Diapason : un environnement de développement pour l'intégration d'une entrée vocale dans des applications de type commande de machine

Gilles Souvay

Université de NANCY I
U.F.R. S.T.M.I.A.
Ecole doctorale IAE+M
DFD Informatique

CRIN-CNRS & INRIA-Lorraine

DIAPASON

un environnement de développement pour l'intégration d'une entrée vocale dans des applications de type commande de machine

par

Gilles SOUVAY

Thèse soutenue publiquement le 12 novembre 1992
pour l'obtention du doctorat de l'Université de Nancy I
(mention informatique)

Présidente :	Monique GRANDBASTIEN
Rapporteurs :	Jean CAELEN Jean-Paul HATON Jacques SIROUX
Examinateurs :	Pierre ALINAT Jean-Marie PIERREL

REMERCIEMENTS

Je tiens à remercier ici l'ensemble des personnes qui, par leurs conseils, leurs encouragements et leur participation ont contribué à l'aboutissement de ce travail :

Jean-Marie PIERREL, professeur à l'université de NANCY I, qui m'a accueilli au sein de l'équipe RFIA puis du projet DIALOGUE ; il a été mon directeur de thèse et par ses conseils avisés m'a guidé dans mes recherches et dans la rédaction de ce document ; il a de plus toujours fait le nécessaire pour m'assurer un financement tout au long de ce travail,

Jean-Pierre FINANCE, directeur du CRIN, professeur à l'université de NANCY I, qui m'a accueilli au CRIN,

Monique GRANDBASTIEN, professeur à l'université de NANCY I, qui a bien voulu présider mon jury de thèse,

Jean-Paul HATON, professeur à l'université de NANCY I, qui m'a accueilli avec Jean-Marie au sein de l'équipe RFIA et qui a bien voulu être mon rapporteur interne,

Jacques SIROUX, professeur à l'IUT de LANNION, et Jean CAELEN chargé de recherche au CNRS, qui ont accepté d'être mes rapporteurs externes,

Pierre ALINAT et Evelyne GALLAIS, ingénieurs à THOMSON SINTRA DASM, qui m'ont beaucoup aidé pour l'avancement de ces travaux, pour leur gentillesse et leur disponibilité, pour les nombreuses fois où ils sont venus me chercher ou me ramener aux gares d'ANTIBES et de TOULON, sans oublier la participation de Pierre à mon jury,

Patrick LONG et Pascal DRUART étudiants de DEA, et Jean-Philippe FROT, étudiant d'ESIAL, sans qui une partie de ce travail n'aurait pas existé,

mes amis qui par leur encouragement m'ont permis de continuer et d'arriver au bout de ce travail et plus particulièrement Jean-Pascal,

mes parents qui ont permis par leur soutien moral et financier de poursuivre mes études dont cette thèse est l'aboutissement.

RESUME

Les machines et systèmes que nous utilisons dans la vie courante ou dans la vie professionnelle deviennent de plus en plus sophistiqués et leur commande s'avère d'autant plus complexe. Il serait donc utile de trouver un moyen aisé pour leur faire exécuter les tâches que nous leur demandons.

Une idée simple en apparence, consiste à remplacer la méthode classique, appui sur une série de boutons, choix dans un menu, par une phrase prononcée dans un microphone relié à la machine qui doit être commandée. La parole est un mode de communication rapide et concis, l'homme l'emploie depuis des millénaires. L'utilisateur d'un tel système joue un rôle plus actif et jouit d'une plus grande liberté d'action.

Seulement dans la pratique cela s'avère plus complexe qu'on ne l'imagine. Introduire une composante orale ne consiste pas seulement à greffer une boite noire qui en entrée reçoit de la parole et en sortie renvoie la phrase prononcée, ce problème est déjà relativement complexe à résoudre. La parole porte en elle un éventail de formulations différentes pour un même sens, des possibilités de mauvaises compréhensions, de mauvaises interprétations, d'ambiguïtés, d'imprécisions qui s'ajoutent aux nouvelles erreurs commises par l'opérateur du fait qu'il est moins astreint à suivre un schéma imposé par la machine.

Il faut alors réaliser une interface entre le système de reconnaissance vocale et la machine ou l'application à diriger : le système d'interprétation et de gestion du dialogue. De plus si l'on veut pouvoir réutiliser ce système pour une nouvelle application, il faut définir des outils d'aide à la mise en place des connaissances spécifiques de l'application. L'ensemble formera un environnement de travail, qui dans notre cas tire son nom du système de dialogue : l'environnement DIAPASON.

MOTS-CLES

dialogue oral homme-machine finalisé, commande orale de console, reconnaissance analytique, outils de gestion de dialogue, réseau syntaxico-sémantique.

TABLE DES MATIERES

INTRODUCTION

CHAPITRE 1 PRESENTATION RAPIDE DE L'ETAT DU DOMAINE

1.1 LA COMMUNICATION HOMME-MACHINE 3
1.1.1 Définition 3
1.1.2 Les moyens de la communication 3
1.1.2.1 la communication de l'homme vers la machine 4
1.1.2.2 la communication de la machine vers l'homme 4
1.1.2.3 vers une communication "naturelle" 5

1.2 L'UTILISATION DE LA PAROLE 5
1.2.1 Aspect ergonomique de l'utilisation de la parole 5
1.2.1.1 la performance opératoire 5
1.2.1.2 les contraintes d'utilisation de la parole 6
a) apprentissage de la machine 6
b) les contraintes linguistiques 6
c) les facteurs humains 6
1.2.1.3 la satisfaction du locuteur 7
1.2.1.4 le danger du "tout-vocal" 7
1.2.2 Qualités à viser 7
1.2.2.1 le temps de réponse 7
1.2.2.2 la facilité et la sureté de fonctionnement 7
1.2.2.3 l'intégration ergonomique à l'application 7
1.2.2.4 le coût des systèmes 8

1.3 LA RECONNAISSANCE DE LA PAROLE 8
1.3.1 Les critères de complexité 8
1.3.2 Les méthodes de reconnaissance 9
1.3.3 Les limites des systèmes de reconnaissance 9

1.4 LES SYSTEMES DE DIALOGUE 10
1.4.1 Les connaissances à mettre en œuvre 10
1.4.1.1 les connaissances statiques 10
a) le modèle du langage 10
b) le modèle de la tâche 10
c) le modèle du dialogue 10
d) le modèle de l'utilisateur 11
1.4.1.2 Les connaissances dynamiques 11
a) le contexte de la tâche 11
b) l'historique du dialogue 11
1.4.2 La gestion du canal de communication 12
1.4.2.1 les limites de la reconnaissance 12
a) les phrases non reconnues 12

b) les phrases partiellement reconnues 12
c) le système a reconnu une phrase 13
1.4.2.2 les énoncés de gestion du dialogue 15
1.4.3 La gestion des spécificités du dialogue 15
1.4.4 La gestion de la tâche 16
1.4.5 Les premiers systèmes de dialogue 16
1.4.5.1 le système KEAL [SIROUX 1984] 16
1.4.5.2 les systèmes MYRTILLE 18
1.4.5.3 le système ESOPE 19
1.4.6 Les systèmes de dialogue du CRIN 21
1.4.6.1 le système DIAL 21
1.4.6.2 le système PARTNER 22
1.4.6.3 dialogue avec un robot 23
1.4.6.4 dialogue d'aide pour une standardiste mal-voyante 23
1.4.6.5 le dialogue multimodal dans MULTIWORKS 23
1.4.6.6 le système DIAPASON 24
1.4.7 D'autres systèmes de dialogue 24
1.4.7.1 le système CARAMEL 24
1.4.7.2 le système STANDIA 25
1.4.7.3 le projet ESPRIT SUNDIAL 25
1.4.7.4 le dialogue sous LOIR 26
1.4.7.5 dialogue multimodal avec un robot manipulateur 27
1.4.7.5 le système ICPplan 28
1.4.8 Conclusion sur les systèmes de dialogue 28

CHAPITRE 2 L'ENVIRONNEMENT DIAPASON

2.1 LES OBJECTIFS GENERAUX 29
2.2 LES COMPOSANTES DE L'ENVIRONNEMENT DIAPASON 29
2.2.1 Le système de reconnaissance 30
2.2.1.1 description de la reconnaissance phonétique 30
2.2.1.2 résultats de la détection des phonèmes 31
2.2.1.3 principe de la reconnaissance de mots et de phonèmes 31
2.2.1.4 résultats de la reconnaissance de mots et de phonèmes 31
2.2.2 L'application à diriger 32
2.2.3 Le système de dialogue 32
2.2.4 Les outils de développement du dialogue 33
2.2.5 Rétro-action vers l'utilisateur 33
2.3 LE CADRE DES TRAVAUX REALISES 34
2.4 IMPLEMENTATION DE L'ENVIRONNEMENT DIAPASON 34

CHAPITRE 3 LE DIALOGUE DANS DIAPASON

3.1 INTRODUCTION 35
3.2 LA NOTION D'ORDRE 35
3.3 LA PHASE DE MISE AU POINT 36

3.3.1 L'ordre initial 36
3.3.2 La correction de l'ordre 37
3.3.3 Les messages d'erreur 37
3.3.4 Le complément d'information 38
3.3.5 L'annulation de l'ordre 38
3.3.6 Le changement de sujet 39

3.4 EXECUTION DE L'ORDRE 40

3.5 ENCHAINEMENT DES ORDRES 41

3.6 REPETITION DE L'EXECUTION 42

3.7 ANNULATION OU CORRECTION APRES EXECUTION 42

3.8 LES GRAMMAIRES DU DIALOGUE 43
3.8.1 Notations et conventions 43
3.8.2 Les axiomes des ordres complets 45
3.8.3 Les axiomes de correction 45
3.8.4 Les axiomes d'enchaînement 46
3.8.4.1 Enchaînement sur le même ordre 47
3.8.4.2 Enchaînement sur un ordre différent 47
3.8.5 Les axiomes de réponse 48
3.8.6 Les axiomes système 48
3.8.7 Validité des axiomes 48

CHAPITRE 4 ARCHITECTURE GENERALE DE DIAPASON

4.1 INTRODUCTION 51

4.2 LA STRUCTURE SYNTAXICO-SEMANTIQUE 53

4.3 LE MODULE DE GESTION DU DIALOGUE 54
4.3.1 La structure interne de DIALOGUE 54
4.3.2 Le sous-module MOTEUR 54
4.3.2.1 Introduction 54
4.3.2.2 L'automate d'états finis 54
4.3.2.3 Les requêtes formulées à MEMOIRE 57
4.3.2.4 Le traitement de l'incohérence 59
4.3.2.5 Exécution et incohérence 60
4.3.2.6 Le fonctionnement de MOTEUR - algorithme simplifié 61
4.3.2.7 Fonctionnement de MOTEUR - un exemple 64
4.3.3 Le sous-module MEMOIRE 66
4.3.3.1 Introduction 66
4.3.3.2 La structure interne de MEMOIRE 67
4.3.3.3 Les différents types d'échanges 67
a) les échanges de MEMOIRE vers MOTEUR 67
b) les échanges de MEMOIRE vers RECONN 68
4.4.3.4 La mémoire de DIALOGUE 69
4.4.3.5 La recherche de valeur 70
a) la recherche dans les mémoires 70
b) la valeur par défaut 71

c) l'appel au module de reconnaissance 71
4.4.3.6 Incohérence et mémoire 72
4.4.3.7 Le traitement des phrases de gestion du dialogue 72

4.4 LE MODULE DE GESTION DE LA TACHE 73
4.4.1 Le réseau syntaxico-sémantique 73
4.4.2 Le sous-module INTERFACE 74

4.5 LE MODULE DE GESTION DE LA RECONNAISSANCE 75
4.5.1 La structure interne de RECONN 75
4.5.2 La représentation sémantique de la phrase 76
4.5.3 Le sous-module principal 77
4.5.3.1 la gestion des sous-modules 77
4.5.3.2 les échanges avec DIALOGUE 78
4.5.3.3 la résolution des ambiguïtés 78
4.5.4 L'entrée clavier 78
4.5.5 L'entrée vocale 79
4.5.6 Autres entrées 80

CHAPITRE 5 LES APPLICATIONS DEVELOPPEES

5.1 L'APPLICATION SONAR 81
5.1.1 Le contexte de l'étude 81
5.1.2 Le choix des ordres donnés en vocal 81
5.1.2.1 Les écrans 82
a) la structure générale de l'écran 82
b) l'image initialisation 84
c) l'image veille 84
d) les autres images sonar 87
e) les écrans d'AIDE 87
5.1.2.2 Les pointages 90
5.1.2.3 Les contrôles 91
a) les contrôles de cohérence 91
b) les contrôles à l'exécution 91
5.1.3 Evaluation du système 92
5.1.3.1 la complexité du langage 92
5.1.3.2 les conditions de l'expérimentation 93
5.1.3.3 les mesures ergonomiques 93
a) le premier scénario 94
b) le deuxième scenario 95
c) le troisième scenario 95
d) analyse des résultats 96
5.1.3.4 Démonstration 97
5.1.4 Les résultats acquis 98
5.1.4.1 mode d'emploi de l'entrée 98
5.1.4.2 aspect cognitif de l'avantage à employer une entrée vocale 98
5.1.4.3 aspects ergonomiques de l'avantage à employer une entrée vocale 99
5.1.5 Modification à l'issue des tests 100

5.2 L'APPLICATION DIVA 100

5.2.1 Le contexte de l'étude 100
5.2.2 Présentation de l'application 100
5.2.3 Le choix des ordres donnés en vocal 101
5.2.4 Interfaçage avec le système de dialogue 102
5.2.4.1 l'organisation logique 102
5.2.4.2 l'organisation physique 103
5.2.5 L'évaluation de DIVA Parole 104
5.2.5.1 les difficultés de l'évaluation 104
a) les difficultés au niveau du système de reconnaissance de la parole 104
b) les difficultés au niveau de l'IHM DIVA 104
c) les difficultés au niveau du système de dialogue 104
d) rapport de synthèse 105
5.2.5.2 le scénario 105
5.2.5.3 conclusion 105

CHAPITRE 6 LES OUTILS POUR LE DIALOGUE

6.1 INTRODUCTION 107
6.2 LA PLATE-FORME DE DEVELOPPEMENT 109
6.2.1 Introduction 109
6.2.2 Aspects visuels de la plate-forme 109
6.2.2.1 aspect visuel général 109
6.2.2.2 la fenêtre titre 109
6.2.2.3 la fenêtre application 110
6.2.2.4 la fenêtre dialogue 110
6.2.2.5 la fenêtre clavier 110
6.2.2.6 la fenêtre hypothèses 110
6.2.2.7 exemples 111
6.2.3 Organisation des fichiers sources 112
6.2.3.1 installation minimale du système de dialogue 112
6.2.3.2 installation d'une nouvelle application 113
6.2.3.3 Installation particulière 113
6.2.4 La communication avec l'application 113
6.2.5 Intégration de la maquette de reconnaissance 114
6.2.5.1 la communication de THOMSON vers DIAPASON 114
6.2.5.2 la communication de DIAPASON vers THOMSON 115
6.2.6 L'analyseur syntaxique écrit ASE 116
6.3 L'EDITEUR DE STRUCTURES SYNTAXICO-SEMANTIQUES 116
6.3.1 Introduction 116
6.3.2 Saisie de la grammaire 116
6.3.2.1 présentation de l'écran de saisie 116
6.3.2.2 principes de la saisie 118
6.3.2.3 la représentation graphique des ordres 119
6.3.2.4 Le fichier objet 119
6.4 LE COMPILATEUR 120
6.4.1 Les fichiers pour la reconnaissance 120

6.4.1.1 le fichier grammaire....120
6.4.1.2 le fichier vocabulaire....122
6.4.2 Les fichiers pour la compréhension....122

6.5 LES FORMATEURS....122
6.5.1 Le formateur pour l'analyseur oral....122
6.5.2 Le formateur pour l'analyseur écrit....122

6.6 LA GESTION DE L'HISTORIQUE....124
6.6.1 Les composantes de l'historique du dialogue....124
6.6.2 La structure des mémoires....124
6.6.3 Les opérations sur les mémoires....125

6.7 LA REPRESENTATION DE L'ETAT DE LA TACHE....125
6.7.1 Utilité du contexte de la tâche....125
6.7.2 Etat actuel des travaux....126

CONCLUSIONS ET PERSPECTIVES

ANNEXES TECHNIQUES *diffusion restreinte*

ANNEXE A....LES ORDRES DE L'APPLICATION SONAR

ANNEXE B....LES ORDRES DE L'APPLICATION DIVA

ANNEXE C....LA PLATE-FORME DE DEVELOPPEMENT

ANNEXE D L'EDITEUR DE STRUCTURES SYNTAXICO-SEMANTIQUES

ANNEXE E LE COMPILATEUR DE STRUCTURES SYNTAXICO-SEMANTIQUES

ANNEXE F LE SCENARIO POUR L'EVALUATION DE DIVA PAROLE

TABLE DES FIGURES

CHAPITRE 1 PRESENTATION RAPIDE DE L'ETAT DU DOMAINE

FIGURE 1.1 : LA COMMUNICATION HOMME-MACHINE 3
FIGURE 1.2 : ORGANISATION FONCTIONNELLE DE KEAL 17
FIGURE 1.3 : SCHEMA GENERAL DU SYSTEME MYRTILLE I 18
FIGURE 1.4 : INTERPRETATION D'UNE PHRASE ET DIALOGUE 19
FIGURE 1.5 : STRATEGIE DU SYSTEME ESOPE 20
FIGURE 1.6 : ARCHITECTURE DU SYSTEME DIAL 21

CHAPITRE 2 L'ENVIRONNEMENT DIAPASON

FIGURE 2.1 : LES DIFFERENTES COMPOSANTES DE L'ENVIRONNEMENT DIAPASON 30

CHAPITRE 3 LE DIALOGUE DANS DIAPASON

FIGURE 3.1 : FORME GRAPHIQUE DE L'ORDRE GRAPHIQUE 44

CHAPITRE 4 ARCHITECTURE GENERALE DE DIAPASON

FIGURE 4.1 : ARCHITECTURE GENERALE DE DIAPASON 52
FIGURE 4.2 : STRUCTURE INTERNE DE DIALOGUE 54
FIGURE 4.3 : STRUCTURE INTERNE DE MOTEUR 55
FIGURE 4.4 : STRUCTURE INTERNE DE MOTEUR 55
FIGURE 4.5 : STRUCTURE INTERNE DE MOTEUR 57
FIGURE 4.6 : LES REQUETES FORMULEES A MEMOIRE 58
FIGURE 4.7 : LES ECHANGES DE MEMOIRE VERS MOTEUR 67
FIGURE 4.8 : LES ECHANGES DE DIALOGUE VERS RECONN 68
FIGURE 4.9 : EXTRAIT DU RESEAU SEMANTIQUE POUR L'APPLICATION SONAR 74
FIGURE 4.10 : LA COMMUNICATION ENTRE DIAPASON ET L'APPLICATION 75
FIGURE 4.11 : LE MODULE RECONN 76
FIGURE 4.12 : INTERFACE ENTRE DIAPASON ET LE SYSTEME DE RECONNAISSANCE DE LA PAROLE 79

CHAPITRE 5 LES APPLICATIONS DEVELOPPEES

FIGURE 5.1 : LES DIFFERENTES ZONES DE L'ECRAN 82
FIGURE 5.2 : LE CONTENU DE LA ZONE DIALOGUE 83
FIGURE 5.3 : LES ZONES DE L'IMAGE INIT 85
FIGURE 5.4 : L'IMAGE INIT 85
FIGURE 5.5 : LES ZONES DE L'IMAGE VEILLE 86
FIGURE 5.6 : L'IMAGE VEILLE 86
FIGURE 5.7 : AIDE - ECRAN MENU image INIT 88
FIGURE 5.8 : AIDE - SYNTAXE COMPLETE ordre LIBERER 88
FIGURE 5.9 : AIDE - SYNTAXE DE <FONCTION> ordre LIBERER 89
FIGURE 5.10 : AIDE - SYNTAXE DES ENCHAINEMENTS ordre LIBERER 89
FIGURE 5.11 : AIDE - SYNTAXE DES CORRECTIONS ordre LIBERER 90
FIGURE 5.12 : HISTOGRAMMES 97

FIGURE 5.13 : L'IHM DIVA 101
FIGURE 5.14 : SIMULATION DES ECRANS IHM-D et IHM-G 102
FIGURE 5.15 : ORGANISATION LOGIQUE DE DIVA PAROLE 103
FIGURE 5.16 : ORGANISATION PHYSIQUE DE DIVA PAROLE 103

CHAPITRE 6 LES OUTILS POUR LE DIALOGUE

FIGURE 6.1 : L'ENVIRONNEMENT DIAPASON 107
FIGURE 6.2 : ASPECT VISUEL DE LA PLATE-FORME DE DEVELOPPEMENT 108
FIGURE 6.3 : LA FENETRE DIALOGUE 109
FIGURE 6.4 : LA FENETRE HYPOTHESES 110
FIGURE 6.5 : ASPECT VISUEL DE LA PLATE-FORME DE DEVELOPPEMENT 110
FIGURE 6.6 : LA PHRASE CORRESPONDANT A L'HYPOTHESE 111
FIGURE 6.7 : LA STRUCTURE ENVOYEE PAR LA MAQUETTE 111
FIGURE 6.8 : LA STRUCTURE SEMANTIQUE CORRESPONDANTE 111
FIGURE 6.9 : LES ZONES DE L'ECRAN DE SAISIE D'INTEGRA 116
FIGURE 6.10 : L'ECRAN DE SAISIE D'INTEGRA 116
FIGURE 6.11 : L'ECRAN D'AIDE D'INTEGRA 117
FIGURE 6.11 : LES FICHIERS ISSUS DU COMPILATEUR 119
FIGURE 6.12 : LE FORMATEUR POUR L'ANALYSEUR ECRIT 121

INTRODUCTION

Les machines et systèmes que nous utilisons dans la vie courante ou dans la vie professionnelle deviennent de plus en plus sophistiqués et leur commande s'avère d'autant plus complexe. Il serait donc utile de trouver un moyen aisé pour leur faire exécuter les tâches que nous leur demandons.

