PDF pour 25440-01 sommaire

Le présent fichier PDF **Sommaire** fournit des extraits du document

Delphia Prolog

Edité par la **Société Delphia**

Système de Programmation Logique

Manuel de Référence

Ce Manuel de référence a été édité an 1986 sous forme de classeur (270 pages) par la société Delphia, size à Seyssinet près de Grenoble.

Ce document est disponible sous le numéro **AC 25440-01** dans l'inventaire de la collection de l'ACONIT, Association pour un conservatoire de l'informatique et de la télématique, Grenoble.

Le présent PDF fournit un scan de :

-La page d'en-tête,
-La table des matières (3 pages),
-L'introduction (2 pages),
-Copie des figures 1, 2, 3, 5, 6,

Fiche rédigée le 19/5/2020 par Jean Ricodeau.

DELPHIA
PROLOG

Système
de
programmation logique

Manuel de Référence

Delphia

Table des matières

Table des figures

Introduction

PROLOG est un langage de programmation simple et puissant basé sur la logique symbolique. C'est un outil précis, qui permet des programmations concises, lisibles, claires, rapides et sans erreurs. Le mécanisme de programmation de base est un procédé de pattern matching appelé unification qui agit sur les structures d'enregistrement générales ("termes" de la logique) [Robi65] [Colm83].

Plusieurs implémentations de PROLOG ont été développées, généralement sous forme d'interprètes du langage (Prolog II, C-Prolog, D-Prolog), ou plus récemment sous forme de compilateurs (Quintus Prolog).

L'utilisation industrielle du langage PROLOG, motivée par le volume croissant d'applications de l'Intelligence Artificielle, amène de nouveaux impératifs qui ne sont pas pris en compte globalement par les systèmes classiques. Parmi les points principaux devant être développés, on peut citer les performances du langage, l'environnement de programmation, la modularité et l'ouverture.

Le système **Prolog de Delphia** apporte une réponse à ces problèmes, avec la volonté de fournir à ses utilisateurs un outil de programmation **performant**, ouvert, **modulaire**, et intégré dans un environnement de programmation ergonomique mettant à profit les techniques actuelles de génie logiciel.

La syntaxe du langage est compatible avec celle des systèmes PROLOG anglo-saxons (DEC-10, C-Prolog), qui s'imposent comme la norme de fait.

Le système est organisé autour d'un compilateur du langage, ce qui permet d'atteindre des performances environ 30 fois supérieures à celles des interpréteurs PROLOG classiques.

L'environnement de programmation a été étudié de façon à fournir de grandes possiblités de mise au point symbolique, un cycle de développement réduit, ainsi qu'un certain nombre d'outils de base de programmation. L'ergonomie du système est augmentée par son intégration aux environnements multi-fenêtres de type station de travail.

L'introduction du concept de modularité dans le langage PROLOG ouvre de multiples horizons. Outre l'utilisation classique des modules comme outils d'abstraction, l'adaptation de ce concept à la nature dynamique de PROLOG autorise la manipulation d'espaces de clauses indépendants et permet des techniques de programmations "orientées objet".

