale.....	80
3.3. Inspection des variables.	84
3.4. Semi-dépendance des fenêtres d'inspection.....	88
3.5. Outils spécifiques pour les accès aux tableaux.....	89
3.5.1. Liaison indice-structure.....	89
3.5.2. Visualisation de l'accès aux valeurs	91
3.6. Types utilisateurs : vision locale vs vision globale	92
4. Interventions de l'auteur.....	95
4.1. Introduction.....	95
4.2. Point d'arrêt et dialogue	95

4.3. Mise en valeur du rôle des assertions (aspect algorithmique)...	96
4.4. Paramètres vs variables locales (aspect flot de données).....	99
4.5. Conclusion.....	100
5. Interprétation contextuelle des erreurs de contrainte de type.....	102
5.1. Introduction.....	102
5.2. Les erreurs de type	103
5.2.1. Compatibilité et contrainte (types de base et sous-types).....	103
5.2.2. Contrainte et valeurs indéfinies.....	105
5.3. Interpréter une erreur de contrainte de type.....	109
5.4. Des hiérarchies d'interprétation : le besoin de contextualiser les règles.....	111
5.5. Conclusion.....	114
6. Conclusion.....	115
Chapitre 5 : Réalisation en Smalltalk/V.....	116
1. Introduction.....	116
1.1. L'approche orientée objet.....	116
1.1.1. Encapsulation.....	116
1.1.2. Héritage	117
1.1.3. Polymorphisme et protocoles	118
1.2. Le besoin d'une représentation graphique.....	120
1.3. Une programmation différentielle.....	121
2. La compilation en Smalltalk/V.....	123
2.1. Compilation	123
2.1.1 L'analyseur lexical (instance de VScanner).....	124
2.1.2. L'analyseur syntaxique (instance de VParser).....	124
2.1.3 Le compilateur interne (instance de VCompiler)	126
2.2. Débogage.....	127
2.2.1. Le processus interrompu.....	127
2.2.2. Re-compilation du texte source.....	128
3. Compilation Pascal/V.....	130
3.1. Introduction.....	130
3.2. Les trois classes de la compilation Pascal/V	130
3.2.1. L'analyseur lexical.....	130
3.2.2. L'analyseur syntaxique.....	131
3.2.3. Compilation et évaluation.....	132
3.2.4. Procédures et fonctions	132

3.3. Implantation dynamique.....	132
3.4. Environnement local.....	134
3.4.1. Compilation et re-compilation des pBlocs.....	135
3.4.2. Exécution.....	137
3.5. Transmission des paramètres.....	139
3.6. Conclusion.....	141
4. Implémentation des types dans Pascal/V	142
4.1. Introduction.....	142
4.2. Identifier les objets	142
4.3. Définir les services.....	142
4.3.1. Allocation.....	143
4.3.2. Contrainte.....	143
4.3.3. Compatibilité.....	144
4.3.4. Affichage.....	145
4.3.5. Liaison identificateur/type.....	146
4.3.6. Analyse syntaxique d'un type.....	147
4.4. Implémentation	147
4.4.1. Graphe des classes.....	148
4.4.2. Graphe de composition.....	150
4.4.3. Coopération d'objets.....	151
4.5. Conclusion.....	152
5. Le PDebugger.....	154
5.1. Introduction.....	154
5.2. Une trace sélective.....	155
5.2.1. Filtrer les méthodes tracées.....	155
5.2.2. Filtrer les variables.....	157
5.2.3. Filtrer les messages non stoppables.....	157
5.2.4. Conclusion	159
5.3. Insérer les interactions dans l'arbre.....	161
5.4. Conclusion.....	162
6. Interfaces et MVC.....	164
6.1. Introduction.....	164
6.2. MVC	164
6.2.1. Relations entre modèle, vue et controleur.....	164
6.2.2. Relations entre les sous-vues.....	167
6.2.3. Protocole de fermeture d'une fenêtre.....	167
6.3. Les inspecteurs Pascal/V.....	168
6.3.1. Introduction.....	168

6.3.2. Dépendance activation-inspecteurs.....	169
6.3.3. Caractériser une activation	171
6.3.4. Conclusion	172
6.4. Défilement synchronisé dans les fenêtres d'inspecteurs	173
6.4.1. Propagation de messages	173
6.4.2. Synchronisation du pilotage de modèles.....	175
6.5. Conclusion.....	175
7. Prolog/V et le diagnostic contextuel.....	177
7.1. Introduction.....	177
7.2. Primitives de diagnostic / règles de l'auteur.....	177
7.2.1. Prolog/V sous Smalltalk/V	177
7.2.2. La classe DynamicExpert.....	178
7.3. L'héritage des prédicats.....	180
7.3.1. Transposer le principe d'héritage dynamique des langages de classe à Prolog	180
7.3.2. Une nouvelle sémantique pour Prolog/V	184
7.3.3. Implémentation.....	185
7.4. Conclusion.....	186
8. Conclusion : des types, des gurus et des rhizomes.....	188
8.1. Des types	188
8.2. Des gurus et des rhizomes.....	190
Chapitre 6 : Bilan et perspectives.....	193
1. Pour Smalltalk/V.....	193
2. Pour Pascal/V	194
ANNEXES	196
Annexe 1 : PASCAL/V vs Turbo-Pascal.....	197
1. Les procédures	197
2. Les fonctions.....	198
3. Variables non initialisées.....	198
Annexe 2 : Règles BNF.....	200
Annexe 3 : "Thunk".....	202
Annexe 4 : Base de règles pour les compatibilités de types.....	203
Annexe 5 : Le dictionnaire identifièrs.....	204
Annexe 6 : Sept étapes vers un bonheur basé sur les objets.....	205
Annexe 7 : Extrait du code Smalltalk/V1.1.....	206

1. Protocoles de compilation Smalltalk/V (interface avec le compilateur interne)	206
2. VPParser>> method.....	207
3. VPParser>>procOrFuncCall.....	208
4. VPParser>>procOrFuncDef:.....	208
5. VPParser>>bloc.....	212
6. PDebugger>>walkback.....	214
7. VPCompiler>>positionsFor:.....	214
Références bibliographiques.....	216