Une idée simple en apparence, consiste à remplacer les méthodes classiques (appui sur une série de boutons, choix dans un menu...) par une commande orale via un microphone relié à la machine qui doit être dirigée. La parole est un mode de communication rapide et concis, que l'homme emploie depuis des millénaires. De plus l'utilisateur d'un tel système joue un rôle plus actif et jouit d'une plus grande liberté d'action.

Seulement dans la pratique cela s'avère plus complexe qu'on ne se l'imagine. Introduire une composante orale ne consiste pas seulement à greffer une boite noire qui en entrée reçoit de la parole et en sortie renvoie la phrase prononcée, même si ce problème est déjà relativement complexe à résoudre. La parole porte en elle un éventail de formulations différentes pour un même sens, des possibilités de mauvaises compréhensions, de mauvaises interprétations, d'ambiguïtés, d'imprécisions qui s'ajoutent aux nouvelles erreurs commises par l'opérateur du fait qu'il est moins astreint à suivre un schéma imposé par la machine. Il faut alors réaliser une interface entre le système de reconnaissance vocale et la machine ou l'application à diriger : le système d'interprétation et de gestion du dialogue. Un tel système doit être le plus indépendant possible de l'application. Il faut donc disposer d'outils qui permettent de mettre en œuvre l'ensemble des connaissances nécessaires à une application donnée.

Au cours de notre thèse nous avons défini un modèle de dialogue pour diriger des applications qui sont normalement gérées à l'aide de menus déroulants. Ce modèle utilise un langage de type artificiel mais possédant des consonances naturelles. Nous avons ensuite développé le système de dialogue, appelé DIAPASON, qui permet de gérer le modèle de dialogue proposé. Nous avons défini les interfaces entre le système de dialogue et le système de reconnaissance de la parole d'une part, et entre le système de dialogue et l'application d'autre part. Nous avons développé un ensemble d'outils facilitant la mise en place des connaissances nécessaires à une application. L'ensemble constitue l'environnement de développement DIAPASON.

Un autre aspect de cette thèse a été la réalisation de deux maquettes de démonstration. La première a été développée en collaboration avec un partenaire industriel THOMSON SINTRA D.A.S.M.[1] sur une application simulée inspirée d'une console de sonar construite par la société. THOMSON s'est occupé de la partie reconnaissance de la parole, domaine dans lequelle elle possède une expérience importante et dispose d'une maquette de reconnaissance. Le CRIN-CNRS & INRIA-Lorraine s'est occupé du niveau dialogue et de la simulation de l'application. Cette démonstration a été financée par le ministère de la défense par l'intermédiaire de la D.R.E.T.[2] et a bénéficié des études du GRECO PRC Communication Homme-Machine. La seconde démonstration a porté cette fois sur

---

1 THOMSON SINTRA Division Activité Sous-Marine

2 Direction de la Recherche et des Etudes Techniques

une application réelle. Elle a réuni les mêmes partenaires, augmentés du C.E.R.D.S.M.[1], organisme dépendant de la D.G.A.[2]. Ce nouveau partenaire a fourni l'application qui est un système d'analyse de signaux acoustiques sous-marins.

Le premier chapitre de cette thèse fait une présentation rapide de l'état du domaine.

Le deuxième chapitre présente les différentes composantes de l'ennvironnement DIAPASON.

Le troisième chapitre définit le modèle de dialogue mis en œuvre.

Le quatrième chapitre traite de l'architecture générale du système de dialogue et décrit les différents modules qui le composent.

Le cinquième chapitre présente les deux applications que nous avons développées et le résultat de leur évaluation.

Le sixième chapitre traite des outils de développement de l'environnement DIAPASON.

Cette thèse termine par les conclusions et les perspectives de notre étude.

Notons enfin qu'un second volume de cette thèse regroupe un ensemble d'annexes techniques qui, pour des raisons de confidentialité industrielle, demeure à diffusion restreinte.

---

1 Centre d'Etude et de Recherche en Détection Sous-Marine

2 Direction Générale de l'Armement

Chapitre 1

PRÉSENTATION RAPIDE DE L'ETAT DU DOMAINE

1.1 LA COMMUNICATION HOMME-MACHINE

1.1.1 Définition

La communication homme-machine (par machine nous entendons ordinateur) peut être vue comme un échange d'informations entre l'homme et la machine. De l'homme vers la machine, il s'agit d'informations qui permettent par exemple d'exécuter une tâche, d'interroger la machine, de programmer l'ordinateur... De la machine vers l'homme il s'agit du résultat d'un calcul, de la réponse à une interrogation, d'un message d'alarme...

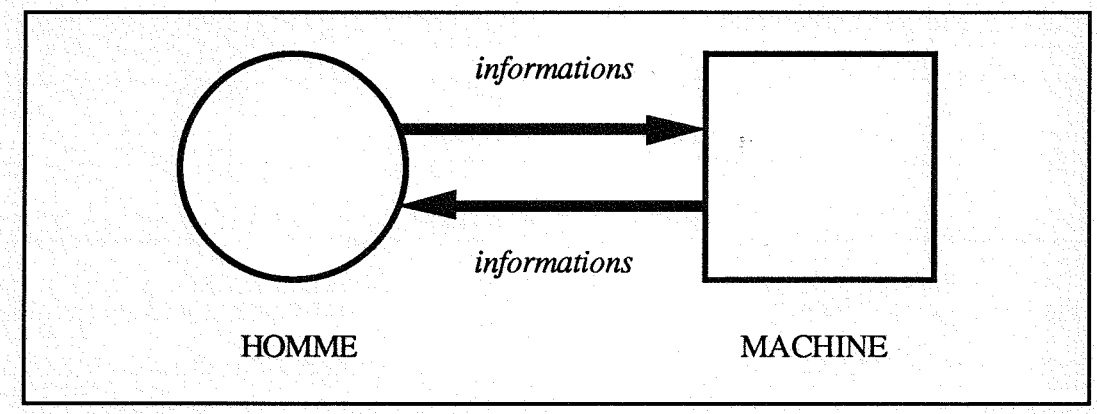


FIGURE 1.1 : LA COMMUNICATION HOMME-MACHINE

1.1.2 Les moyens de la communication

La communication entre l'homme et la machine a beaucoup évolué depuis la première machine jusqu'aux systèmes actuels. Nous présenterons dans ce paragraphe ces évolutions en présentant en premier la communication de l'homme vers la machine puis la communication dans le sens inverse.

1.1.2.1 la communication de l'homme vers la machine

Dans l'évolution de la communication de l'homme vers la machine on peut distinguer deux grandes étapes. Une première étape, où la communication est réservée à des spécialistes, les informaticiens, par qui les utilisateurs (des spécialistes dans un autre domaine : des physiciens, des gestionnaires de grandes entreprises...) sont obligés de passer pour faire travailler l'ordinateur. Au début de l'informatique, l'informaticien, pour communiquer, doit connaître le langage spécifique de la machine. Lorsque les langages de programmation apparaissent, la communication devient indépendante de la machine, mais elle reste toutefois du domaine du spécialiste. Communiquer avec la machine demande des connaissances importantes et donc un apprentissage par une formation très spécialisée.

La deuxième étape commence avec le développement de la micro-informatique, l'ordinateur s'ouvre à un public très large. Il apparaît alors nécessaire de donner à l'utilisateur la possibilité de dialoguer directement avec la machine, et cela avec un minimum de connaissances sur le fonctionnement de l'ordinateur. C'est à cette époque que se développent les interfaces logicielles [COUTAZ 1990]. Au départ le clavier est le seul moyen de communication, il suffit de taper sur une série de touches, de pousser un bouton, pour afficher un menu et choisir l'action à déclencher ou le traitement à effectuer. Ensuite, avec l'apparition des périphérique de désignation (souris et écrans tactiles pour le grand public, gant de désignation pour des utilisations professionnelles), l'utilisateur a la possibilité de désigner des zones de l'écran. Enfin en partant de la constatation que le moyen de communication le plus spontané entre deux personnes est la parole, l'utilisation du langage humain dans sa forme écrite (clavier ou écriture manuscrite) ou dans sa forme orale fait son apparition. Tous ces moyens de communication constituent un ensemble de média (on parle de communication multimedia). Dans un premier temps ces media étaient utilisés séparément en prenant le plus approprié pour la tâche à réaliser. La tendance actuelle est de les utiliser de façon concomitente afin d'obtenir une efficacité maximale et d'avoir plus de spontanéité dans l'expression des demandes. On parle alors de communication multimode [COUTAZ & CAELEN 1990].

Progressivement la communication, qui s'effectuait dans le cadre rigide de langages de programmation et donc de procédures informatiques, est passée à des moyens plus souples, plus "naturels". Ces nouveaux modes de communication introduisent des incertitudes au niveau des objets désignés, des phrases écrites ou des paroles prononcées, posent le problème de la validité des informations à plus ou moins long terme, laissent plus d'initiative à l'utilisateur qui a de moins en moins besoin de suivre un schéma rigoureux imposé par la machine.

1.1.2.2 la communication de la machine vers l'homme

Parallèlement les moyens de communication de la machine vers l'homme se diversifient. La communication s'appuie sur l'utilisation croissante des sens humains.

La **vue** est le moyen le plus ancien : l'homme constate que la machine est dans l'état demandé, un voyant lumineux s'allume, le résultat du calcul est imprimé sur papier, il est écrit à l'écran, la machine affiche l'image demandée...

L'**ouïe** aura tendance à être exploitée de plus en plus souvent. Au départ un bip sonore indiquait l'arrivée d'un résultat, la détection d'une erreur. Avec la multiplication des périphériques et de leur capacité de stokage, la machine peut alors faire écouter l'information demandée s'il s'agit d'un enregistrement sonore. Enfin avec le développement de l'utilisation de la parole, la machine peut répéter ce qu'elle a entendu ou compris, ou encore interroger l'homme.

Un troisième sens commence à être utilisé : le **toucher**. Il consiste en des périphériques qui donnent l'impression de pression pour simuler la préhension, le contact...

On passe progressivement du monde réel où l'homme interagit avec le monde physique à un monde virtuel où les sens sont activés par la machine.

1.1.2.3 vers une communication "naturelle"

La tendance actuelle dans le domaine de la communication homme-machine est de permettre à l'utilisateur de formuler ses requêtes en langage "naturel" [CARRE 1991]. On distingue plusieurs niveaux de communication :

- la communication par mots-clés qui consiste à rechercher dans la phrase les mots porteurs de sens,
- la communication en langage artificiel où non seulement les mots sont recherchés mais aussi leur organisation syntaxique pour en déduire un sens,
- la communication en langage naturel où l'homme donne une phrase quelconque de la langue semble à l'heure actuelle utopique, les études se limitent pour l'instant au langage naturel restreint à un domaine d'expertise.

1.2 L'UTILISATION DE LA PAROLE

L'utilisation de la parole dans la communication humaine est naturelle et ne pose pas, pour la plupart d'entre-nous, de difficultés majeures. Il en va différemment lorsque la machine a la possibilité de comprendre le langage humain. Ce paragraphe présentera l'apport de la parole, les contraintes qu'elle entraîne et les critères à respecter pour obtenir une intégration efficace de la parole dans un système informatique.

1.2.1 Aspect ergonomique de l'utilisation de la parole

1.2.1.1 la performance opératoire

La parole joue un grand rôle dans la performance opératoire de l'homme lorsqu'il est en face d'une machine disposant d'une entrée vocale :

- c'est un mode **rapide et concis**, l'opérateur n'a plus à réfléchir au moyen de déclencher une série d'actions pour obtenir le résultat souhaité,
- elle amène alors à réaliser des **interfaces plus naturelles** qui ne présentent que les données pertinentes de la tâche en se passant d'une partie de l'habillage logiciel,

- elle **libère les mains** de la saisie ou du choix dans les menus, elles peuvent être utilisées pour des actes plus en lien avec la tâche,
- elle permet au **regard** de rester **fixé sur l'endroit de l'écran** où se déroule l'action demandée, le va-et-vient des yeux sur différentes parties de l'écran est nettement diminué.

Elle apporte globalement un confort d'utilisation et de souplesse si l'intégration est correctement réalisée (§1.2.2). Il faut néanmoins tenir compte de situations exceptionnelles, par exemple en situation d'alarme, où ses performances peuvent se dégrader de manière importante.

1.2.1.2 les contraintes d'utilisation de la parole

La parole introduit des contraintes d'utilisation dont il faut tenir compte avant de la mettre en œuvre pour la communication avec l'ordinateur. Il s'agit principalement de l'apprentissage de la machine, de contraintes linguistiques et de facteurs humains.

a) apprentissage de la machine

La machine doit être capable de reconnaître les paroles de la personne qui s'adresse à elle. L'idéal serait qu'il n'y ait pas de d'apprentissage de la part de l'ordinateur qui reconnnaît spontanément toute personne qui s'adresse à lui. Si ce n'est pas possible, il faut alors que la phase d'apprentissage soit la plus courte possible, l'idéal étant peut-être que le locuteur prononce une phrase unique qui permette d'acquérir les caractéristiques de sa voix. Un système où l'utilisateur aurait à prononcer tous les mots ou toutes les phrases de l'application ne pourrait pas être accessible au grand public.

b) les contraintes linguistiques

Il existe des contraintes linguistiques à l'introduction de la parole. Elles concernent la taille du vocabulaire, c'est-à-dire le nombre de mots que le système aura à reconnaître. Une autre contrainte linguistique concerne la structure syntaxique des phrases : seront-elles issues d'un langage artificiel ou du langage naturel ? Enfin quel sera le mode d'élocution de l'opérateur humain ? Parlera-t-il en mots isolés, en mots enchaînés, ou en parole continue ?

c) les facteurs humains

Dans les contraintes d'utilisation le facteur humain se révèle être prépondérant. Il existe des facteurs physiques. L'introduction de la parole demande l'ajout d'un microphone et éventuellement d'un casque qui peuvent apporter une gêne. Il existe des facteurs psychologiques de même nature que lors de l'arrivée des micro-ordinateurs, où des personnes ont présenté un blocage voire un rejet du nouvel arrivant. Il y a la peur d'être soi-même rejeté par la machine. C'est une réaction qui n'est pas rare lorsqu'un groupe vient assister à une démonstration où il y a une entrée vocale, certaines personnes refusent de parler dans le microphone de peur de perdre la face devant les autres membres du groupe. Néanmoins ces facteurs devraient disparaître quand l'utilisation de la parole sera plus répandue.

L'apprentissage de l'utilisateur est une contrainte supplémentaire. Il englobe les aspects présentés auparavant mais vus du coté de l'homme. Cela concerne par exemple, le nombre de mots à retenir ou les tournures syntaxiques à connaître, le rythme des paroles...

1.2.1.3 la satisfaction du locuteur

L'utilisation de la parole doit amener la satisfaction du locuteur. Elle ne doit pas être une gêne mais une aide à la communication avec la machine. Elle ne pourra être atteinte qu'en minimisant les contraintes et en respectant des critères de qualité §1.2.2.

1.2.1.4 le danger du "tout-vocal"

Le danger qui apparaît lorsqu'on introduit une entrée orale dans un système est de prévoir son utilisation pour toutes les actions que l'opérateur aura à réaliser. C'est la tentation du "tout vocal". La parole doit être un complément aux autres média et ne doit en aucun cas remplacer des procédures plus adaptées à un autre média, par exemple la désignation d'un objet ou son déplacement doivent continuer à se faire à la souris.

1.2.2 Qualités à viser

Lorsque la parole est le vecteur de la communication homme-machine, il existe des critères de qualité à respecter pour la mettre en œuvre dans un système.

1.2.2.1 le temps de réponse

Un des critères de qualité prépondérant est le temps de réponse du système doté de l'entrée vocale. Il faut que la réponse soit immédiate, dans le cas contraire l'opérateur risque d'avoir l'impression de perdre du temps et délaissera la parole au profit d'une autre procédure qui pourra être plus longue globalement. Il préférera cette dernière car il n'aura pas de phase d'attente.

1.2.2.2 la facilité et la sureté de fonctionnement

Il faut que la parole soit facile à utiliser, pour cela il faut minimiser le nombre de contraintes imposées.

Il faut que les performances soient stables. Un locuteur qui serait enroué ou en état de stress par exemple, devrait pouvoir être reconnu. Si le bruit ambiant devient important, il faut que le système puisse s'adapter aux nouvelles conditions ou se rendre compte qu'il n'est plus performant. C'est pourquoi il faut prévoir un mode de remplacement de la parole.

1.2.2.3 l'intégration ergonomique à l'application

L'intégration de la parole dans une application doit être faite avec attention. En aucun cas elle ne doit compliquer l'utilisation du système. Elle ne doit pas être imposée pour des actions qui sont plus efficaces avec un autre média, par exemple

la désignation avec la souris. Elle doit être de même nature que le mode classique de fonctionnement, l'utilisateur ne dois pas avoir à se poser la question : «Je suis en mode vocal ou non...?» avant d'effectuer une action.

1.2.2.4 le coût des systèmes

La parole doit enfin avoir une influence négligeable ou limitée sur le coût du système dans lequel elle est intégrée. Il ne faut pas que les utilisateurs potentiels d'un système doté d'une entré vocale achètent de la parole mais bien une machine dotée d'une entrée vocale.

1.3 LA RECONNAISSANCE DE LA PAROLE

Il existe de nombreux systèmes dont le but est la reconnaissance de la parole. Il est possible de trouver des critères qui permettent de classer ces systèmes du point de vue de la complexité des paroles à reconnaître et des méthodes mises en œuvre pour le faire [CALLIOPE 1989], [HATON 1991].

1.3.1 Les critères de complexité

Il existe des critères de complexité qui sont liés aux paroles à reconnaître.

Un système peut être **mono-locuteur** (il ne peut reconnaître qu'un seul locuteur) ou **multi-locuteur** (la reconnaissance est indépendante du locuteur). La reconnaissance multi-locuteur s'avère plus difficile à réaliser car il faut non seulement tenir compte de la variablilité intra-locuteur (émotions, stress, rhume...), mais aussi de la variabilité inter-locuteur (accents régionaux, caractéristiques anatomiques...).

La reconnaissance peut porter sur des **mots isolés**, le locuteur marque des pauses entre chaque mot. Elle peut porter sur des **mots enchaînés**, Dans ce cas le système ne possède plus les marques prosodiques fortes des pauses pour analyser la phrase et les mots peuvent subir des altérations en début et en fin (coarticulation).

La **taille du vocabulaire** est un paramètre important, il est plus facile de reconnaître un mot parmi une dizaine que parmi plusieurs milliers. La taille n'est pas le seul facteur au niveau du vocabulaire, il faut en plus tenir compte de la proximité phonétique des mots du vocabulaire.

Le **langage**, support des paroles de l'utilisateur, pourra être régi par des règles syntaxiques fortes dans le cadre de langages artificiels, ce qui facilitera la reconnaissance, ou offrir une plus grande liberté avec des langages naturels qui augmenteront la complexité de la reconnaissance.

L'**environnement sonore** dans lequel se trouve le locuteur est un facteur important pour le classement des systèmes :

- robustesse du système aux bruits qui dégradent en général fortement les performances,
- prise en compte des variations phonétiques de la parole dans un environnement bruité (effet LOMBARD) [ANGLADE 1990].

1.3.2 Les méthodes de reconnaissance

Les systèmes de reconnaissance peuvent être classés suivant les méthodes qu'ils mettent en œuvre.

Il peut s'agir de **méthodes globales** de reconnaissance, qui utilisent des techniques de comparaison de mots avec des portions de la phrase, par opposition à des **méthodes analytiques** où l'on recherche dans la phrase des entités élémentaires qui seront traités par un niveau supérieur.