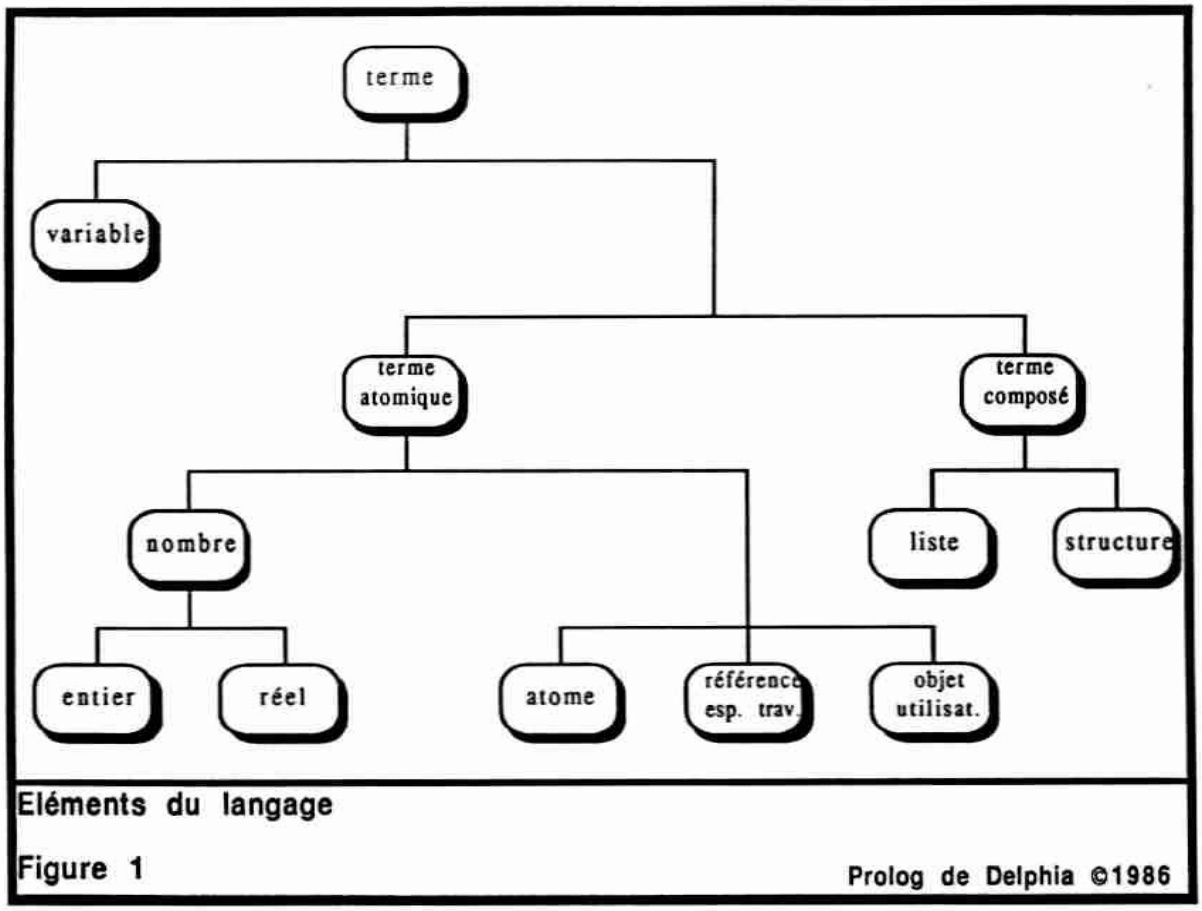
L'ouverture du système **Prolog de Delphia** vers les autres langages a été particulièrement étudiée. Cette caractéristique favorise la réalisation de parties d'applications spécifiques, la réutilisation de composants logiciels existants, ainsi que l'intégration des applications dans les systèmes d'exploitation.

Le présent manuel sert de référence pour toute utilisation de **Prolog de Delphia.**
Les trois premiers chapitres présentent le vocabulaire employé ainsi que les fondements de PROLOG.

Le chapitre 4 présente successivement les particularités de l'environnement de **Prolog de Delphia** .
Les prédicats prédéfinis du chapitre 5 sont classés par thème, mais sont accessibles par ordre alphabétique depuis l'annexe A qui présente un résumé de chacun d'entre eux.
L'index de l'annexe B renvoie à toutes les pages traitant d'un concept important. Chaque concept développé dans ce manuel est écrit en caractères gras.

1.1 Termes

Tous les objets manipulés par un programme PROLOG sont des termes. Les termes dits **atomiques** sont des feuilles, et sont différents des **termes composés**, que l'on peut assimiler à des arbres.
Les **variables** permettent de désigner n'importe quel terme. Le schéma suivant montre la décomposition des termes employés dans **Prolog de Delphia.**



Eléments du langage

Figure 1

Prolog de Delphia ©1986

1.1.5 Termes composés

Les termes composés (ou structures) permettent de décrire les données structurées. Ils sont constitués d'un atome appelé **foncteur** (ou symbole fonctionnel), et d'une suite de termes appelés **arguments**. Le nombre d'arguments est également appelé **arité**. Par extension, un atome peut être considéré comme un terme composé d'arité zéro. L'atome correspondant au foncteur de la structure s'écrit éventuellement suivi de ses arguments entre parenthèses et séparés par des virgules.

Exemple : pere(alain, paul)

est un terme composé dont le **foncteur** est *pere*, et dont les **arguments** sont *alain* et *paul*. Puisque ce terme composé a deux arguments, *pere* est un foncteur d'**arité** 2.

Notation : pour faciliter leur identification, les termes composés sont notés sous la forme : foncteur/arité.

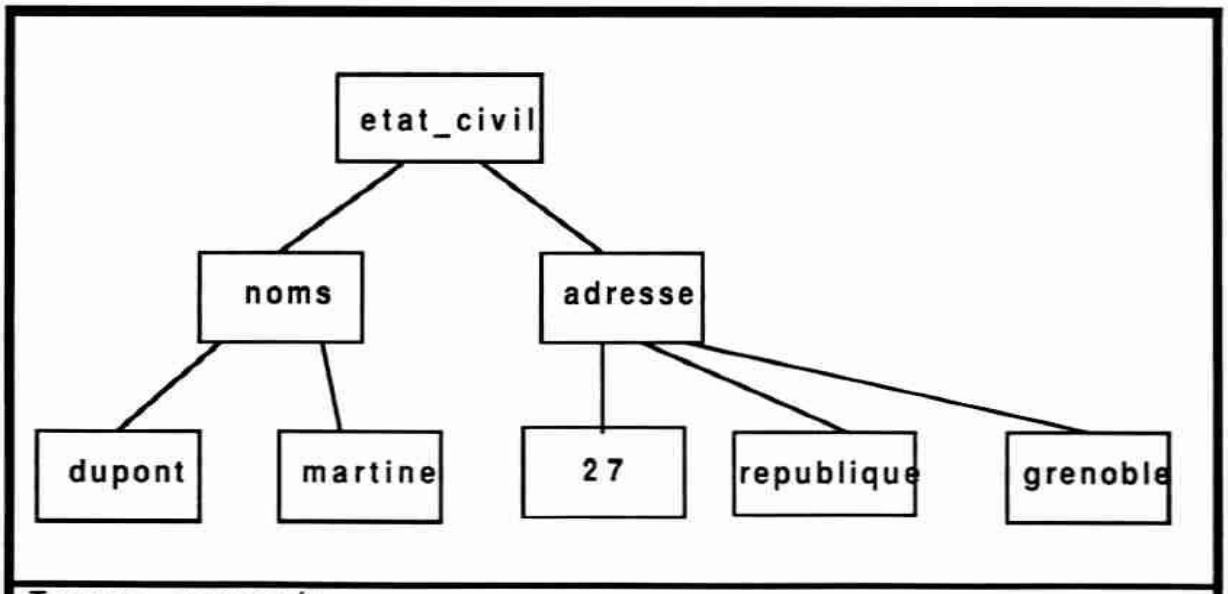
Par exemple, le terme composé *pere* ci-dessus, peut s'écrire : pere/2.

La définition d'une structure est récursive. Les arguments d'une structure peuvent consister en des termes PROLOG quelconques. Donc, une structure peut avoir pour arguments des termes atomiques, des variables, ou des termes qui sont eux-mêmes des structures.