Il peut s'agir de **systèmes auto-organisateurs** qui s'appuient sur des méthodes mathématiques telles que la programmation dynamique ou la modélisation Markovienne, par opposition à des **systèmes à base de connaissances** qui s'appuient sur des connaissances acoustiques, phonétiques ou linguistiques.

Il existe deux grandes étapes dans la reconnaissance de phrases :

- une étape **perceptive** où il faut reconnaître les sons (méthodes globales) ou les phonèmes (méthodes analytiques). Dans les deux cas, les systèmes pourront être de type auto-organisateur ou à base de connaissances.
- une étape de **reconnaissance de phrases.** Dans le cas de langages artificiels (langages entièrement décrits par leur syntaxe), les systèmes utilisent des méthodes guidées par la syntaxe comme dans MYRTILLE I [PIERREL 1978], ESOPE [MARIANI 1978] ou KEAL [MERCIER 1977]. Dans le cadre de langages plus naturels, ils font appel à une coopération entre différents processus syntaxiques et sémantiques tel DIAL [CARBONNEL 1986].

1.3.3 Les limites des systèmes de reconnaissance

A l'heure actuelle, il existe quelques systèmes de reconnaissance de la parole qui donnent satisfaction pour des petits et moyens vocabulaires. On peut citer par exemple le système de reconnaissance DATAVOX (monolocuteur, méthodes globales, mot enchaînés, 200-300 mots) commercialisé par VECSYS et réalisé à partir d'études menées au LIMSI, et la maquette de reconnaissance de la parole développée par THOMSON SINTRA DASM (multilocuteur, méthodes analytiques, mots enchaînés, 150 mots) [ALINAT 1987]. Lorsque l'on veut intégrer de tels systèmes dans une application pour d'obtenir un ensemble opérationnel, on s'aperçoit rapidement qu'ils s'avèrent nettement insuffisants. Reconnaître les phrases est nécessaire, mais ce n'est pas suffisant. Il faut prendre en compte des informations qui ne sont pas présentes dans l'énoncé courant. En particulier il faut pouvoir interpréter le résultat de la reconnaissance en fonction des énoncés précédents et du contexte courant de la tâche. C'est donc naturellement que les premiers travaux qui portaient sur des systèmes de reconnaissance de la parole se sont orientés vers des études sur les systèmes de dialogue.

1.4 LES SYSTEMES DE DIALOGUE

1.4.1 Les connaissances à mettre en œuvre

Les connaisances à mettre en œuvre dans un système de dialogue oral sont nombreuses. Elles peuvent être regroupées en deux grandes catégories : les connaissances statiques qui ne varient pas au cours du temps et les connaissances dynamiques qui sont liées à l'aspect évolutif de la tâche et du dialogue [PIERREL 1987].

1.4.1.1 les connaissances statiques

Les connaissances statiques interviennent au niveau de chacune des composantes du système de dialogue. Chaque niveau possède ses connaissances spécifiques, elles se présentent sous forme de modèles.

a) le modèle du langage

Le modèle du langage est la description des énoncés possibles par le locuteur en terme de syntaxe et de vocabulaire. Il intervient au niveau de la reconnaissance (description phonétique des mots, description syntaxique des phrases), au niveau de l'interprétation des énoncés (utilisation des ellipses et des anaphores) et au niveau génération des réponses du système.

Il est important de noter que le niveau génération doit utiliser le même modèle de langage. Il semble en effet que le locuteur a tendance à s'adapter au niveau de langage de la machine [GRECO 1985].

b) le modèle de la tâche

Le modèle de la tâche comprend deux aspects :

- une définition des objets manipulés et des relations qui existent entre-eux : opérations réalisables sur un objet, les conditions de validité d'une opération, ...,
- une définition en terme de buts ou de sous-buts qui spécifient les chemins d'accès aux données, fonctions ou renseignements de l'application.

C'est au niveau du modèle de la tâche que sont gérées les valeurs par défaut proposées au système de dialogue.

c) le modèle du dialogue

Le modèle du dialogue définit les stratégies mises en œuvre pour l'interprétation d'un énoncé :

- les stratégies de fonctionnement selon le type de la phrase (affirmation, confirmation, annulation, correction, réponse à une question) et selon que le type est attendu ou non (confirmation intempestive, non réponse à une question posée...),
- les stratégies de recherche dans les sources de connaissances (mémoire du dialogue, proposition de valeurs par défaut, interrogation du locuteur),
- les stratégies de détermination des interventions du système (relance du locuteur, demande de confirmation).

d) le modèle de l'utilisateur

Le modèle de l'utilisateur décrit les particularités du locuteur.

Il y a tout d'abord les connaissances liées à la reconnaissance et à l'interprétation de la phrase :

- les caractéristiques acoustico-phonétiques du locuteur (homme/femme, prosodie, phonèmes, ...),
- le vocabulaire/syntaxe du locuteur selon son niveau de connaissance de l'application ou de la langue.

Il y a ensuite les droits d'accès du locuteur afin que seuls les utilisateurs autorisés puissent effectuer des opérations sensibles.

1.4.1.2 Les connaissances dynamiques

a) le contexte de la tâche

Le contexte de la tâche a un rôle important à jouer. Il permet d'orienter les interprétations d'un énoncé dans une direction en acceptant, par exemple, les phrases compatibles avec l'état courant de la tâche. Il joue un rôle dans la prédiction en limitant les énoncés possibles à un moment donné. Il aide à déterminer la valeur par défaut en relation avec le modèle de la tâche.

b) l'historique du dialogue

L'historique du dialogue est la mémoire du dialogue. Il sert d'organisation logique du discours. S'il n'existait pas, il serait impossible au système de dialogue de résoudre les tournures elliptiques ou les références anaphoriques qui sont caractéristiques de la langue.

La durée de vie des informations contenues dans l'historique est variable, elle peut être égale à la durée d'une session d'utilisation ou limitée à quelques énoncés (on évite ainsi certaines résolutions elliptiques incompatibles avec l'état courant de la tâche).

Il peut contenir l'origine de la phrase ou des portions de phrases dans le cas d'un système multimédia. Le système peut ainsi accorder plus de poids à une interprétation plutôt qu'à une autre.

L'historique du dialogue se divise en deux composantes :

- la **mémoire à court terme** conserve des énoncés partiels : affirmation, réponses aux questions, corections...,
- la **mémoire à long terme** conserve des énoncés complets qui sont le résultat de l'interprétation des énoncés partiels.

Il en résulte que la durée de vie des informations dans la mémoire à court terme sera plus faible que celles contenues dans la mémoire à long teme, il sera courant de vider la mémoire à court terme lorsqu'une nouvelle interprétation aura été réalisée et mise dans la mémoire à long terme.

1.4.2 La gestion du canal de communication

1.4.2.1 les limites de la reconnaissance

La gestion du canal de communication doit tenir compte des limites actuelles des systèmes de reconnaissance. En effet le taux de reconnaissance des bas niveaux n'étant pas de 100%, il existe donc des incertitudes quant à la phrase reconnue et transmise aux niveaux supérieurs. On peut distinguer différents cas de figures et essayer de gérer cette situation.

a) les phrases non reconnues

Dans le cas des phrases non reconnues, les origines possibles sont :
- la phrase ne fait pas partie du langage,
- le locuteur a parlé trop faiblement / fortement, trop loin / près du microphone,
- un bruit extérieur est venu couvrir la phrase.

Pour résoudre ce problème, le système émet une demande de répétition. Les réponses que l'on pourrait envisager pour chaque cas seraient :

« *Je n'ai pas compris, pourriez-vous répéter s'il vous plait !* »

« *Je n'ai pas compris, parlez plus fort s'il vous plait !* »...

« *Je n'ai pas entendu, pourriez-vous répéter s'il vous plait !* »

b) les phrases partiellement reconnues

Dans le cas des phrases partiellement reconnues, on peut envisager comme origine possible de la mauvaise reconnaissance différents cas :
- un mot n'appartient pas au vocabulaire connu du système,
- un bruit extérieur a couvert une portion de la phrase,
- il y a une hésitation du niveau reconnaissance entre plusieurs mots.

La gestion de ces cas de figure dépendra fortement des parties bien reconnues de la phrase. Si le système est capable de construire une interprétation partielle de l'énoncé, il lui suffira ensuite de demander au locuteur de repréciser les éléments manquants. Le danger de ce genre de situation est le bouclage, si le locuteur continue par exemple à utiliser le mot inconnu du système.

EXEMPLE 1.1 : mot couvert par du bruit
Locuteur : « *Prends le cube* ###### »
→ détection du bruit
Système : « *Quel cube dois-je prendre ?* »

EXEMPLE 1.2 : mot inconnu
Locuteur : « *Prenez le cube vermillon* »
→ non-détection du mot inconnu
Système : « *Quel cube devons-nous prendre ?* »
Locuteur : « *Le cube vermillon* »
Système : « *Quel cube devons-nous prendre ?* »

⇒ bouclage
Il faut donc envoyer un message plus explicite du genre :
Système : « *Je n'ai pas compris quel cube nous devons prendre ?* »
Système : « *Je ne connais pas le mot précisant le cube que nous devons prendre* »

EXEMPLE 1.3 : choix entre plusieurs mots
Locuteur : « *Prends le cube rouge* »
→ "Prends le cube rouge|rose"
Si le système n'arrive pas à faire la différence entre "rouge" et "rose", il peut se retrouver en situation de blocage.
Système : « *Je prends le cube rouge ou le rose ?* »
Locuteur : « *Le rouge* »
→ "Le rouge|rose"
Système : « *Je prends le cube rouge ou le rose ?* »
⇒ bouclage
Ce n'est donc pas de cette manière qu'il faut procéder. Une solution plus judicieuse est de proposer une solution et la faire confirmer. Nous reviendrons sur cette notion dans le paragraphe suivant et l'exemple 1.4.
Le contexte de la tâche peut jouer un rôle important à ce niveau, il peut permettre d'éliminer une des deux hypothèses. Si l'on suppose qu'il n'existe pas de cube rose, il n'y aura plus d'hésitation possible.
Locuteur : « *Prends le cube rouge* »
→ "Prends le cube rouge|rose"
Système : « *J'ai pris le cube rouge* »

c) le système a reconnu une phrase

Dans le cas où le système a reconnu une phrase, on peut se demander si **la phrase reconnue est bien la phrase prononcée**, ou si l'interprétation finale de la machine correspond bien à la volonté du locuteur. Il faut dans ce cas prévoir la possibilité de confirmer ou de contester les interprétations du système, en prenant garde de ne pas alourdir inutilement le dialogue.

EXEMPLE 1.4 : dialogue avec une confirmation systématique
Locuteur : « *Prends le cube rouge* »
Système : « *Dois-je prendre le cube rouge ?* »
Locuteur : « *Oui* »
Système : « *J'ai pris le cube rouge* »
Locuteur : « *Prends la sphère rose* »
Système : « *Dois-je prendre la sphère rouge ?* »
Locuteur : « *Non la rose !* »
Système : « *Dois-je prendre la sphère rose ?* »
Locuteur : « *Oui* »

EXEMPLE 1.5 : dialogue sans confirmation
Locuteur : « *Prends le cube rouge* »

Système : « *J'ai pris le cube rose* »
Locuteur : « *Non le rouge !* »
Système : « *J'ai pris le cube rouge* »

Le système peut reconnaître une **phrase incohérente**. L'incohérence peut se situer, soit par rapport à l'état courant de la tâche (par exemple manipulation d'objets inexistants), soit par rapport à l'état courant du dialogue (par exemple oubli de confirmer une action). On peut distinguer plusieurs origines à cette incohérence :

(1) le système de reconnaissance de la parole a mal reconnu la phrase prononcée,
(2) le locuteur a prononcé une phrase incorrecte dans l'état courant de la tâche,
(3) le système de dialogue a mal interprété une phrase bien reconnue.

Le système de dialogue doit pouvoir déterminer le cas de figure dans lequel il se trouve. Une solution qui peut être mise en œuvre est de se fier au score de reconnaissance global de la phrase ou au score des mots qui composent cette phrase. On fixe un seuil, si le score est supérieur à ce seuil, le système considère que la phrase ou le mot est bien reconnu (cas 2 et 3), si le score est inférieur, il considère que la phrase ou le mot est mal reconnu (cas 1). La distinction entre le cas 2 et 3 semble difficile voire impossible et révèle certainement dans le cas 3 une ambiguïté ou une erreur dans la conception de l'interpréteur.

EXEMPLE 1.6 : incohérence par rapport à la tâche
Locuteur : « *Prends le cube rouge* »
→ "Prends le cube rose"
Le système selon les cas enverra un de ces mesages :
Système : « *Il n'y a pas de cube rose* » incohérence du locuteur (bon score de la phrase),
Système : « *Je n'ai pas compris* » mauvaise reconnaissance (score de la phrase insuffisant),
Système : « *Je n'ai pas compris la couleur du cube que vous vouliez prendre* » (score du mot rose insuffisant).

EXEMPLE 1.7 : incohérence par rapport au dialogue
Locuteur : « *Prends le cube* »
Système : « *Quel cube dois-je prendre ?* »
Locuteur : « *Pose-le sur la table* »

Le système peut émettre **plusieurs hypothèses** pour une même phrase. Il aura alors à choisir la plus cohérente vis-à-vis de la tâche et du dialogue. La réponse pourra tenir compte ou non de son choix en exigeant alors une confirmation.

EXEMPLE 1.8 :
Locuteur : « *Prends le cube* »
Hypothèse 1 → "Prends le cube rose" incohérente

Hypothèse 2 → "Prends le cube rouge" cohérente
Si le système se rappelle qu'il a fait un choix :
Système : « *Dois-je prendre le cube rouge ?* »
Si le système oublie qu'il a fait un choix :
Système : « *J'ai pris le cube rouge* »

1.4.2.2 les énoncés de gestion du dialogue

Un autre aspect de la gestion du canal de communication concerne les énoncés de gestion du dialogue. Il s'agit :

- des messages de relance pour faire parler le locuteur :
  « *Etes-vous toujours en ligne ?* »
  « *Pourriez-vous répondre à ma question ?* »,
- des messages de mise en attente lorsque le traitement demandé est long :
  « *Veuillez patienter quelques instants s'il vous plait* »,
- des messages de maintient du dialogue pour combler les silences et pour encourager l'utilisateur à s'exprimer :
  « *Continuez, je vous écoute* ».

1.4.3 La gestion des spécificités du dialogue

A partir du moment où il y a dialogue il va y avoir, dans les propos de l'un des interlocuteurs, des références à ses paroles antérieures ou à celles de son vis-à-vis. C'est ainsi qu'apparaissent les phénomènes d'ellipses et d'anaphores qui permettent de réduire la longueur des échanges.

L'**ellipse** permet de faire référence, sans qu'il y ait de mots pour la réaliser, elle correspond à des phrases incomplètes.

EXEMPLE 1.9 : exemple d'ellision
Locuteur : « *Quel temps fait-il à Epinal ?* »
Système : « *Il fait beau* »
Locuteur : « *Et à Nice ?* »
→ "Quel temps fait-il à Nice"
Système : « *Il pleut* »

L'**anaphore** permet de faire des références qui cette fois-ci sont marquées linguistiquement par des mots, il s'agit souvent de références pronominales simples.

EXEMPLE 1.10 : exemple d'anaphore
Locuteur : « *Prends le cube rouge* »
Système : « *Je **le** tiens déjà* »
→ "Je tiens déjà le cube rouge"

La **déitique** est un autre type de référence qui apparaît dans le dialogue. La référence fait partie de la situation de communication au niveau du locuteur (je, tu, il, ...), du temps (hier, maintenant, ...), du lieu (ici, là, ...)...

EXEMPLE 1.11 : exemple de déitique
Locuteur : « *Prends le cube rouge* »
...
Locuteur : « *Pose le ici* »

Un phénomène qui peut se produire dans le dialogue est l'apparition d'**ambiguïtés**, par exemple l'interlocuteur ne peut pas choisir l'objet de la référence. L'ambiguïté peut être résolue en tenant compte par exemple du contexte de la tâche qui permet d'éliminer une des hypothèses. Si ce n'est pas possible, le choix peut être fait arbitrairement par la machine associé à une confirmation. L'ordinateur peut aussi demander à l'homme de faire le choix.

Dans tous les cas, ellipses, anaphores, ambiguïté, seule une interprétation contextuelle permet de résoudre les phénomènes liées aux spécificités du dialogue.

1.4.4 La gestion de la tâche

Un système de dialogue homme-machine doit assurer la gestion de la tâche qu'il dirige. Les fonctions réalisées sont :

- la détection des buts de l'utilisateur et la détermination de la nature de la tâche à exécuter (il s'agit d'une demande de renseignement ou de l'exécution d'une commande),
- la négociation avec l'utilisateur des buts et des sous-buts lorsque le système ne possède pas tous les éléments nécessaires à l'accomplissement de la tâche, où lorsqu'il détecte une incohérence,
- l'exécution de la tâche, une fois que toutes les informations nécessaires sont réunies,
- la détection des limites de ses compétences pour déterminer les demandes qui peuvent être traitées.

1.4.5 Les premiers systèmes de dialogue

Les premiers systèmes de dialogue qui apparaissent en France sont des sytèmes qui du point de vue dialogue sont très pauvres et qui sont très spécifiques à l'application qu'ils dirigent. L'expérience acquise avec ces premiers systèmes a permis de donner naissance à une seconde version de système où le dialogue prenait une part plus importante.

1.4.5.1 le système KEAL [SIROUX 1984]

Les premières études au CNET sur les systèmes de dialogue ont commencé en 1974. Deux applications furent choisie : la description et la vérification de tracés en conception assistée par ordinateur et un mini-centre de renseignement téléphonique [BUISSON 1976]. Le système mis en œuvre pour la première application était pauvre du point de vue dialogue car seules la confirmation des messages qu'il avait compris et la possibilité de correction d'une mauvaise reconnaissance étaient prise en compte. La seconde application [VIVES 1978] a permis de mettre en œuvre un dialogue dirigé par la machine à l'aide d'un automate,

l'utilisateur devant répondre le plus simplement possible aux questions de la machine. La réponse est analysée par un système de reconnaissance de mots isolés.

La poursuite de ces travaux a porté sur l'assouplissement des contraintes imposées à l'utilisateur et a donné naissance au système KEAL [MERCIER 1977], l'application visée étant toujours le mini-centre de renseignement téléphonique. Le dialogue reste dirigé par la machine à l'aide d'un automate, seul l'énoncé initial du locuteur est libre.

A partir de 1979, les études vont porter sur un système complet intégrant des modules d'analyse syntaxique et sémantique et un module de dialogue plus sophistiqué [NOUHEN&SIROUX 1977][FIGURE 1.2]. Le module de gestion du dialogue n'est plus un simple automate, il permet de gérer un dialogue coopératif de type question/réponse pour l'accomplissement d'une tâche particulière. Le contrôle du dialogue par le système demeure discret, tant que cela est possible (bonne reconnaissance, pas de problème de formulation). Parmi les énoncés du locuteur, seuls sont pris en compte ceux concernant la tâche à réaliser. L'utilisateur peut fournir des informations non demandées et le système peut, de son coté, effectuer des inférences sur les énoncés et éviter ainsi des phases de dialogue. Ces recherches ont débouché sur le logiciel CADI (Constructeur Automatique de Dialogue Intelligent) [NOUHEN & SIROUX 1981] qui utilise une modélisation du dialogue sous forme de phrases. Une phrase est une arborescence dont chaque nœud représente une situation de dialogue (début du dialogue, information à acquérir...) et dont le processus autorise des effets de focalisation. Dans ce système les principes du dialogue sont fortement intégrés à la description de la tâche.