Par exemple :

etat_civil(noms(dupont, martine), adresse(27, republique, grenoble))

Cette définition récursive conduit à représenter une structure sous la forme d'un arbre. La structure donnée en exemple peut être représentée sous la forme de l'arbre de la figure suivante :



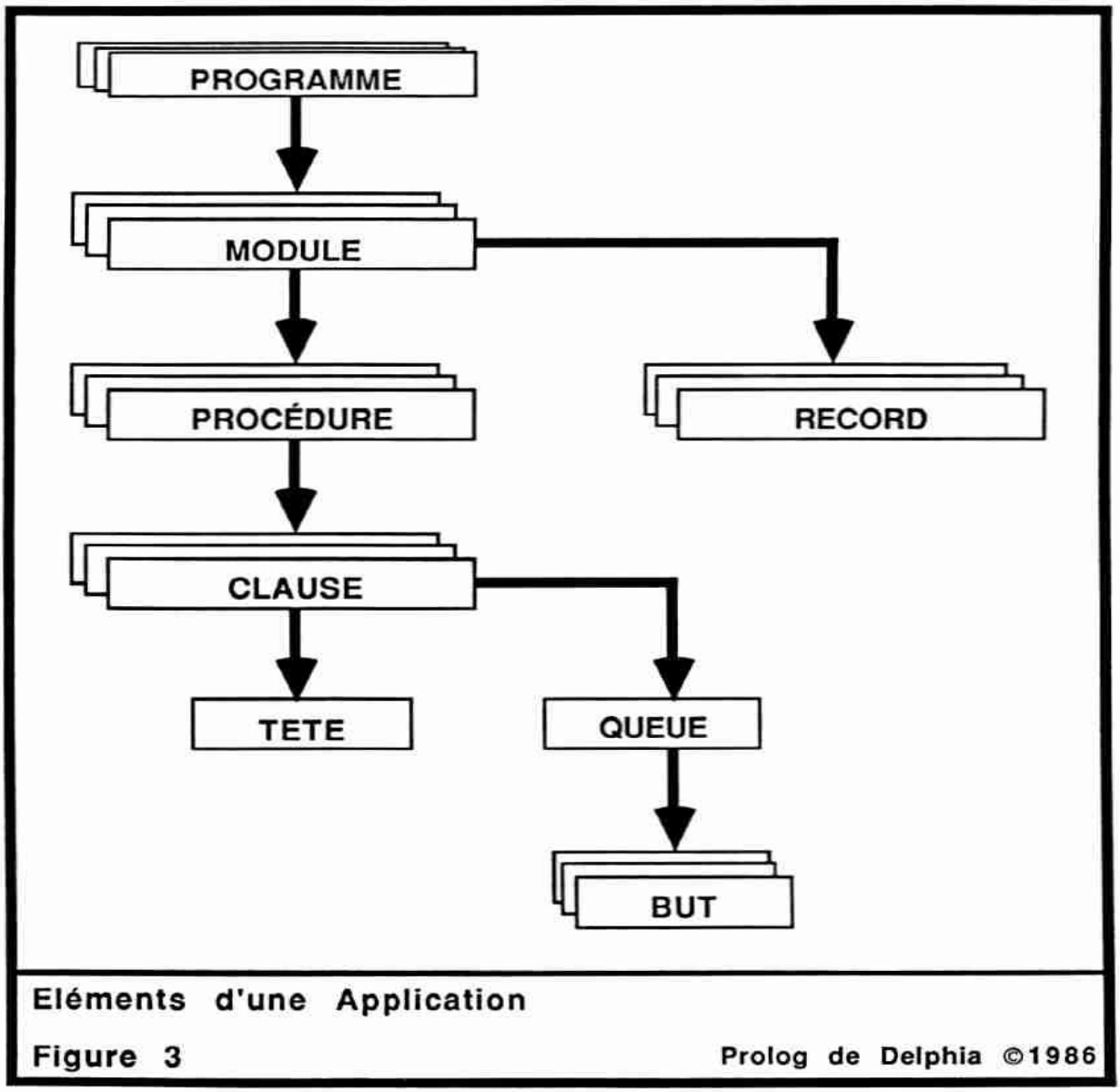