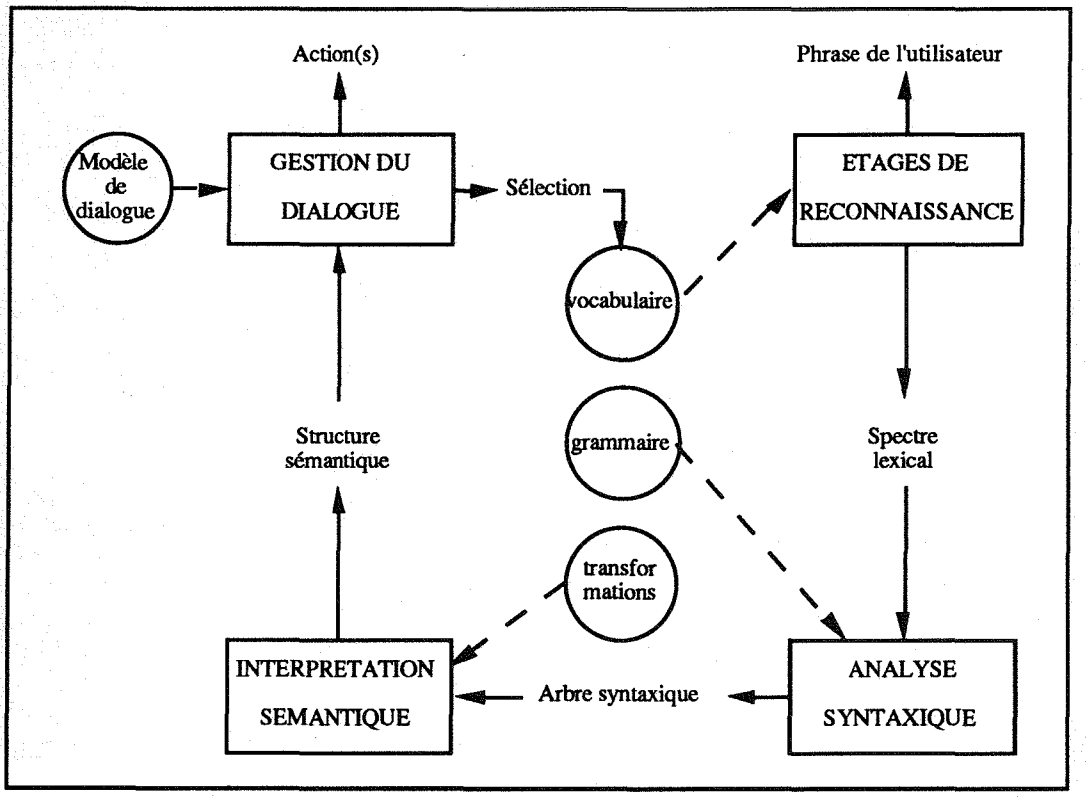


FIGURE 1.2 : ORGANISATION FONCTIONNELLE DE KEAL

1.4.5.2 les systèmes MYRTILLE

Le système MYRTILLE I [PIERREL 1975] est un système développé au CRIN [FIGURE 1.3]. Ayant été prévu pour fonctionner avec des langages artificiels, il est centré sur un analyseur syntaxique descendant qui guide le processus de reconnaissance. C'est un système qui se veut indépendant du langage de l'application choisie, la définition du lexique et de la syntaxe étant un paramètre du système. Il a été implémenté pour deux applications : langage de commande d'un standard téléphonique [PIERREL 1975] et pour un langage de commande d'un système informatique [LAFONT 1983]. Le système utilise une grammaire sémantique dont les terminaux sont des classes sémantiques et les non-terminaux des concepts utiles à la représentation des connaissances [BONNET 1980]. La procédure de dialogue mise en œuvre dans MYRTILLE I correspond alors à un parcours d'un arbre sémantique.

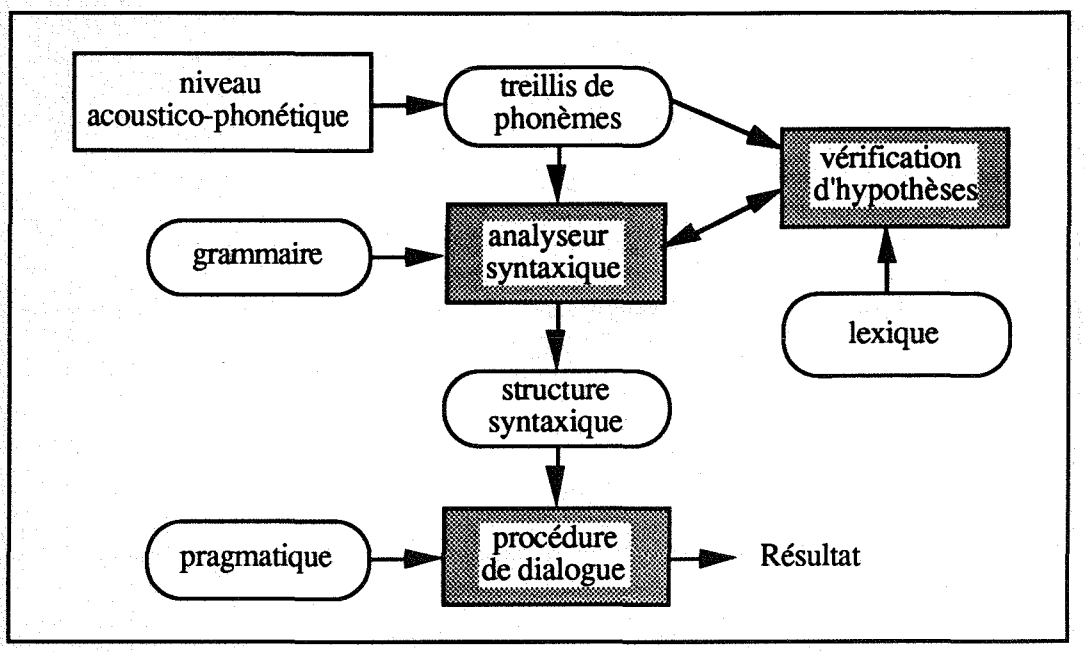


FIGURE 1.3 : SCHEMA GENERAL DU SYSTEME MYRTILLE I

Ce type de système est bien adapté à la communication à l'aide d'un langage artificiel. Un système plus ambitieux permettant de prendre en compte la compréhension de langages pseudo-naturels a suivi cette étude. Il s'agit de MYRTILLE II [PIERREL 1981]. Il a nécessité la mise en place d'un processus de compréhension et d'une procédure de dialogue beaucoup plus complexe que dans le cas précédent. Le modèle du dialogue associe une analyse sémantique fondée sur une grammaire par cas [FILLMORE 1968] et une procédure de dialogue. L'analyseur sémantique construit une interprétation de la phrase et détermine, dans le cas d'énoncé incomplet, les questions à poser pour obtenir des informations complémentaires. La procédure de dialogue a pour but de diriger le dialogue. On peut donc essayer d'utiliser un système plus simple pour traiter les réponses du locuteur [FIGURE 1.4] (dans ce cas de MYRTILLE I).

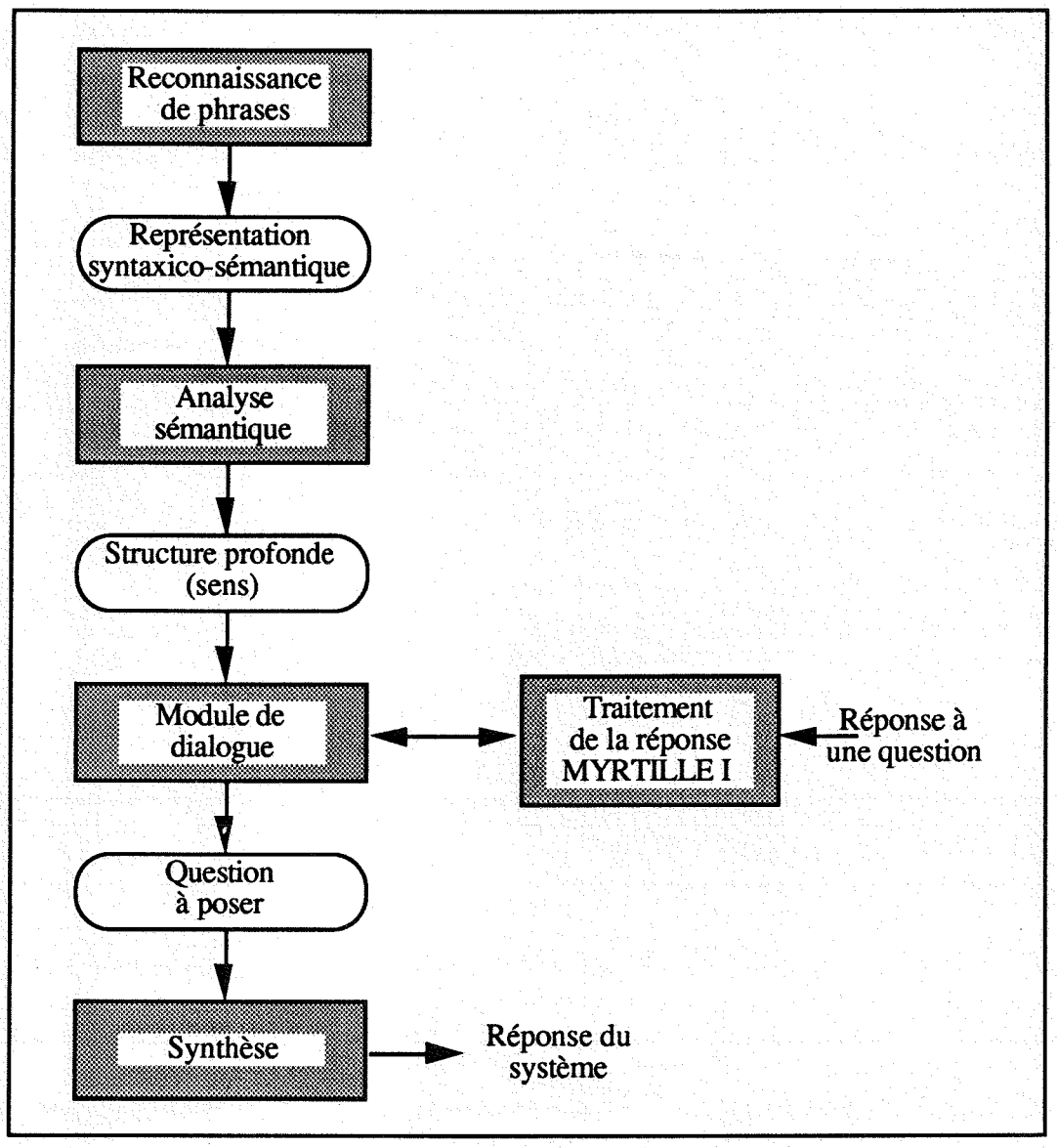


FIGURE 1.4 : INTERPRETATION D'UNE PHRASE ET DIALOGUE UTILISATEUR-SYSTEME DANS MYRTILLE II

1.4.5.3 le système ESOPE

Le projet ESOPE développé au LIMSI a pour objectif la réalisation d'un système de reconnaissance de la parole continue [MARIANI 1978] [MARIANI 1982].

Le système ESOPE dispose de quatre composantes :

- le niveau de décodage acoustico-phonétique délivre un treillis phonétique, Ce treillis est ensuite transmis au niveaux supérieurs afin de déterminer la phrase prononcée en tenant compte des contraintes du langage d'application,
- le niveau lexique comporte les représentations graphémiques et phonétiques des mots,

- le niveau syntaxe et sémantique est lié à l'utilisation d'une grammaire sémantique représentée par des ATN [WOODS 1970],
- le niveau pragmatique et dialogue a pour rôle de prédire le sous-langage (lexique et syntaxe) correspondant à la progression du dialogue, d'interpréter la phrase reconnue, de générer un message vocal. La structure du dialogue est représentée par un arbre syntaxique.

La stratégie utilisée pour obtenir la phrase prononcée à partir du treillis phonétique et avec l'aide des niveaux supérieurs, est de type prédiction-vérification avec analyse de la droite vers la gauche et conservation des meilleures solutions en parallèle sans retour arrière.

Un souci d'adaptation à l'univers de la tâche a conduit à la réalisation d'un système d'aide à la génération de langage d'application [MEMMI 1982] permettant de construire et de modifier simplement la syntaxe et le vocabulaire relatif à une application. Les grammaires construites sont de type ATN.

Une version pour une application de standard téléphonique a été étudiée.

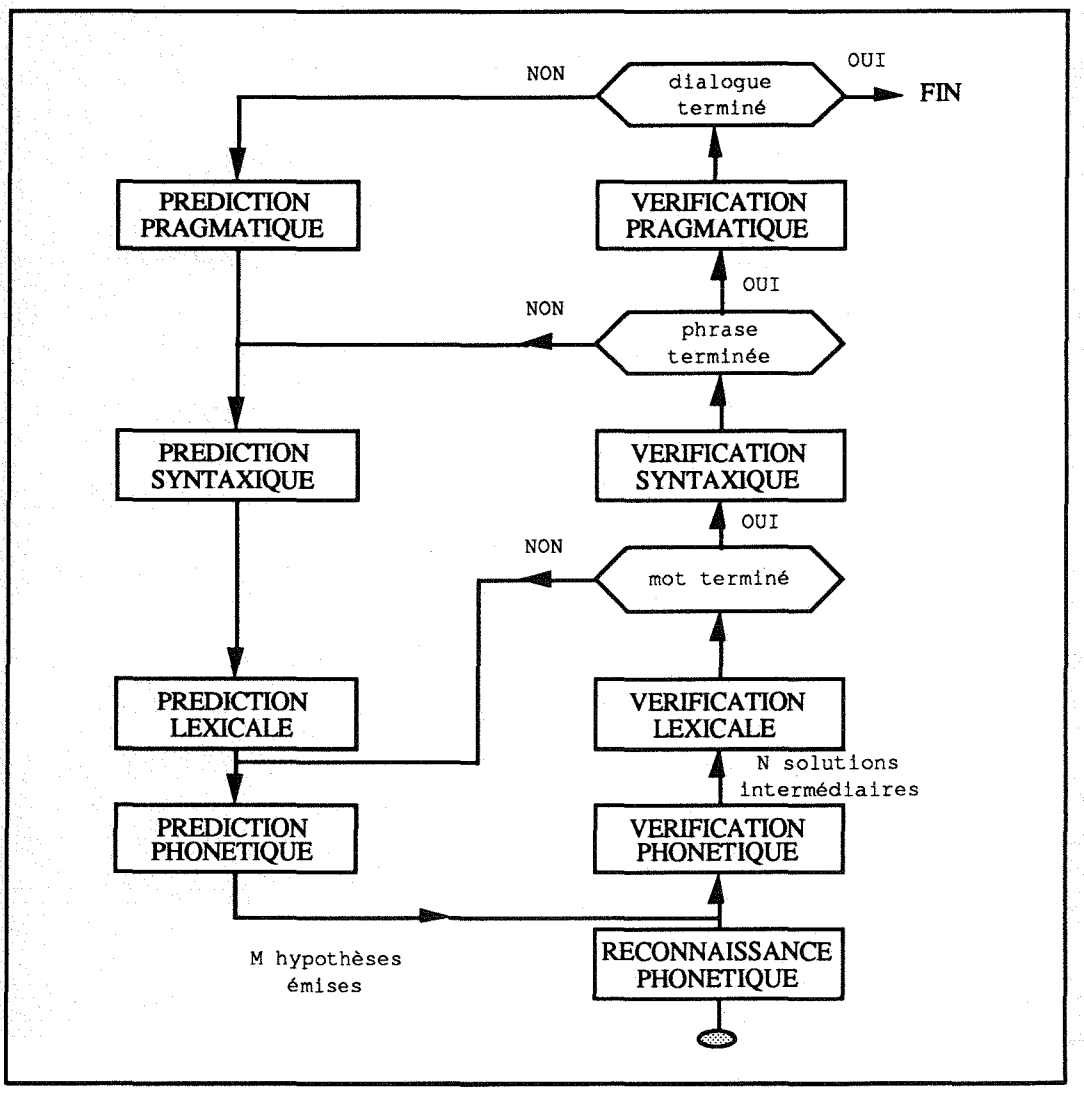


FIGURE 1.5 : STRATEGIE DU SYSTEME ESOPE

1.4.6 Les systèmes de dialogue du CRIN

1.4.6.1 le système DIAL

Les études menées au CRIN pour MYRTILLE I [PIERREL 1975] et MYRTILLE II [PIERREL 1981] ont conduit à la mise en œuvre d'un véritable système de dialogue oral homme-machine : le système DIAL. L'objectif de ce projet est de construire une interface orale homme-machine efficace et confortable pour l'utilisateur, dans le cadre d'applications informatiques destinées au grand public. L'application retenue pour tester le système est la consultation des informations administratives contenues dans les pages roses de l'annuaire.

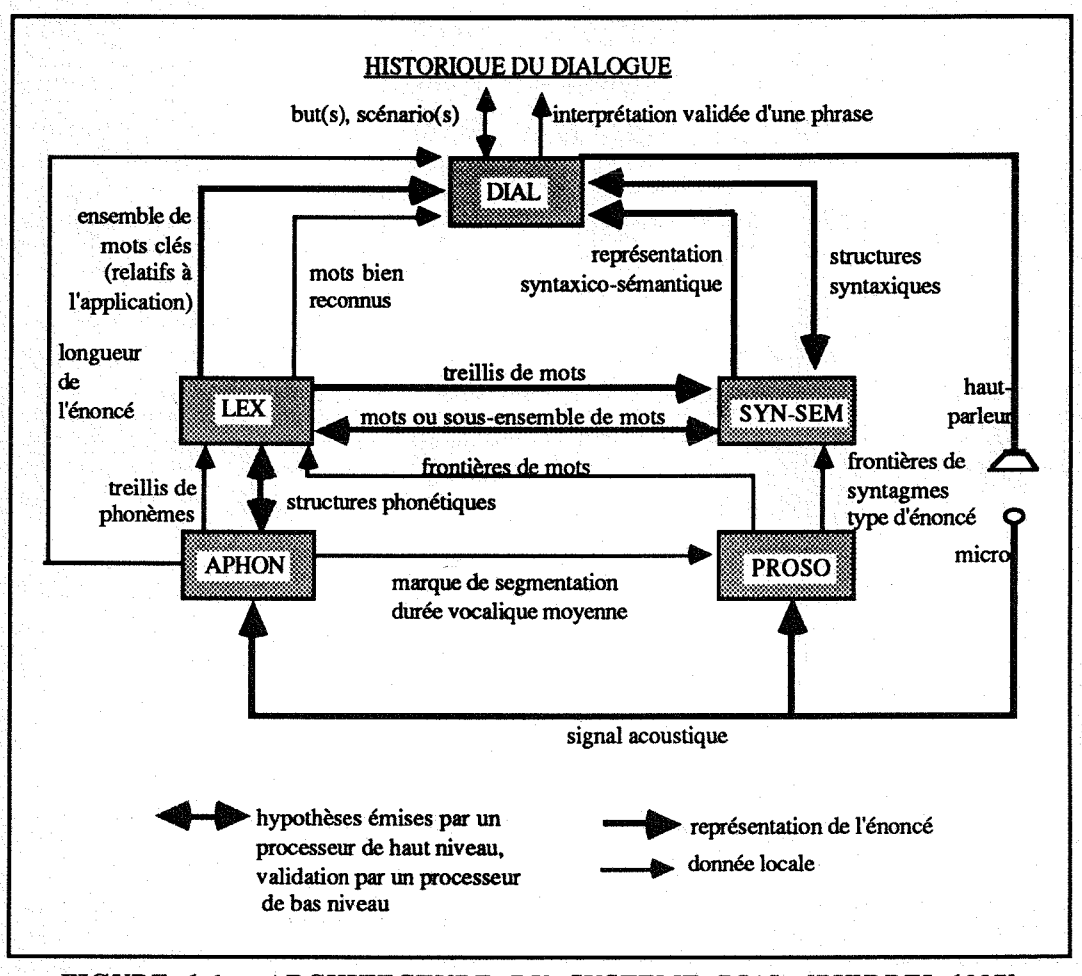


FIGURE 1.6 : ARCHITECTURE DU SYSTEME DIAL [PIERREL 1987]

L'architecture d'ensemble du système est de type multi-expert [CARBONNEL 1986], fondé sur différentes sources de connaissances. Les différents constituants du système, appelés processeurs, sont :

- le processeur de décodage acoustico-phonétique APHON, qui à partir du signal acoustique construit un treillis d'unités phonétiques [FOHR 1986],
- le processeur de traitements prosodiques PROSO, qui est chargé de détecter les marqueurs prosodiques indiquants les frontières des mots ou de

syntagmes et la nature de l'énoncé à traiter : affirmation, question, contestation [BONIN 1990],
- le processeur de traitement lexical LEX, qui assure la vérification des mots ou sélection des mots selon le mode de fonctionnement [MANGEOL 1988],
- les analyseurs syntaxico-sémantiques SYN-SEM, qui peuvent fonctionner de manière ascendante et descendante en utilisant un modèle syntaxique (réseau à nœuds procéduraux) et un modèle sémantique dérivé des grammaires de cas [DEVILLE 1989][MOUSEL 1990],
- une composante dialogue DIAL, qui effectue l'interprétation contextuelle, le raisonnement et la gestion du dialogue [ROUSSANALY 1988] [ROUSSANALY 1992]

Chaque processeur est chargé de gérer les connaissances statiques spécifiques. Il peut échanger des informations dynamiques avec les autres processeurs.

Le projet DIAL ne conduira pas à la réalisation d'un système opérationnel. Il se veut être un instrument d'étude sur le dialogue et les idées mises en œuvre dans ses différentes composantes pourront être réutilisées pour des systèmes moins ambitieux mais plus opérationnels, tels ceux présentés ci-dessous.