Termes composés

Figure 2 **Prolog de Delphia ©1986**

Chapitre2

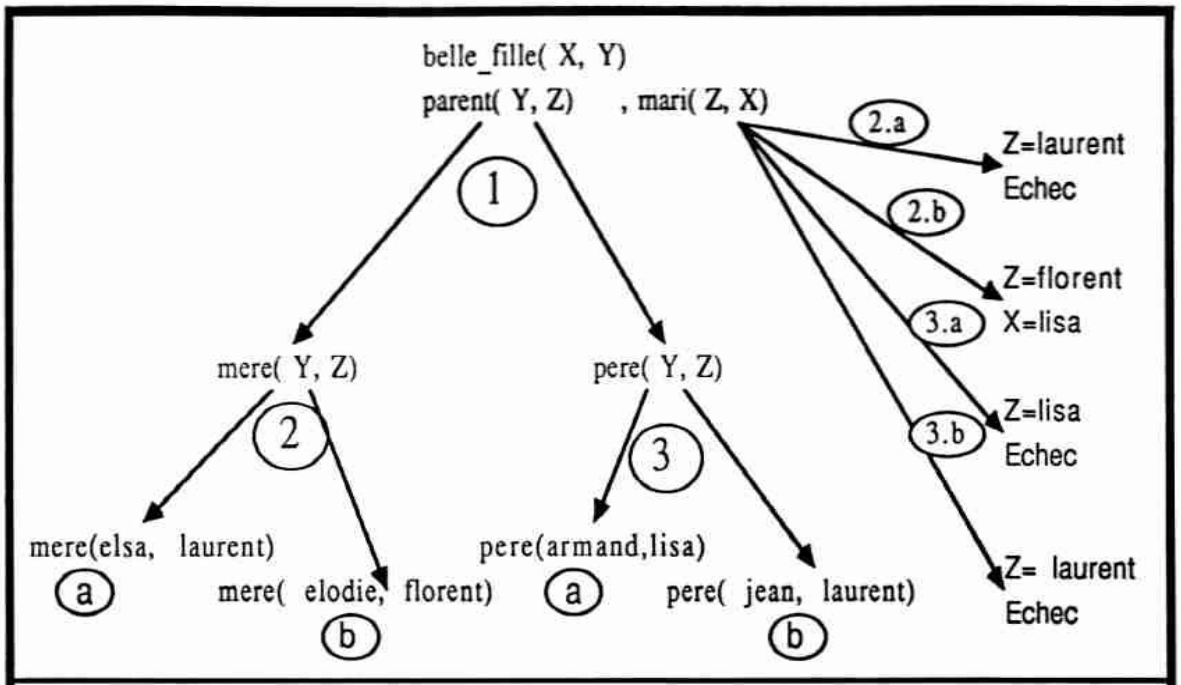
Programmes

La structure des éléments d'une application est résumée dans la figure suivante et détaillée ensuite, en allant des éléments les plus simples aux plus complexes.