1.4.6.2 le système PARTNER

Le système PARTNER [MORIN 1987] a pour objectifs de concevoir et réaliser une interface homme-machine souple et conviviale disposant d'une entrée vocale et de définir une architecture permettant le développement d'application multi-media. Il se veut multilingue. L'architecture est de type modulaire, le système est composé de trois modules :
- un module de gestion de l'application. Une implémentation de PARTNER a été réalisée pour une application de manipulation d'objets graphiques à la fois en français et en anglais,
- un module de gestion des entrées-sorties : parole, clavier et souris. L'entrée parole est de type parole continue, avec un langage de type articiel à consonance naturelle. Elle est réalisée par le système APHODEX [FOHR 1986]. L'entrée clavier tolère des fautes de frappe. Les entrées ne sont pas multimodes,
- un module de gestion du dialogue qui permet de prendre en compte des énoncés elliptiques, anaphoriques ou incomplets, et qui autorise les contestations, corrections, répétitions, assistance...

Dans la version laboratoire développée au CRIN, il n'existe pas de connexion directe avec un système de reconnaissance fonctionnant en temps réel. Une version développée en collaboration avec STL[1] [MORIN 1992] est en cours de développement avec une connexion à un système de reconnaissance fonctionnant en temps réel, mais néanmoins, malgré une volonté d'indépendance vis-à-vis de l'application, il n'existe pas d'outil d'aide à l'implémentation d'une nouvelle application.

---

[1] STL Speech Technology Laboratory, Santa-Barbara, California USA

1.4.6.3 dialogue avec un robot

Les études sur le dialogue avec un robot [KLEIN 1990] s'intègrent dans le cadre d'un projet plus vaste, le projet ORASIS [MOHR 1988], qui consiste au développement d'un système de vision pour un robot mobile se déplaçant dans un univers d'intérieur.

L'idée de départ de ce travail était de fournir à ce robot une interface de communication permettant de lui demander de réaliser quelques actions simples sur l'univers dans lequel il évolue. Les contraintes de description de l'univers et d'identification des objets dans lequel le robot se déplace ont amené à utiliser la langue naturelle (exemple : le petit livre rouge près de la fenêtre). Le système développé utilise une grammaire de cas, où à chaque cas (agent, objet, destination...) est associé un ensemble de grammaires locales définissant les formes possibles que peut prendre le cas (exemple : un cas objet peut être représenté par un groupe nominal...). En plus de ces grammaires locales, il existe un ensemble de règles permettant de gérer les coordinations et qualification de groupe de manière à reconnaître des groupes complexes (exemple : le livre rouge et le cahier près de la fenêtre). L'instanciation des groupes consiste à rechercher dans l'univers l'objet désigné. Le système est capable d'émettre des diagnostics d'échec, de résoudre des ambiguïtés.

Il s'agit dans ce système d'une étude plus axée sur les Groupes Nominaux qui n'est pas véritablement intégrée dans un dialogue.

1.4.6.4 dialogue d'aide pour une standardiste mal-voyante

Le but de ce projet est d'étudier et de réaliser l'automatisation d'un poste de standardiste pour mal-voyants utilisant un dialogue homme-machine multi-media incluant une forte composante orale [ANGLADE 1991]. Le système de dialogue va assurer les liaisons et l'interface avec les quatres autres composants du système :

- l'opératrice qui est dans ce cas une mal-voyante,
- un annuaire "intelligent" dans le sens où il permettra d'accéder à un même numéro par plusieurs chemins, il sera constamment mis à jour et l'accès sera tolérant aux fautes,
- le système de reconnaissance (monolocuteur, mots isolés) et de synthèse de la parole,
- l'autocommutateur.

Le nombre de cas rencontrés par la standardiste étant restreint le dialogue est représenté sous forme d'automate et permet d'accéder aux données de la base d'informations par le nom, le nom de service ou par des mots-clés. Il peut poser des questions en cas d'homonymie ou proposer de choisir entre plusieurs personnes. Il autorise la correction et l'annulation.

Il s'agit d'un système dédié qui ne permet aucune paramétrisation par la tâche.

1.4.6.5 le dialogue multimodal dans MULTIWORKS

Le projet Esprit MULTIWORKS (Multimedia Integration Workstation) a pour objectif de définir une station de travail bureautique multi-media à faible coût [SMAILI 1991]. Dans ce projet le CRIN-CNRS & INRIA-Lorraine a lui-même

pour objectif la définition d'un gestionnaire de dialogue à forte composante orale pour la commande d'un éditeur multi-media. Cette étude a porté en particulier sur les points suivants :

- l'analyse de l'ensemble des fonctions utilisateur afin de modéliser au niveau du système de dialogue l'activité de l'utilisateur,
- la définition des types d'objets susceptibles d'être manipulés dans la tâche ainsi que les différentes relations (spatiales et conceptuelles) associées,
- enfin, à partir des données précédentes, la définitions de la langue associée à la tâche, afin d'implémenter rapidement le lexique et la grammaire [GAIFFE 1991].

A l'heure actuelle, ce système n'est pas encore entièrement opérationnel, la connexion à l'entrée parole n'étant pas encore réalisée.

1.4.6.6 le système DIAPASON

Les caractéristiques du système DIAPASON sont assez proches du système PARTNER dont il s'inspire au niveau architecture. On trouvera toutes les informations le concernant dans cette thèse.

1.4.7 D'autres systèmes de dialogue

1.4.7.1 le système CARAMEL

Le système CARAMEL (Compréhension Automatique de Récits, Apprentissage et Modélisation des Echanges Langagiers) développé au LIMSI, permet une collaboration étroite entre plusieurs systèmes experts [SABAH 1990].

Il comprend trois éléments fondamentaux :

- une mémoire structurée contenant les connaissances et les représentations nécessaires aux processus. La mémoire est divisée en trois zones :
  - une mémoire perceptuelle (ou mémoire à court terme) qui contient les phrases données en entrée,
  - une mémoire de travail qui contient les différentes interprétations construites à partie des données,
  - une mémoire à long terme qui contient des connaissances qui aident à la construction des interprétations,
- un ensemble de processus réalisant les différentes tâches nécessaires à la compréhension. La gestion de ces processus est fondée sur la notion de planification, en calculant dynamiquement quels sont les processus les mieux adaptés pour résoudre le problème en cours,
- un processus "maître", nommé SIROP (Superviseur Intelligent, Régisseur de l'Ordre des Processus), dont le rôle est de veiller au déclenchement et au fonctionnement cohérent des différents processus, et de représenter leur enchaînement.

Tout comme DIAL (§1.4.7.1), le système CARAMEL n'a pas pour objectif de donner directement un système opérationnel mais de servir de fondement pour la réalisation de systèmes opérationnels (cf STANDIA dans le paragraphe qui suit).

1.4.7.2 le système STANDIA

Le système STANDIA [VILNAT 1992], développé au LIMSI, a pour objectif l'élaboration d'un standard téléphonique intelligent capable de gérer deux formes de communication : l'écrit et l'oral. STANDIA utilise une architecture de type CARAMEL.

Les fonctionnalités du système permettent de rechercher un correspondant suivant son nom, sa fonction, d'obtenir des renseignements, de gérer des agendas individuels, de gérer une messagerie...

Le modèle du dialogue est issu des études de Roulet [ROULET 1985] sur la structure de dialogue homme-homme qui décompose le dialogue en fonction de trois primitives : l'échange, l'intervention et l'acte de langage.

Le système STANDIA est en cours d'étude, il n'est donc pas encore opérationnel.

1.4.7.3 le projet ESPRIT SUNDIAL

Le projet ESPRIT SUNDIAL [PECKHAM 1991] est un projet ambitieux qui a pour objectif de développer un système de dialogue oral. Il a pour originalité de prendre en compte les limitations imposées par le réseau téléphonique. Pour la partie française l'application visée est la réservation téléphonique de billets d'avion [CHARPENTIER 1992]. Les performances souhaitables (pour les aspects linguistiques et les phénomènes de dialogue) ont été identifiées grâce à une expérience de simulation selon la méthode du magicien d'Oz qui a permis de constituer un corpus [ANDRY 1990].

Du point de vue architecture, le système possède trois composantes : le dialogueur, l'analyseur linguistique et le système de reconnaissance.

Le noyau du système est le **dialogueur** dont la tâche est de gérer les deux canaux d'échange avec l'utilisateur (compréhension des énoncés de l'utilisateur et réponse vocale) et l'accès à la base de donnés. Il est constitué de cinq modules qui communiquent les uns avec les autres :

- le module de tâche a accès à la base de données de l'application et interroge le module de dialogue pour obtenir les paramètres d'accès à la base. Il peut fonctionner avec plusieurs applications différentes sans qu'il soit nécessaire de modifier les autres modules du dialogueur ;
- le module de dialogue produit des actes de dialogue [MAGADUR 1992] à chaque tour de parole à partir des messages envoyés par le module de la tâche et de la représentation des énoncés envoyés par l'interface linguistique. Il assure l'interprétation des actes de dialogue produits par l'utilisateur et calcule des prédictions sur les actes futurs de l'utilisateur [BILANGE 1991] ;
- le module de croyance maintient une représentation des objets et des relations du domaine,
- l'interface linguistique est un module d'interfaçage avec l'analyseur linguistique. Il reçoit la représentation de l'énoncé de l'utilisateur et la transmet au module de dialogue, il gère l'historique linguistique ;
- le planificateur de messages construit la représentation sémantique du prochain énoncé du système à partir de la liste des actes de dialogue à

produire envoyé par le module de dialogue. Le message généré est vocalisé par le système de synthèse à partir du texte de CNET.

L'**analyseur linguistique** reçoit en entrée le treillis lexical construit par le module de reconnaissance et produit l'arbre syntaxique et la représentation sémantique de la meilleure hypothèse destinés au dialogueur. Il s'appuie sur un formalisme grammatical dérivé des UCG (grammaires Catégorielles d'Unification) [ZEEVAT 1987] où les informations morphologiques, syntaxiques et sémantiques sont contenues dans les entrées lexicales sous forme d'attributs valués. La grammaire est constituée de quelques règles de combinaison fondées sur l'unification. Il reçoit du dialogueur des prédictions sur les prochains énoncés qu'il transmet au niveau reconnaissance.

La **reconnaissance vocale** est assurée par le système de reconnaissance de la parole continue développé par LOGICA [MICCA 1990]. L'algorithme de reconnaissance est basé sur une modélisation HMM des phones du français, indépendante du contexte phonétique. Les modèles ont été obtenus par apprentissage sur une base de données de 99 locuteurs enregistrés sur le réseau téléphonique commuté, chacun ayant prononcé 150 phrases (phrases phonétiquement équilibrées et phrases liées à l'application aérienne) et 180 mots isolés (dont les chiffres, les jours, les mois, les lettres).

SUNDIAL est un projet en cours de développement. Les performances du système de reconnaissance sont encore insuffisantes pour en faire un système opérationnel.

1.4.7.4 le dialogue sous LOIR

L'objectif poursuivi à l'IRIT (Institut de Recherches en Informatique de Toulouse) est de réaliser une interface multimodale. Le cadre de l'expérimentation des études est un dialogue finalisé par la tâche entre un enseignant expert d'un domaine, simulé par la machine, et un utilisateur novice.

Les travaux réalisés reposent sur l'interpréteur LOIR (Lisp Objet à Inférences Réflexes), structure d'accueil pour la conception de dialogues multimodaux. Il utilise un formalisme unifié des connaissances (lexicales, syntaxiques, sémantiques et pragmatiques) constitué de graphes conceptuels de Sowa [SOWA 1984] Ce formalisme unique devrait permettre de gérer la multimodalité.

Un modèle de l'application décrit les concepts utilisés et leur comportement à l'aide de classes, de schémas d'actions et de règles [ARRONATEGUI 1992]. Des graphes de compréhension sont construits à partie de graphes conceptuels [SOWA 1984] des énoncés et à l'aide de règles de formation. L'interprétation des graphes se fait en tenant compte des connaissances statiques et dynamiques de la tâche. Les techniques utilisées sont le filtrage (intégration dans le monde), la reconnaissance de plans (anticipation) et la jointure dirigée (intégration dans l'historique) [ARRONATEGUI 1991]

Un superviseur spécifie les stratégies du dialogue en fournissant une description générale des diverses situations de dialogue spécifiques à une application : demande d'information, demande de répétition, confirmation, contestation.

Le système a été validé sur une exemple simpliste de la cueillette de fruits où l'agent commande un personnage graphique dans un monde virtuel et où

l'observateur gère l'environement modifié par les actions de l'agent, contrôle les actes de l'agent (réponse à des questions, demande de précisions, anticipations d'actions). Il est à l'étude pour un système d'aide sur l'enseignement des procédures juridiques liées aux "remises de majoration de retard" en lien avec l'URSSAF (Union de Recouvrement de la Sécurité Sociale et des Allocations Familiales).

Ce projet est un essai d'intégration sans véritable réflexion et apport sur les aspects spécifiques des dialogues naturels oraux.

1.4.7.5 dialogue multimodal avec un robot manipulateur

L'IRIT (Institut de Recherche en Informatique de Toulouse), le LAAS (Laboratoire d'Automatique et d'Analyse des Systèmes) et l'AIP (Atelier Interuniversitaire de Productique) développent en commun un système de gestion de dialogue pour un robot doté de la vission artificielle et utilisant plusieurs modes de communication [CONDOM 1988][CONDOM 1992] Le type d'application visé est la saisie et le déplacement d'objets dans un univers réduit où ces objets peuvent être facilement désignés par leur étiquette. L'entrée des informations dans le système se fait, soit oralement à l'aide du système de reconnaissance de la parole RME 186 développé par VECSYS à partir des recherches du LIMSI, soit au clavier (notamment pour les entrées numériques ou les opérations exigeant une grande fiabilité). La sortie des informations se fait par synthèse vocale à l'aide de la carte TELEVOX développée par ELAN INFORMATIQUE sous licence CNET, associée à un affichage textuel plus complet sur la tâche.

Le système de dialogue (SD) est composé de plusieurs modules :

- le module de recueil, représentation et diffusion des événements provenant ou à destination des medias (reconnaissance vocale, vision, synthèse vocale, affichage sur écran, robot),
- le module de dialogue qui s'occupe de la gestion du canal de communication (gestion des messages incompréhensibles ou incomplet, attente, relance), et du dialogue (dialogue fortement orienté par la tâche),
- le module de raisonnement permet de définir un objectif propre au SD ou d'effectuer une interprétation contextuelle de la commande (contrôles de validité et détermination de l'objectif de l'utilisateur),
- le module de la tâche intervient lorsque la commande de l'opérateur a été validée. Il va gérer la tâche correspondante. Il doit pour cela élaborer un plan d'actions (pour les tâches complexes), traiter les cas d'échec et permettre à l'opérateur de contrôler la tâche.

Le modèle du dialogue est représenté par un ensemble de réseaux de dialogue du type ATN [WOODS 1990]. Un réseau de dialogue peut correspondre à un échange simple (une question suivie d'une réponse) ou à un échange complexe (commande de manipulation).

Le modèle de la tâche est lui aussi décrit par des ATN. Un réseau peut décrire la saisie ou le déplacement d'un objet... Les actions peuvent correspondre à l'appel à un sous-réseau de la tâche, l'appel à un réseau de dialogue, commande du robot...

Le modèle du langage est décrit par un ensemble de grammaires régulières et analysé par un automate d'états finis.

En conclusion le système utilise un niveau reconnaissance global et ne dispose pas d'outils d'aide au développement.

1.4.7.5 le système ICPplan

Le système ICPplan [BOURGET 1992] appartient aux systèmes récents où les aspects multimodaux sont pris en compte dans le dialogue homme-machine. Il s'agit d'un logiciel d'aide à la conception de plans architecturaux. L'interface multimodale propose, pour l'interaction homme-machine, le geste, la parole et le langage naturel écrit en entrée, la vision et la parole en sortie.

La tâche est décrite en termes hiérarchiques de plans, sous-plans, scenarii, scripts et actions. L'analyse des commandes est faite au niveau morphologique ce qui permet la fusion des informations multimodales (association du déictique avec l'objet selectionné à la souris) et active des scripts. Un dialogueur recherche ensuite dans son historique les informations utiles et manquantes au script. Cette stratégie ascendante est combinée à une stratégie descendante lorsque les plans de l'utilisateur ont été inférés au préalable. Le script choisi doit s'inscrire dans un scénario actif et faire progresser ainsi la réalisation d'un plan. S'il y a contradiction avec les intentions de l'utilisateur le système peut remettre en cause les intentions de l'utilisateur, proposer au dialogueur une autre interprétation de commande, engager un dialogue avec l'utilisateur.

Ce système est en cours de développement, il est dédié à une application.

1.4.8 Conclusion sur les systèmes de dialogue

Après avoir présenté ces quelques systèmes de dialogue nous pouvons dire qu'ils se répartissent en deux catégories.

La première catégorie a pour but d'étudier la modélisation du dialogue homme-machine et de proposer une implémentation informatique. Ils abordent donc des aspects très théoriques et on ne peut envisager pour les années qui viennent de réelles implémentations complètes de ces systèmes. C'est le cas de DIAL et de CARAMEL par exemple.

La seconde catégorie de système a pour but de réaliser un véritable système opérationnel. Malheureusement les performances des systèmes de reconnaissance de la parole actuels ne permettent pas toujours de rendre ces systèmes complétement opérationnels. On peut aussi distinguer les systèmes qui dépendent fortement de l'application, où lorsque l'on change l'application il faut tout réécrire, et ceux qui sont paramétrables par cette application. Enfin, parmi les systèmes paramétrables, il faut tenir compte des facilités données au concepteur de tels systèmes pour réduire le travail à réaliser pour une nouvelle implémentation.

Notre objectif avec DIAPASON n'est pas aussi ambitieux que DIAL, SUNDIAL ou CARAMEL, mais il se situe à un niveau comparable à STANDIA ou PARTNER version STL. Notre étude a pour but de réaliser un système qui soit véritablement opérationnel. Il a pour originalité d'utiliser un niveau reconnaissance de type analytique et multilocuteur et de disposer d'outils de développement qui en font une véritable plate-forme de développement pour une catégorie d'application (la commande de machine) et non pas un système dédié.

Chapitre 2

L'ENVIRONNEMENT DIAPASON

2.1 LES OBJECTIFS GENERAUX

Notre objectif principal au cours de cette thèse, a été la conception d'un système de dialogue oral homme-machine.

Nous avons voulu nous placer dans le cadre d'un **véritable dialogue applicatif** pour réaliser un prototype **opérationnel** de système qui puisse servir de démonstrateur pré-industriel. Le système visé ne se contenterait pas de faire de la reconnaissance de phrases, mais devait prendre en compte les phénomènes typiques du dialogue que sont la gestion d'un historique et les interactions avec la tâche.

Le système de dialogue devait entrer en interaction forte avec un système de **reconnaissance analytique** de la parole.

Le système devait être **paramètrable par l'application**, donc il devait être indépendant du vocabulaire-syntaxe, ce qui a conduit à définir un modèle de dialogue et au développement d'outils de mise en place des connaissances spécifiques à l'application et nécessaires au système de dialogue.

Ces objectifs nous ont amené à la réalisation d'un environnement de définition de système de dialogue.

2.2 LES COMPOSANTES DE L'ENVIRONNEMENT DIAPASON

L'environnement DIAPASON est constitué de quatre composantes [FIGURE 2.1] :

(a) le système de reconnaissance de la parole,

(b) le système de dialogue,

(c) l'application à diriger,

(d) les outils de mise en place des connaissances spécifiques à l'application.

Notre travail a consisté plus particulièrement, outre la définition de l'architecture générale du système, à définir et implanter les composantes (b) et (d) et, pour la partie simulation, la composante (c). Par ailleurs nous avons eu à définir les protocoles de communications entre ces quatre composantes.

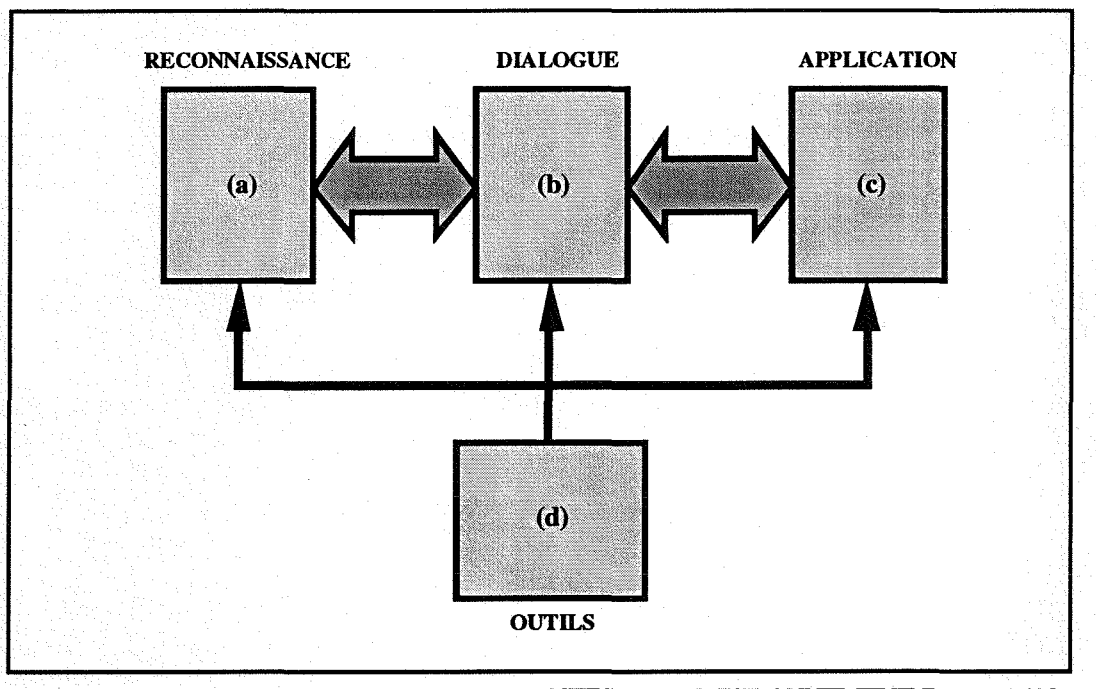


FIGURE 2.1 : LES DIFFERENTES COMPOSANTES DE L'ENVIRONNEMENT DIAPASON

2.2.1 Le système de reconnaissance

Le système de reconnaissance de la parole a été développé par l'équipe de Pierre ALINAT de THOMSON SINTRA DASM. Le système est analytique, il utilise des connaissances acoustiques, phonétiques et linguistiques pour construire une phrase à partir du signal de parole [ALINAT 1975] [ALINAT 1987] [GALLAIS 1991].

2.2.1.1 description de la reconnaissance phonétique

Le système de reconnaissance analytique est composé d'une chaîne de traitement ascendante du signal acoustique vers les phonèmes :

- analyseur,
- estimation des paramètres acoustiques,
- localisation des phones (phone = phonème reconnu),
- estimation des paramètres des phones,

et d'une chaîne de traitement descendante de la phrase vers les phonèmes assez comparable aux système MYRTILLE I [PIERREL 1978] :

- vocabulaire-syntaxe,
- analyse syntaxique,
- comparaison des phones (phone = phonème reconnu) avec la chaîne de phonèmes du vocabulaire.

De plus la stratégie générale est dans l'ensemble, gauche-droite avec de légers retours arrière possibles à l'intérieur des syllabes.

L'analyseur est constitué par un banc de quarante-huit filtres passe-bande linéaires suivis chacun d'une détection intégration. Les fonctions de transfert de ces filtres ont été choisies d'après ce qui est connu de la partie mécanique de la cochlée

humaine. En ce sens on peut dire qu'il est de type cochlée artificielle et fait partie avec le système développé par J. CAELEN [CAELEN 1979][CAELEN 1985] des rares systèmes français utilisant une modélisation de l'oreille en reconnaissance automatique du français oral.

Un certain nombre de paramètres acoustiques sont estimés toutes les huit microsecondes. Il s'agit de paramètres qui sont utilisés par la suite pour vérifier l'existence des traits (voisement, friction, position des formants...).

La détection de la présence de parole est basée sur l'occurrence de signaux voisés.

La localisation des phones ne suit pas la segmentation traditionnelle telle que l'on peut l'utiliser au CRIN [HATON 1990]. Il ne s'agit pas de découper le signal d'entrée en signaux adjacents mais plutôt de détecter la présence de telle ou telle catégorie de phones (voyelles, semi-voyelles, fricatives, plosives et plosives nasales, phonème R). Elles sont recherchées en parallèle en utilisant les traits propres à chaque catégorie.

Pour chaque phone détecté et selon sa catégorie un certain nombre de paramètres sont estimés. Les paramètres sont relatifs aux différents traits impliqués dans la description de chaque catégorie. Certains concernent le phonème en tant que faisant partie de sa catégorie (par exemple pour les voyelles : amplitude, durée, force de voisement...), les autres concernent la classification fine du phone, c'est-à-dire à l'intérieur de sa catégorie (par exemple pour les voyelles : description de la position des formants et degré de nasalisation).

2.2.1.2 résultats de la détection des phonèmes

On obtient ainsi une chaîne de phones, chacun étant décrit par la catégorie à laquelle il appartient et les paramètres correspondants. Une petite analyse a montré qu'au niveau résultat les paramètres sont extraits correctement sauf la nasalisation des voyelles et la place d'articulation des plosives [GALLAIS 1990].

2.2.1.3 principe de la reconnaissance de mots et de phonèmes

L'analyseur syntaxique émet des hypothèses de mots à reconnaître, la chaîne de phonèmes du mot est comparée à la chaîne des phones reconnus. La meilleure correspondance entre les deux chaînes est obtenue par programmation dynamique. Les règles de comparaison font intervenir des règles de classification, des règles de coarticulation et jonction, les forces des phones reconnus et accentuations des phonèmes.

La syntaxe de base est une syntaxe de mots enchaînés.

2.2.1.4 résultats de la reconnaissance de mots et de phonèmes

Avec un vocabulaire de 100 mots et des facteurs de branchement de l'ordre de 15 à 30, les résultats suivants ont été obtenus en multi-locuteur (7 locuteurs, 3 locutrices) [GALLAIS 1989] :

- phrases reconnues correctement : 85%,
- mots reconnus correctement : 95%,
- nombre de phrases nécessaires pour qu'une commande soit correctement exécutés : 116%.

Il n'y a pas de différences marquées de taux d'erreur entre les locuteurs et ces résultats ont été obtenus en ambiance de laboratoire, pour des vitesses de prononciation normale laissées au libre choix de chaque locuteur.

2.2.2 L'application à diriger

Une des composantes de l'environnement DIAPASON est l'application à diriger. Le système de dialogue s'adresse à une catégorie d'application : la commande de machine, où toutes les phrases peuvent être entièrement décrites par leur syntaxe à l'aide d'un langage artificiel restreint. Les phrases peuvent être définies à l'aide d'une grammaire à contexte libre [CHOMSKY 1965] [CHOMSKY 1968].

Nous avons testé le système de dialogue avec deux applications :

- l'application SONAR qui consistait à commander une console sonar simulée sur PC et inspirée d'un matériel développé par THOMSON SINTRA,
- l'application DIVA qui consistait à commander un système d'analyse de signaux sous-marins, application réelle développée par le CERDSM au BRUSC près de TOULON.

Ces deux applications seront présentées plus en détail dans le chapitre 5.

Ces deux expériences nous ont permis de définir nos besoins pour connecter DIAPASON à une application et à définir des conditions à vérifier par l'application pour que cela soit possible.

La première condition reprend ce que nous disions plus haut, c'est-à-dire que l'application doit être de type commande de machine. On pourra alors définir des ordres composés d'une commande et de paramètres. Les applications qui sont dirigées par des menus hierarchiques entrent dans ce cadre.

Ensuite il faut qu'il y ait possibilité d'interaction entre l'application et le système de dialogue. L'application doit être conçue de façon ouverte, c'est-à-dire que son état doit être à tout moment accessible et modifiable par le système de dialogue. Cela nécessite donc que l'état de l'application soit exprimé de manière explicite et non pas dilué de façon implicite dans un grand nombre d'objets. En particulier le système de dialogue doit pouvoir prendre connaissance de ce qui est possible ou non à l'instant t.

Enfin il faut pouvoir effectuer la confirmation des ordres vocaux en place : il faut simuler l'effet qu'aura l'ordre vocal sur l'application sans pour autant modifier l'état interne des objets. Il peut s'agir de changement de couleur ou d'inversion video dans la zone de l'écran que l'utilisateur est sensé regarder pendant qu'il donne l'ordre vocal.

Notons que d'un point de vue informatique la durée de communication entre le système de dialogue et l'application doit être le plus court, il vaut donc mieux qu'il y ait un exécutable unique pour le système de dialogue et l'application.

2.2.3 Le système de dialogue

Le système de dialogue est au cœur de l'environnement DIAPASON. Il communique avec le système de reconnaissance et l'application.

Le système de dialogue s'appuie sur un modèle de dialogue qui permet la correction, l'annulation, la confirmation et la répétition. Il autorise des tournures elliptiques. Ce modèle est présenté en détail avec de nombreux exemples dans le chapitre 3.

Son architecture est modulaire, il se compose d'un module chargé de gérer les échanges avec le système de reconnaissance, d'un module chargé de gérer le dialogue, d'un module chargé de gérer la tâche et la communication avec l'application. Chaque composante sera développée dans le chapitre 4.

Le système de dialogue a été développé au CRIN-CNRS & INRIA-Lorraine et correspond à une part importante de ce travail de thèse.

2.2.4 Les outils de développement du dialogue

Les outils de développement du dialogue, originalité importante de DIAPASON qui en ce sens correspond plus à une plate-forme de développement qu'à un simple système de dialogue, vont permettre de mettre en place les connaissances spécifiques à l'application et qui sont nécessaires aux niveaux lexical, syntaxique et sémantique. Il s'agit principalement :

- d'une plate-forme d'accueil qui regroupe la maquette de reconnaissance, le système de dialogue et l'application,
- d'un éditeur qui permet la saisie des structures syntaxico-sémantiques,
- d'un compilateur qui transforme les structures en données assimilables par le système de dialogue et qui produit les structures syntaxiques et lexicales,
- de formateurs qui mettent les structures lexicales et syntaxiques au format des analyseurs syntaxiques.

Nous avons développé les outils qui s'adressent au système de dialogue. Après en avoir fait les spécifications précises [SOUVAY 1990] et ANNEXE E, les implémentations ont été réalisées et testées [LONG 1991][DRUART 1992] [FROT 1992]. Je tiens à remercier Patrick LONG et Pascal DRUART, étudiants de DEA, et Jean-Philippe FROT, étudiant d'ESIAL[1], pour l'aide qu'ils m'ont apporté dans la phase finale de la programmation de ces outils.

Les outils qui s'adressent à la maquette de reconnaissance de la parole sont en cours de spécification [SOUVAY 1992].

2.2.5 Rétro-action vers l'utilisateur

Les applications que nous avons développées se situent dans le domaine de l'acoustique sous-marine. Les opérateurs travaillant sur les consoles sont à l'écoute du signal. En conséquence les demandes ou les retours oraux nous étaient interdits. Deux mécanismes ont alors été mis en œuvre. Une première technique a consisté à afficher les phrases reconnues ou la demande du système dans une zone réservée de l'écran. Cette solution est peu satisfaisante car elle oblige l'opérateur à déplacer son regard vers cette zone. Une seconde technique a consisté à effectuer une rétro-action en place c'est-à-dire par exemple à simuler le résultat de l'exécution de l'ordre à l'endroit où se déroule l'action.

---

[1] ESIAL : Ecole Supérieure d'Informatique et Automatique de Lorraine

2.3 LE CADRE DES TRAVAUX REALISES

Les travaux que nous avons réalisés au cours de notre thèse se situent dans le cadre de plusieurs contrats pour le compte de la DRET, en collaboration avec THOMSON SINTRA DASM.

Les motivations de la DRET peuvent s'expliquer ainsi. Les industriels proposent aux militaires des matériels qui s'avèrent de plus en plus complexes à faire fonctionner. Ces derniers cherchent des moyens d'améliorer les conditions de travail des opérateurs qui, sur certains systèmes, sont rapidement saturés et d'augmenter leur productivité. Une solution semble être l'utilisation de la parole. Des études ont été lancées sur des systèmes de type globaux (pour des applications en avionique en particulier [TARNAUD 1984]) et sur des systèmes de type analytique (démonstrations SONAR [GALLAIS 1989] et DIVA [ALINAT 1992]).

Les motivations de la THOMSON sont différentes selon les contrats. Il s'agissait dans un premier contrat et pour la première démonstration, de tester la fiabilité d'un système de reconnaissance original et très performant en conditions réelles. Des problèmes d'ordre technique ne permettaient pas de se connecter sur le matériel réel d'où le choix d'une simulation aussi réaliste que possible. Le but de la deuxième démonstration a donc été de se connecter à une application réelle. Une troisième démonstration devrait avoir lieu, toujours sur une application réelle, mais avec une maquette de reconnaissance de la parole modernisée. La précédente date du début des années 80 et repose donc sur une technologie ancienne. Elle ne permet ni d'améliorer le temps de réponse du système, ni de prendre en compte dans les algorithmes l'expérience acquise au cours des études qui se sont succédées pour cause d'insuffisance de capacité mémoire. La version modernisée disposera de microprocesseurs plus rapides, de mémoires plus vastes et sera plus facilement programmable (utilisation du langage C à la place de l'assembleur).

Les motivations du CRIN-CNRS & INRIA-Lorraine sont de valider les recherches théoriques dans le domaine du dialogue oral homme-machine que le laboratoire effectue depuis quelques années et de s'ouvrir vers le monde industriel.

Il s'agit pour les deux partenaires de montrer ce qu'il est possible de réaliser dans l'état actuel des connaissances en communication orale homme-machine.

Ces études ont permis de réaliser un prototype de démonstration d'utilisation de la parole dans un contexte applicatif. Il est régulièrement présenté aux cadres supérieurs de THOMSON ou des armées pour que les décideurs soient amenés dès la phase de conception d'un nouveau système à y introduire une composante orale.

2.4 IMPLEMENTATION DE L'ENVIRONNEMENT DIAPASON

La première version du système de dialogue a été développé sur PC en TURBO C. Aujourd'hui cette version n'est plus maintenue.

L'environnement DIAPASON a été développé sur station de travail de type SUN sous UNIX et dans un environnement X Window System. Deux implémentations X ont été retenues : une première utilisant la boite à outils Athena Widgets, la seconde l'environnement OSF-MOTIF.

Chapitre 3

LE DIALOGUE DANS DIAPASON

3.1 INTRODUCTION

Le dialogue dans DIAPASON est constitué d'une suite d'échanges entre l'utilisateur et le système. Un dialogue élémentaire se déroule en trois phases. L'opérateur donne un ordre initial, une phase de mise au point suit pour faire coïncider l'interprétation du système à la demande de l'usager, puis enfin l'ordre est exécuté. DIAPASON est alors prêt à recevoir de nouvelles directives. Nous présenterons dans ce chapitre les différents éléments qui composent un ordre, nous étudierons ensuite les possibilités offertes au cours de la phase de mise au point et enfin les conditions qui régissent l'exécution de l'ordre donné. Nous présenterons finalement les grammaires qui sont sous-jacentes et qui dirigent le dialogue. Les exemples que nous présenterons seront tirés soit de l'application SONAR, soit de l'application DIVA. Nous présenterons ces applications plus en détail dans les chapitres qui suivront.

3.2 LA NOTION D'ORDRE

Il existe deux catégories de phrases dans DIAPASON, les phrases de gestion du dialogue et les phrases de commande de l'application. Nous les appellerons les ordres.

Les phrases de gestion du dialogue sont indépendantes de l'application et sont constituées d'un seul mot pour le type d'application considéré. Elles ont une fonction spécifique que nous présenterons dans ce chapitre. Ce sont les phrases :

ok	*confirmation de l'ordre,*
annuler	*annulation de l'interprétation,*
négatif	*annulation de la reconnaissance,*
encore	*répétition de l'ordre.*

Les phrases de commande permettent d'effectuer des actions sur des objets de l'application ou de donner des valeurs aux composantes de ces objets. Les ordres sont décomposés en éléments. Ils portent un nom issu de leur élément principal appelé commande, les autres éléments sont appelés paramètres. Dans les exemples qui suivent, les parenthèses indiqueront les différents éléments et le texte en caractères gras l'élément principal.

Exemples d'ordres de l'application SONAR :
(**afficher**) (bruiteur ALPHA 1)
(**remplacer**) (mémoire VPR) (par VPLB)

Exemples d'ordres de l'application DIVA :
(**afficher**) (contexte)
(image 2) (**gamme**) (40)

3.3 LA PHASE DE MISE AU POINT

La phase de mise au point permet de mettre en correspondance l'ordre que le système a compris et interprété avec ce que l'opérateur a voulu exprimer. En effet d'une part les taux de reconnaissance ne sont jamais de 100%, d'autre part une erreur d'interprétation de la machine est toujours possible. De plus l'opérateur peut lui-même commettre certaines fautes. Cette phase doit être la plus courte possible afin de ne pas alourdir le dialogue. L'utilisateur ne doit pas être gêné par des échanges trop longs pour entrer un ordre. Une durée trop importante de cette phase peut provoquer le mécontentement de l'usager, le fatiguer par des répétitions successives, le distraire s'il a l'impression de perdre du temps, etc. Une autre raison pour minimiser la durée de cette phase est que plus le nombre d'échanges augmente plus la probabilité d'une mauvaise reconnaissance ou d'une erreur d'interprétation s'accroît elle aussi. Enfin la parole doit rester compétitive par rapport aux autres modes en terme de charge de travail et de rapidité.

3.3.1 L'ordre initial

La phase de mise au point suppose qu'un ordre initial ait été donné au système. DIAPASON analyse cette demande et envoie en réponse un message. Dans le meilleur des cas, il s'agit de la bonne interprétation de la phrase donnée en entrée. Le locuteur est satisfait, le système passe alors à la phase d'exécution de l'ordre.

LEGENDE DES EXEMPLES DE DIALOGUES	O	:	phrase prononcée par l'**O**pérateur
	D	:	interprétation de **D**IAPASON
			# commentaires sur le fonctionnement de DIAPASON #

EXEMPLE 3.1	O	:	Graphique annexe LOFAR 1 gamme CD
	D	:	Graphique annexe LOFAR 1 gamme CD
			# exécution de graphique annexe LOFAR 1 gamme CD #

3.3.2 La correction de l'ordre

Le système peut malheureusement proposer un ordre incorrect (mauvaise reconnaissance ou mauvaise interprétation), il offre alors la possibilité de corriger un des éléments : commande ou paramètres. La correction se fait en utilisant le mot "négatif" suivi de la commande ou du paramètre correct .

EXEMPLE 3.2

O : Faire DEMON sur image 2
D : Faire LOUPE sur image 2
O : **Négatif DEMON**
D : Faire DEMON sur image 2

O : Afficher raies
D : Effacer raies
O : **Négatif afficher**
D : Afficher raies

Une limitation existe au niveau de la porté de la correction. On ne peut modifier un paramètre que globalement et non pas seulement un de ses composants. Par exemple dans le cas d'un bruiteur, on ne pourra pas changer la lettre ou le chiffre mais seulement l'ensemble lettre+chiffre. Cette restriction est à l'origine très fortement liée à la maquette de reconnaissance et destinée à réduire le facteur de branchement lors de l'analyse des phrases correspondant à une ellipse ou une à correction.

EXEMPLE 3.3

O : Graphique annexe fréquence ALPHA 1
D : Graphique annexe fréquence SIERRA 2
O : **Négatif ALPHA**
D : Je ne vous ai pas compris !
O : **Négatif ALPHA 1**
D : Graphique annexe fréquence ALPHA 1

3.3.3 Les messages d'erreur

Le message n'est pas toujours l'interprétation de l'ordre. Le système peut envoyer un message d'erreur indiquant la nature du problème rencontré. Il faut que le système évite au maximum ce genre de situation qui alourdit le dialogue et agace l'opérateur s'il s'agit d'une erreur d'interprétation. Nous étudierons en détail ce type d'événement lorsque nous traiterons de la cohérence des ordres.

EXEMPLE 3.4

O : Cap tête
D : **Le cap est déjà fixé à tête**

O : Afficher raies
D : **Les raies sont déja affichées**

3.3.4 Le complément d'information

Un troisième type de message que DIAPASON peut envoyer est une demande de complément d'information. Le système ne possède pas tous les éléments nécessaires à l'exécution d'un ordre, il pose alors une question à l'opérateur.

EXEMPLE 3.5

O : Afficher
D : **Afficher quoi ?**
O : Bosses extraites
D : **Afficher bosses extraites de quelle image ?**
O : Image 2
D : Afficher bosses extraites image 2

O : Libérer
D : **Libérer quelle mémoire ?**
O : VPR
D : Libérer mémoire VPR

3.3.5 L'annulation de l'ordre

DIAPASON offre la possibilité d'invalider l'ordre reconnu. Deux types d'annulation sont possibles.

L'annulation de la reconnaissance utilise le mot "négatif" et indique au système que le niveau reconnaissance a fourni une hypothèse erronée. Après cette annulation, DIAPASON fonctionnera comme s'il n'avait jamais reconnu la phrase incorrecte.

EXEMPLE 3.6

O : Effacer bruiteur pointé
D : Raz bruiteurs
O : **Négatif**
D : Que dois-je faire ?
 # la phrase reconnue est enlevée de la mémoire #
Raz
Raz bruiteurs ou raies ?

L'annulation de l'interprétation utilise le mot "annuler" et indique au système que l'ordre ne doit pas être pris en compte, l'utilisateur a changé d'avis. Il reste toutefois dans la mémoire du sytème de dialogue, DIAPASON pourra si besoin, utiliser des éléments de l'ordre.