Eléments d'une Application

Figure 3 **Prolog de Delphia ©1986**



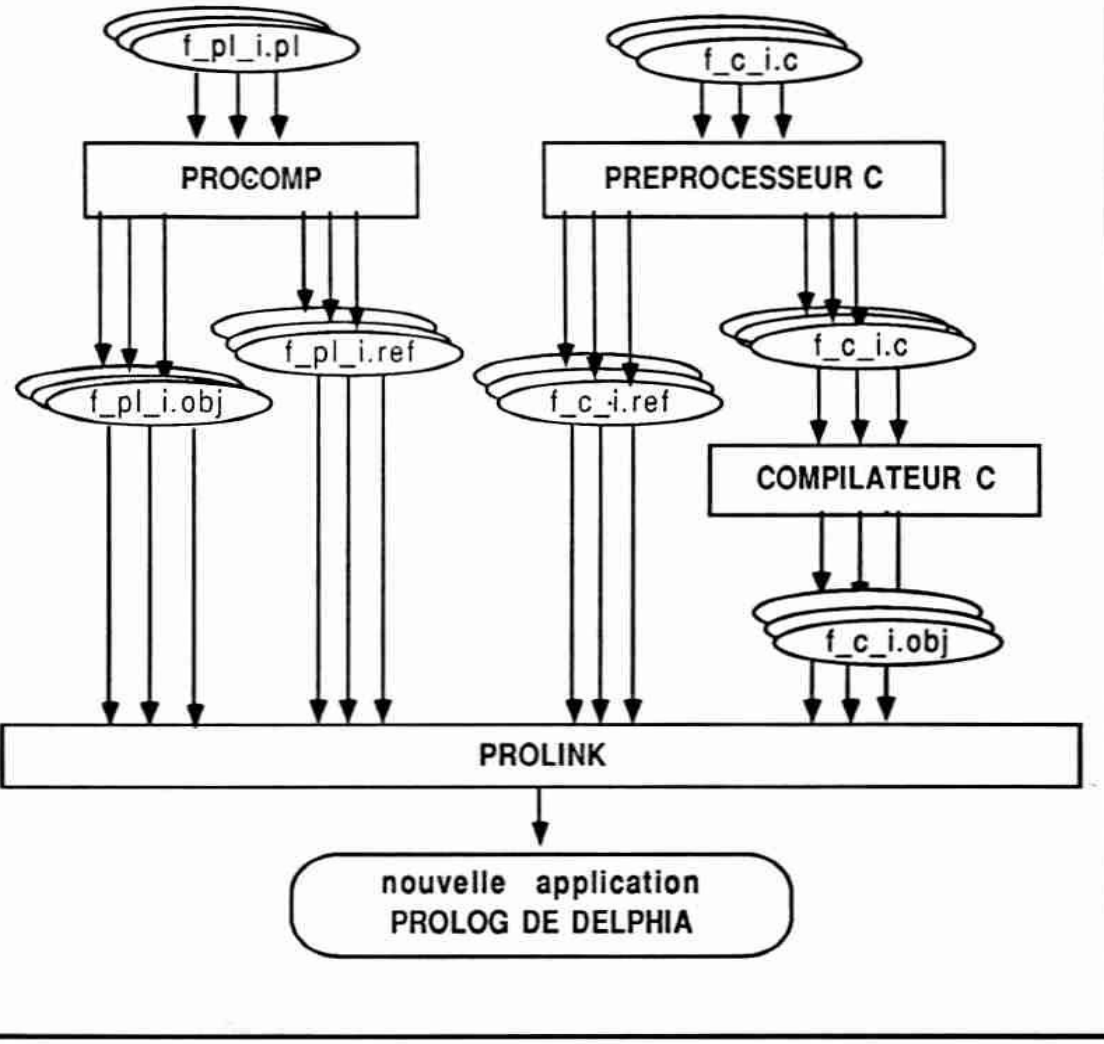
Points de choix

Figure 5 **Prolog de Delphia** **©1986**

Il y a création de points de choix chaque fois qu'une solution (partielle ou totale) est trouvée, alors qu'il existe d'autres possibilités. Les feuilles ***mere(elsa , laurent)***, ***mere(elodie , florent)***, ***pere(armand , lisa)*** de la figure 5 en sont des exemples, alors que ***pere(jean, laurent)*** n'en est pas un puisque c'est la dernière solution de ***parent(Y, Z)***.

Lors d'un échec, comme lorsqu'on ne trouve personne dont laurent soit le mari (2.a), il y a retour-arrière jusqu'au point de choix précédent, pour tenter de satisfaire le but ***pere***.
Toutefois même après un succès, il y a retour-arrière pour rechercher toutes les solutions du but ***pere***.

Aucun point de choix n'est créé dans la partie d'arbre de racine ***mari(Z, X)***, puisque lorsque la variable Z est instanciée, Z ne peut être mari de plusieurs X.



Résumé schématique de l'environnement

Figure 6 **Prolog de Delphia** **©1986**