EXEMPLE 3.7

O : Raz
D : Raz bruiteurs ou raies ?
O : Bruiteurs
D : Raz bruiteurs
O : **Annuler**
D : Que dois-je faire ?
 # la phrase reconnue reste en mémoire #
O : Raz
D : Raz bruiteurs

3.3.6 Le changement de sujet

On distingue deux types de changement de sujet. Le changement définitif est autorisé pendant la phase de mise au point. L'utilisateur donne un ordre n'ayant pas de rapport avec l'opération en cours. DIAPASON se recale sur ce dernier et oublie le précédent.

EXEMPLE 3.8

O : Affecter
D : Affecter quelle image ?
O : **Gamme 40**
D : Gamme 40
 # exécution de gamme 40 #

Le changement provisoire dans le but d'exécuter une sous-tâche ou d'effectuer une action qui devait être réalisée antérieurement n'est pas autorisé, aucun mécanisme ne permettant de récupérer les éléments de l'ordre initial. Les sous-dialogues imbriqués ne sont pas compris.

EXEMPLE 3.9

O : Affecter image 1
D : Affecter image 1 à quelle case ?
O : Effacer raies
D : Effacer raies image 1
 # exécution de effacer raies image 1 #
O : **Cases 2 à 3** *(réponse à la question)*
D : Je ne vous ai pas compris

3.4 EXECUTION DE L'ORDRE

On suppose que la commande et les paramètres sont corrects. Il faut donc maintenant que DIAPASON exécute l'ordre donné par l'opérateur. L'exécution ne se fait pas de manière automatique, il faut confirmer le couple (commande, paramètres). Il existe dans DIAPASON trois types de confirmation :

- la **confirmation explicite** impose au locuteur de valider l'ordre reconnu. Tant que cet ordre n'aura pas été confirmé ou annulé le système refusera toute autre phrase,
- la **confirmation implicite** permet à un ordre d'être exécuté si le système ne détecte pas de contestation dans les instants qui suivent l'interprétation de l'ordre,
- dans le cas de la **confirmation inexistante**, l'ordre est immédiatement exécuté sans aucune intervention de l'opérateur.

Il faut associer à chaque commande un type de confirmation. Cela se fera en fonction des conséquences qu'entraînent l'exécution de l'ordre dans le fonctionnement de l'application.

EXEMPLE : Pour l'application SONAR une libération de mémoire est une action qui fait perdre irrémédiablement des données, on lui attribue donc une confirmation explicite. Nous n'avons pas retenu de confirmation implicite pour l'application SONAR, on pourrait l'envisager par exemple pour l'ordre répartition mémoire. Un changement de cap (un des paramètres SONAR) n'a pas de conséquence grave sur son fonctionnement, l'opérateur peut revenir sur le changement de valeur, on lui attribue donc une confirmation inexistante.

EXEMPLE 3.10

	O	:	Libérer mémoire VPR
confirmation	D	:	Confirmez : libérer mémoire VPR
explicite	O	:	Ok
	D	:	Ok
			# exécution de libérer mémoire VPR #
	O	:	Répartition mémoire préétablie 1
confirmation	D	:	Répartition mémoire préétablie 1
implicite	O	:	Cap bâtiment
			# exécution de répartition mémoire préétablie 1 #
	D		Cap bâtiment
confirmation	O	:	Cap bâtiment
inexistante	D	:	Cap bâtiment
			# exécution de cap bâtiment #

Les deux premiers éléments composant un ordre sont la commande et les paramètres. Un ordre n'est complet qu'avec la confirmation qui lui est associée.

3.5 ENCHAINEMENT DES ORDRES

Un ordre vient d'être exécuté. Un des points forts des systèmes de dialogue [CARRE 1991] mis en œuvre dans DIAPASON est de pouvoir alors utiliser des éléments de cet ordre pour exécuter le suivant. L'interprétation d'une phrase ne se fait pas seulement en fonction des éléments recueillis dans la phrase courante, mais aussi en utilisant ceux du contexte global du dialogue. L'opérateur a donc la possibilité dans notre système d'élider la commande ou un des paramètres. DIAPASON se chargera alors de récupérer les parties manquantes. Dans les exemples qui suivent l'ellipse sera représentée par {}.

L'ellipse sur la commande est utilisée pour effectuer une même opération sur des objets différents.

EXEMPLE 3.11

O : Libérer mémoire VPR
D : Libérer mémoire VPR
O : {} Mémoire CLASS
D : **Libérer** mémoire CLASS
O : {} Mémoire VPLB
D : **Libérer** mémoire VPLB

L'ellipse sur le paramètre est utilisée pour effectuer plusieurs actions sur un même objet .

EXEMPLE 3.12

O : Afficher bruiteur ALPHA 1
D : Afficher bruiteur ALPHA 1
O : Lever de doute {} gauche
D : Lever de doute **bruiteur ALPHA 1** gauche
O : Graphique annexe fréquence {}
D : Graphique annexe fréquence **bruiteur ALPHA 1**

Il se pose alors le problème du vieillissement des données. Leur durée de vie ne devra pas être excessive. Le mécanisme de récupération des éléments doit fournir non seulement des éléments corrects mais aussi des éléments qui soient toujours valides à l'instant présent. Il faut d'autre part que l'opérateur puisse comprendre la démarche de la machine lorsqu'elle propose une interprétation, il n'a pas forcément en tête l'ordre précis qu'il avait donné quelques phrases ou minutes auparavant. Pour cette raison DIAPASON ne tient compte uniquement que du contexte de quelques ordres précédents (critère paramètrable selon l'application).

3.6 REPETITION DE L'EXECUTION

DIAPASON offre la possibilité de répéter le dernier ordre exécuté.

EXEMPLE 3.13

 # largeur initiale 20 Hz #
O : Largeur plus
D : Largeur plus
 # la largeur passe à 40 Hz #
O : Encore
D : Encore
 # la largeur passe à 80 Hz #

Tous les ordres ne peuvent être répétés, DIAPASON envoie dans ce cas un message d'erreur.

EXEMPLE 3.14

O : Extraction demon
D : Extraction demon
 # affichage du pavé de saisie de l'extraction demon #
O : Encore
D : L'extraction demon est déjà en cours de saisie

3.7 ANNULATION OU CORRECTION APRES EXECUTION

L'annulation ou la correction d'un ordre est parfois possible après exécution, certains ordres l'autorisent d'autres non. Cette possibilité est liée aux fonctionnalités de l'application. La phrase pour annuler l'ordre est dans ce cas "Négatif".

EXEMPLE 3.15

O : Gamme 40
D : Gamme 40
 # exécution de gamme 40 #
O : Négatif
D : Négatif
 # restauration de l'ancienne valeur de gamme #

O : Peignes harmoniques
D : Peignes harmoniques
 # lancement de l'outil graphique #
O : Négatif
D : L'annulation de l'ordre est impossible

EXEMPLE 3.16

O : Gamme 40
D : Gamme 40
 # exécution de gamme 40 #
O : Négatif 80
D : Gamme 80
 # exécution de gamme 40 #

O : Peignes harmoniques
D : Peignes harmoniques
 # lancement de l'outil graphique #
O : Négatif de résonance
D : La correction de l'ordre est impossible

3.8 LES GRAMMAIRES DU DIALOGUE

3.8.1 Notations et conventions

Pour fonctionner, DIAPASON doit reconnaître une phrase prononcée au microphone ou tapée au clavier. Il existe dans le système, une description de ces phrases : les grammaires. Elle contient un ensemble d'axiomes décrivant chacun les phrases qui peuvent être entrées dans un contexte donné, un contexte correspondant aux cas de figures présentés dans le paragraphe précédent : entrée d'un ordre initial, correction, enchaînement des ordres, complément d'information.

Dans les paragraphes qui suivent, nous présenterons le contenu de la grammaire et nous l'illustrerons à l'aide de l'ordre GRAPHIQUE ANNEXE de l'application SONAR. Nous utiliserons les conventions suivantes :

n le nombre de paramètres de l'ordre pour un arrangement donné,

P_0 le premier élément de l'ordre,

P_i le paramètre i, $i \neq 0$,

\+ signifie suivi de.

#SEUL# correspond à toutes les possibilités d'éléments isolés : P_i, $i \in [0,n]$,

#SEUL*# correspond à toutes les possibilités d'éléments isolés : P_i, $i \in [1,n]$,

#TOUS# correspond à tous les éléments d'un arrangement mis bout à bout : $P_0+P_1+\ldots+P_i+\ldots+P_n$ $i \in [0,n]$,

#COMB# correspond à toutes les combinaisons de paramètres autorisées par le système : $P_i+\ldots+P_j$ $i \in [0,n]$, $j \in [0,n]$ tels que $i \neq j$ (combinaisons $\neq$ #SEUL#) et non($i \neq 0$ et $j \neq n$) (combinaisons $\neq$ #TOUS#).

#COMB*# correspond à toutes les combinaisons de paramètres autorisées par le système : $P_i+\ldots+P_j$ $i \in [1,n]$, $j \in [1,n]$ tels que $i \neq j$ (combinaisons $\neq$ #SEUL#) et non($i \neq 0$ et $j \neq n$) (combinaisons $\neq$ #TOUS#).

Les grammaires sous-jacentes à la définition des ordres sont de type contexte libre ou plus précisément des grammaires sémantiques [BURTON 1976] [BONNET 1980] et peuvent être présentées sous forme de règles. Nous avons adopté des conventions pour une représentation graphique permettant de distinguer les différents éléments qui composent un ordre. Ces conventions ont servi de base pour la spécification de l'éditeur de structures syntaxico-sémantiques et sont présentées dans le paragraphe 6.3.2.3.

EXEMPLE 3.18 : cet exemple présente l'ordre GRAPHIQUE sous forme de représentation graphique (§6.3.2.3) et sous forme de règles de grammaire (les règles de désignation des éléments sont données avec les spécifications du compilateur de structures syntaxico-sémantiques en ANNEXE E) . L'ordre GRAPHIQUE correspond à l'affichage de données graphiques en haut de l'image Veille.

ordre GRAPHIQUE sous forme graphique :

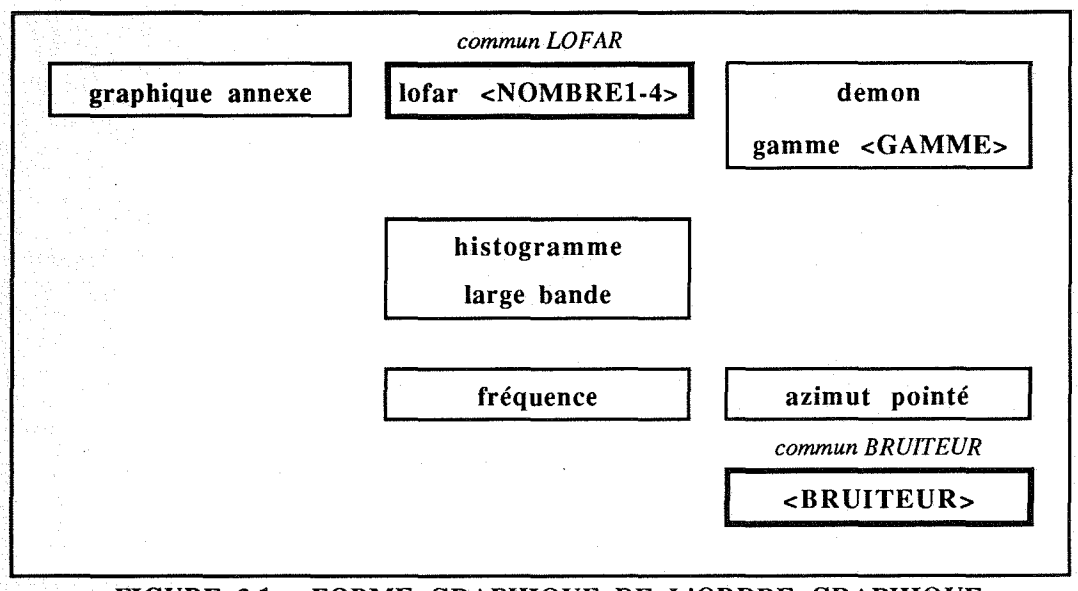


FIGURE 3.1 : FORME GRAPHIQUE DE L'ORDRE GRAPHIQUE

Les non-terminaux <NOMBRE1-4>, <GAMME> et <BRUITEUR> sont supposés définis.

ordre GRAPHIQUE sous forme de règles :
Dans ce formalisme, inspiré des règles de syntaxe d'un logiciel unix YACC [JOHNSON 1978], le caractère ':' marque le début de définition d'une règle, le caractère '|' les alternatives, le caractère ';' la fin de la règle. Les terminaux commencent par le caractère '_'. Ce formalisme est utilisé pour les outils compilateur et formateurs.

```
ordre_graphique     : o_graphique_a01_p0 c_graphique_a01_p1_lofar o_graphique_a01_p2
                    | o_graphique_a01_p0 o_graphique_a02_p1
                    | o_graphique_a01_p0 o_graphique_a03_p1 o_graphique_a03_p2
                    | o_graphique_a01_p0 o_graphique_a03_p1 c_graphique_a04_p2_bruiteur
                    ;
```

```
o_graphique_a01_p0  : _graphique _annexe
                    ;
c_graphique_a01_p1_lofar : _lofar <NOMBRE1-4>
                    ;
o_graphique_a01_p2  : _demon
                    | _gamme <GAMME>
                    ;
o_graphique_a02_p1  : _histogramme
                    | _large _bande
                    ;
o_graphique_a03_p1  : _fréquence
                    ;
o_graphique_a03_p2  : _azimut _pointé
                    ;
c_graphique_a04_p2_bruiteur : <BRUITEUR>
                    ;
<NOMBRE1-4>         : _1
                    | _2
                    | _3
                    | _4
                    ;
```

3.8.2 Les axiomes des ordres complets

Les axiomes des ordres complets correspondent aux ordres comportant tous les éléments. Il s'agit des formes syntaxiques #TOUS#.

EXEMPLE 3.19 : Formes syntaxiques complètes de l'ordre GRAPHIQUE :

```
complet_graphique   : o_graphique_a01_p0 c_graphique_a01_p1_lofar o_graphique_a01_p2
                    | o_graphique_a01_p0 o_graphique_a02_p1
                    | o_graphique_a01_p0 o_graphique_a03_p1 o_graphique_a03_p2
                    | o_graphique_a01_p0 o_graphique_a03_p1 c_graphique_a04_p2_bruiteur
                    ;
```

ce qui donne les phrases :
graphique annexe lofar <NOMBRE1-4> demon
graphique annexe lofar <NOMBRE1-4> gamme <GAMME>
graphique annexe histogramme
graphique annexe large bande
graphique annexe fréquence azimut pointé
graphique annexe fréquence bruiteur <BRUITEUR>
graphique annexe fréquence <BRUITEUR>
graphique annexe fréquence bruiteur pointé

3.8.3 Les axiomes de correction

Les axiomes de correction donnent la syntaxe des modifications que l'on peut effectuer sur le dernier ordre reconnu. Ils portent essentiellement sur la commande ou sur un ou plusieurs paramètres, ce qui correspond aux formes :

négatif+#SEUL#

négatif+#COMB*#

Toutes les corrections ne sont pas possibles, à un moment donné seules les formes compatibles avec la branche courante sont valides.

EXEMPLE 3.20 : Formes syntaxiques de correction de l'ordre GRAPHIQUE :

```
correction_graphique : _negatif o_graphique_a01_p0
                     | _negatif c_graphique_a01_p1_lofar
                     | _negatif o_graphique_a01_p2
                     | _negatif o_graphique_a02_p1
                     | _negatif o_graphique_a03_p1
                     | _negatif o_graphique_a03_p2
                     | _negatif c_graphique_a04_p2_bruiteur
                     | _negatif c_graphique_a01_p1_lofar o_graphique_a01_p2
                     | _negatif o_graphique_a03_p1 o_graphique_a03_p2
                     | _negatif o_graphique_a03_p1 o_graphique_a04_p2
                     ;
```

ce qui donne en autre les phrases :
négatif graphique annexe
négatif lofar <NOMBRE1-4>
négatif demon
négatif gamme <GAMME>
négatif lofar <NOMBRE1-4> demon
négatif lofar <NOMBRE1-4> gamme <GAMME>
négatif histogramme
négatif large bande
négatif fréquence ...

Les formes négatif+commande ne sont pas toutes pertinentes, seules les commandes très proches phonétiquement et ayant les mêmes types de paramètres ont besoin d'une correction (exemple : afficher / effacer, intégration / pondération). Pour toutes les autres commandes elles ne présentent pas un grand intérêt et encombrent inutilement le système de reconnaissance.

EXEMPLE 3.21

O : Pondération on
D : Intégration on
O : **Négatif pondération**
D : Pondération on

O : Raz bruiteur pointé
D : Quel nom de baptême pour baptiser bruiteur pointé ?
O : **Négatif raz**
D : Je ne vous ai pas compris !

3.8.4 Les axiomes d'enchaînement

Les axiomes d'enchaînement permettent l'ellision d'un des éléments qui compose un ordre.

3.8.4.1 Enchaînement sur le même ordre

L'enchaînement le plus simple consister à élider la commande. Il s'agit donc des formes #SEUL*# et #COMB*# . Tous les enchaînements ne sont pas possibles, à un moment donné seules les formes compatibles avec la branche courante sont valides.

EXEMPLE 3.22 : Formes syntaxiques d'enchaînement de l'ordre GRAPHIQUE :

```
correction_graphique : o_graphique_a01_p0
                     | c_graphique_a01_p1_lofar
                     | o_graphique_a01_p2
                     | o_graphique_a02_p1
                     | o_graphique_a03_p1
                     | o_graphique_a03_p2
                     | c_graphique_a04_p2_bruiteur
                     | c_graphique_a01_p1_lofar o_graphique_a01_p2
                     | o_graphique_a03_p1 o_graphique_a03_p2
                     | o_graphique_a03_p1 o_graphique_a04_p2
                     ;
```

ce qui donne en autre les phrases :
lofar <NOMBRE1-4>
B_1 : demon
B_1 : gamme <gamme>
lofar <NOMBRE1-4> demon
lofar <NOMBRE1-4> gamme <gamme>
histogramme
large bande
B_3 : <BRUITEUR> ...

EXEMPLE 3.23

O	:	Graphique annexe LOFAR 1 gamme CD
D	:	Graphique annexe LOFAR 1 gamme CD
O	:	LOFAR 2
D	:	Graphique annexe LOFAR 2 gamme CD
O	:	gamme EF
D	:	Graphique annexe LOFAR 2 gamme EF
O	:	bruiteur pointé *#branche invalide#*
D	:	Je ne vous ai pas compris !

3.8.4.2 Enchaînement sur un ordre différent

L'opérateur peut enchaîner sur un ordre différent en faisant l'ellision d'un ou plusieurs éléments communs avec l'ordre précédent, cela correspond aux formes :
$P_0+\ldots+P_{i-1}+P_{i+1}+\ldots+P_n$ $i \in [1,n]$ et P_i un élément commun.
Les formes d'enchainement sont regroupé par élément commun.

EXEMPLE 3.24 : Formes syntaxiques d'enchaînement de l'ordre GRAPHIQUE :

```
enchaine_lofar      : o_graphique_a01_p0 o_graphique_a01_p2
                    …
                    ;
enchaine_bruiteur   : o_graphique_a01_p0 o_graphique_a03_p1
                    | o_afficher_a01_p0
                    …
                    ;
```

ce qui donne pour les éléments de l'ordre GRAPHIQUE les phrases :

commun LOFAR : graphique annexe demon
graphique annexe gamme <GAMME>

commun BRUITEUR : graphique fréquence

EXEMPLE 3.25

O : Graphique annexe fréquence ALPHA 2
D : Graphique annexe fréquence ALPHA 2
O : afficher {}
D : afficher ALPHA 2
O : {} BRAVO 3
D : afficher BRAVO 3
O : Graphique annexe fréquence {}
D : Graphique annexe fréquence BRAVO 3

3.8.5 Les axiomes de réponse

Dans certains cas de figure DIAPASON est amené à poser une question à l'opérateur. Il faut donc prévoir un axiome de réponse. Les questions posées portent sur les paramètres. Cela correspond aux formes #SEUL*# et #COMB*# donc ce sont les formes identiques aux enchaînements sur le même ordre.

3.8.6 Les axiomes système

Les axiomes systèmes correspondent aux phrases de gestion de dialogue :
- "ok" pour la confirmation de l'ordre,
- "annuler" pour l'annulation de l'interprétation,
- "négatif" pour l'annulation de la reconnaissance,
- "encore" pour la répétition de l'ordre.

3.8.7 Validité des axiomes

En fonction de l'état courant du dialogue et de la tâche, seul un nombre limité d'axiomes est valide. Il s'agit, lorsque l'ordre i a été reconnu :
- des axiomes systèmes,
- des formes syntaxiques de tous les ordres valides dans le contexte courant de l'application,
- des formes de correction de l'ordre i,
- des formes d'enchaînement sur l'ordre i,

- des formes d'enchaînement sur les ordres valides ayant des éléments communs identiques à i,
- des formes de réponse à une question s'il y a une question posée par DIAPASON.

Initialement seuls les axiomes système et les formes syntaxiques de tous les ordres valides sont permis.

Chapitre 4

ARCHITECTURE GENERALE DE DIAPASON

4.1 INTRODUCTION

DIAPASON reprend en partie les caractéristiques de PARTNER [MORIN 1987]. Il est composé de trois modules principaux TACHE, DIALOGUE et RECONN [FIGURE 4.1].

Le **module de gestion du dialogue DIALOGUE** contrôle l'ensemble du système de dialogue. Cette fonction est assurée par un automate d'états finis qui régule l'acquisition des infomations. Il assure la gestion de l'historique du dialogue. C'est à son niveau que sont détectées la confirmation et les contestations. Il effectue des demandes de phrase au module RECONN (1) et reçoit en retour la représentation syntaxico-sémantique de l'énoncé de l'opérateur (2). Il interroge le module TACHE (3) pour obtenir les informations spécifiques à l'ordre en cours de compréhension (cohérence de l'ordre, type de confirmation associé...) qui lui répond (4). Ce module est entièrement indépendant de l'application.

Le **module de gestion de la tâche TACHE** assure deux fonctions. D'une part il gère les informations spécifiques à l'application à l'aide d'un réseau syntaxico-sémantique (syntaxe et contexte de la tâche). D'autre part il assure la gestion de la communication avec l'application (5).

Le **module de reconnaissance RECONN** assure la gestion des entrées vocales et écrites du système. Il construit la représentation syntaxico-sémantique du treillis de mots issu du système de reconnaissance de la parole (6) ou récupère la représentation syntaxico-sémantique de la phrase tapée au clavier (7). Il transmet la structure au module de gestion de dialogue (2). Il s'appuie sur des grammaires syntaxico-sémantiques.

Un module secondaire d'aide à l'opérateur **AIDE** a été développé pour l'application SONAR. Il sera présenté dans le chapitre 5.

L'entrée souris est gérée au niveau de l'application, ce qui fait qu'actuellement on ne peut traiter la multimodalité. Nous n'avons pas eu au cours de nos études l'occasion d'étudier ces aspects. Pour la première application (application SONAR) qui a donné naissance à une première version du système de dialogue, nous n'avons pas eu le temps pour nous y intéresser, et pour la seconde(application DIVA), il n'était pas possible de récupérer l'entrée souris au niveau du module de gestion des entrées.

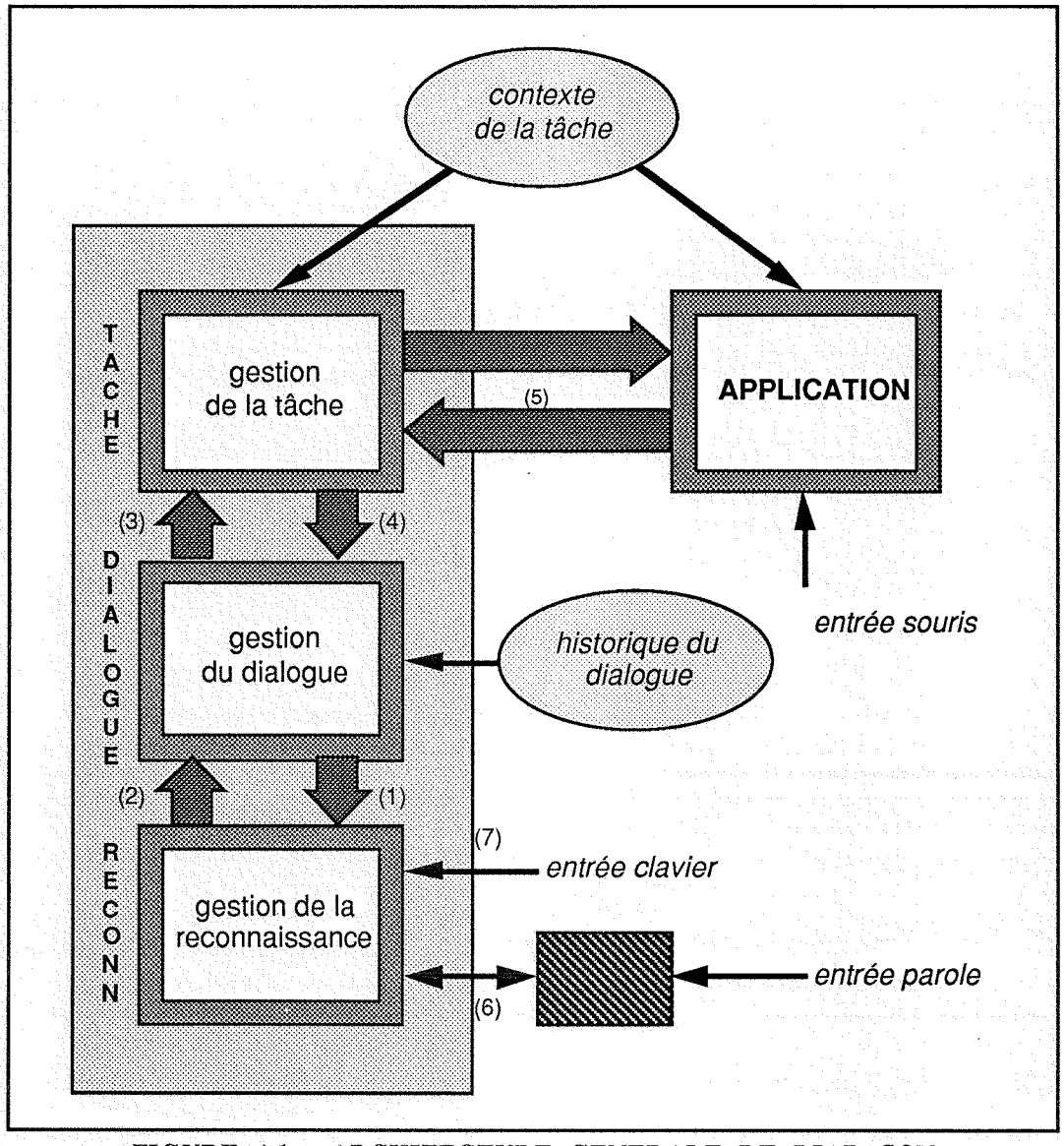


FIGURE 4.1 : ARCHITECTURE GENERALE DE DIAPASON

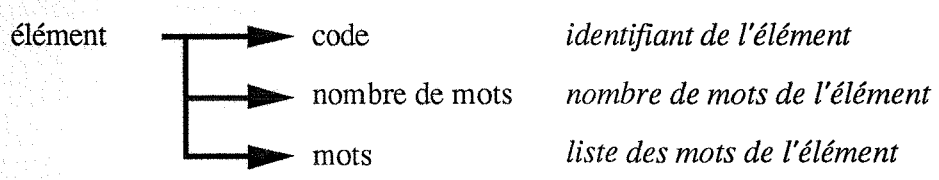
Nous étudierons dans ce chapitre le fonctionnement de chacun de ces modules en présentant leur structure interne, les échanges qu'ils réalisent avec les autres modules et les spécificités propres à leur fonction. Avant de les présenter en détail, nous nous intéresserons à la structure syntaxico-sémantique qui est à la base de tous les échanges entre les modules de DIAPASON. Nous verrons dans les paragraphes qui suivent le fonctionnement de chacune de ces composantes en détaillant leur structure interne, les échanges qu'elles réalisent avec les autres modules et les spécificités propres à leurs fonctions.

Cette architecture modulaire permet d'obtenir une grande indépendance entre les différentes composantes du système. Ce qui est particulier à l'application est alors très localisé (dans TACHE) et le système DIAPASON possède donc lui-même une relative indépendance vis-à-vis de l'application.

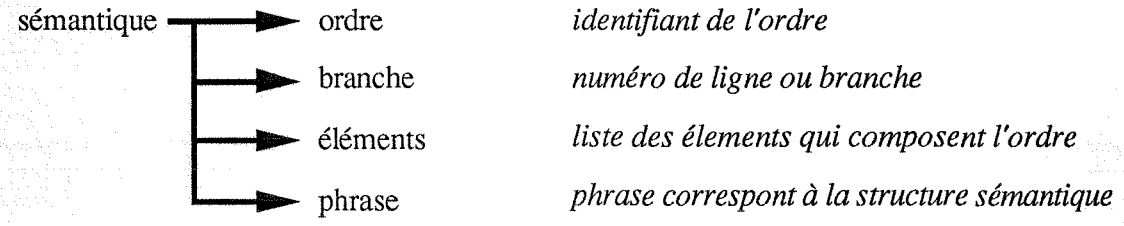
4.2 LA STRUCTURE SYNTAXICO-SEMANTIQUE

Comme nous le précisions dans le paragraphe précédent, la structure syntaxico-sémantique est à la base de tous les échanges entre les modules de DIAPASON. Une représentation syntaxico-sémantique de l'énoncé de l'opérateur est construite au niveau du module RECONN. Elle est transmise au module DIALOGUE qui la conserve dans l'historique du dialogue. L'automate manipule ces structures élément par élément, on retrouve ainsi la décomposition en prédicat/arguments qui caractérise le langage. Les structures syntaxico-sémantiques modèles sont stockées dans le réseau syntaxico-sémantique de TACHE.

Structure d'un élément :



Structure syntaxico-sémantique :



EXEMPLE 4.1 : la représentation syntaxico-sémantique de la phrase "graphique annexe large bande" est la suivante :

ordre :	20	
branche :	2	
elements[0] ->	code	O_GRAPHIQUE_A01_P0
	nombre_mots	2
	mot[0]	graphique
	mot[1]	annexe
elements[1] ->	code	O_GRAPHIQUE_A02_P1
	nombre_mots	2
	mot[0]	large
	mot[1]	bande
phrase :	"graphique annexe large bande"	

4.3 LE MODULE DE GESTION DU DIALOGUE

4.3.1 La structure interne de DIALOGUE

Le module de gestion du dialogue contrôle l'ensemble du système à partir d'un automate d'états finis et assure le gestion de la mémoire du dialogue. On trouve donc deux sous-modules, MOTEUR et MEMOIRE, internes à DIALOGUE chargés de gérer ces deux aspects [FIGURE 4.2].

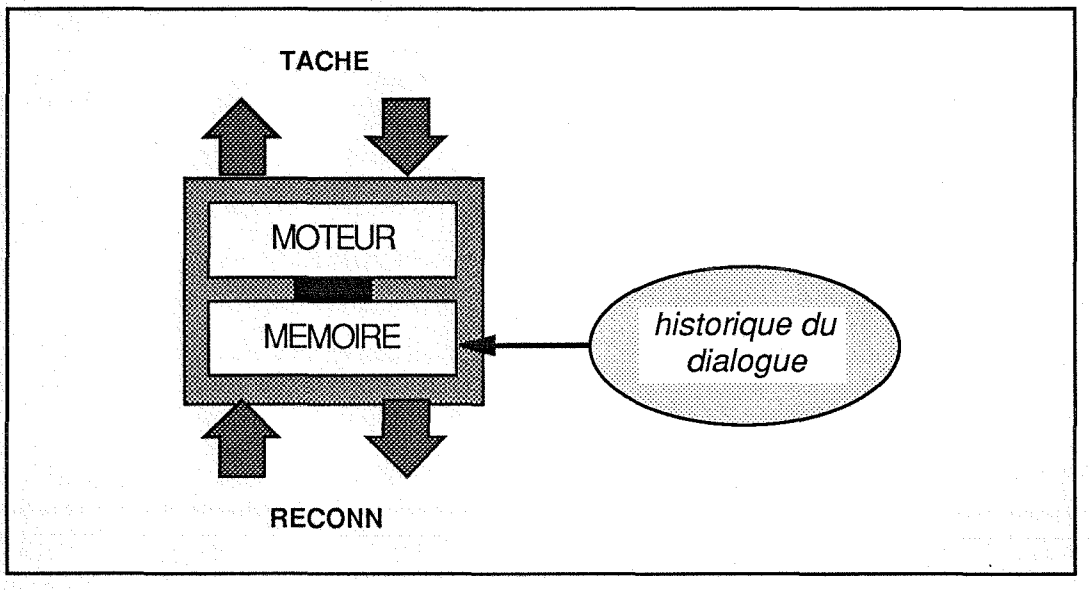


FIGURE 4.2 : STRUCTURE INTERNE DE DIALOGUE

4.3.2 Le sous-module MOTEUR

4.3.2.1 Introduction

Le sous-module MOTEUR est la composante qui dirige l'ensemble de DIAPASON. Il est construit en fonction d'un modèle de dialogue, lui-même dépendant d'un type de tâche : la commande de machine. Cela explique que ce sous-module traite à la fois des aspects liés au dialogue et à la tâche. Il y a donc un choix à faire : implémenter ce module au niveau de DIALOGUE ou de TACHE. La première implémentation l'avait placé au niveau de TACHE, à cette époque la notion de réseau syntaxico-sémantique n'avait pas été développée et les connaissances relatives à la tâche étaient noyées dans MOTEUR. Nos dernières réflexions nous ont amené à reconsidérer ce choix. En partant du fait que MOTEUR s'appuyait fortement sur le modèle du dialogue, nous l'avons déplacé au niveau de DIALOGUE, le réseau syntaxico-sémantique de TACHE permettant de regrouper les connaissances spécifiques à l'application.

4.3.2.2 L'automate d'états finis

La structure de MOTEUR repose sur un automate d'états finis dont les états de satisfaction sont : exécution de l'ordre ou incohérence de l'ordre acceptée. Il

récupère successivement la commande et les paramètres en faisant une requête au sous-module MEMOIRE [FIGURE 4.3]. L'interrogation du réseau sémantique de TACHE lui permet d'affecter les champs de la requête.

En fait le fonctionnement ne se déroule pas de manière aussi linéaire, le module peut revenir dans un état antérieur. Les transitions se font en fonction des réponses de TACHE et des réponses fournies par MEMOIRE.

Le locuteur a la possibilité de corriger la phrase reconnue. Dans ce cas MEMOIRE envoie un signal de non satisfaction (arc correction sur la [FIGURE 4.4]) sur la commande ou un des paramètres, l'automate revient dans son état initial quelle que soit l'élément sur lequel porte la correction.

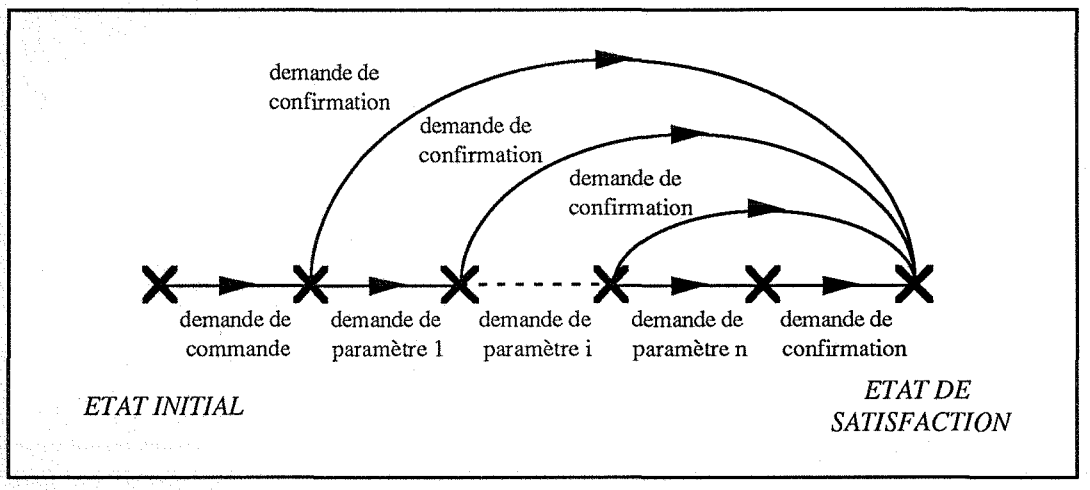


FIGURE 4.3 : STRUCTURE INTERNE DE MOTEUR

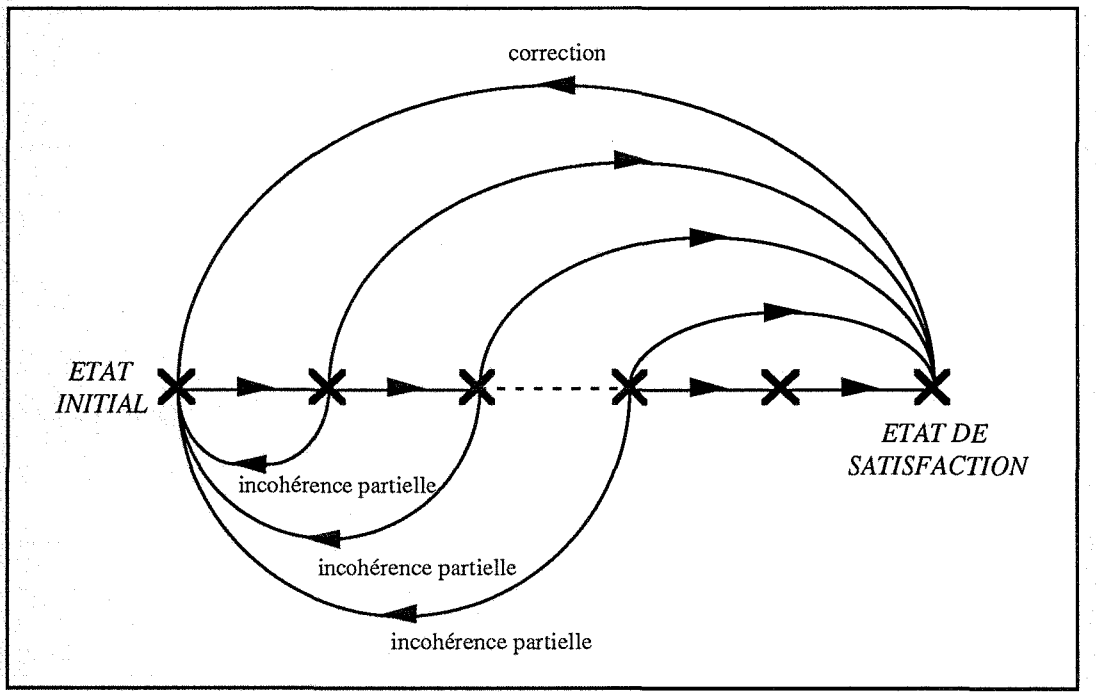


FIGURE 4.4 : STRUCTURE INTERNE DE MOTEUR

Il existe d'autre-part des conditions de validité des couples (commande, paramètres) qui sont liées au contexte de l'application (par exemple on ne peut effacer un objet qui n'est pas affichée). Si une de ces règles n'est pas respectée, on dit qu'il y a incohérence, l'automate revient dans son état initial dans l'attente d'un nouveau cycle (arcs incohérence). La procédure à déclencher est obtenue grâce au réseau syntaxico-sémantique.

La vérification de la cohérence se fait au fur et à mesure de la récupération des éléments, DIAPASON n'attend pas d'avoir l'ordre complet : c'est ce que nous appelons la cohérence partielle d'un ordre. Cela évite de poser inutilement une question lorsque les éléments manquants suivent l'élément provoquant l'incohérence. Cette vérification n'est pas toujours possible.

Les avantages de cette vérification de cohérence partielle d'un ordre sont illustrés dans l'EXEMPLE 4.2 qui présente l'ordre diriger composé de trois éléments : diriger+bruiteur+direction. L'exemple montre que le système de dialogue pose la question "Où diriger bruiteur pointé" (cas 1) alors qu'il peut déjà se rendre compte que cette question est erronée (cas 2).

EXEMPLE 4.2	O	:	Diriger bruiteur pointé
	D	:	Où diriger bruiteur pointé ?
cas 1	O	:	Vers azimut pointé
			# diriger bruiteur pointé vers azimut pointé #
	D	:	Il n'y a pas de bruiteur pointé
	O	:	Diriger bruiteur pointé
cas 2			*# diriger bruiteur pointé - incohérence partielle détectée #*
	D	:	Il n'y a pas de bruiteur pointé

Après prise en compte de ces phénomènes de cohérence partielle, on obtient alors l'automate de la [FIGURE 4.5]. On remarque qu'il y a des états qui ont fusionné, les mécanismes concernant les demandes de paramètres étant similaires. Il en va de même pour la cohérence et la correction. Les quatre états résultats sont :

- T0 état initial de l'automate,
- T1 état de réception d'un élément,
- T2 état de gestion de la confirmation,
- T3 état de gestion de l'incohérence.

Les six types d'arc sont :

- l'arc "demande de commande" qui permet d'obtenir l'élément commande de l'ordre,
- l'arc "demande de paramètre" qui permet d'obtenir les éléments paramètres de l'ordre,
- l'arc "demande de confirmation" qui permet de demander la confirmation de l'ordre,
- les arcs "fin d'échange" qui correspondent à l'acceptation par TACHE d'un ordre qu'il soit cohérent ou non,

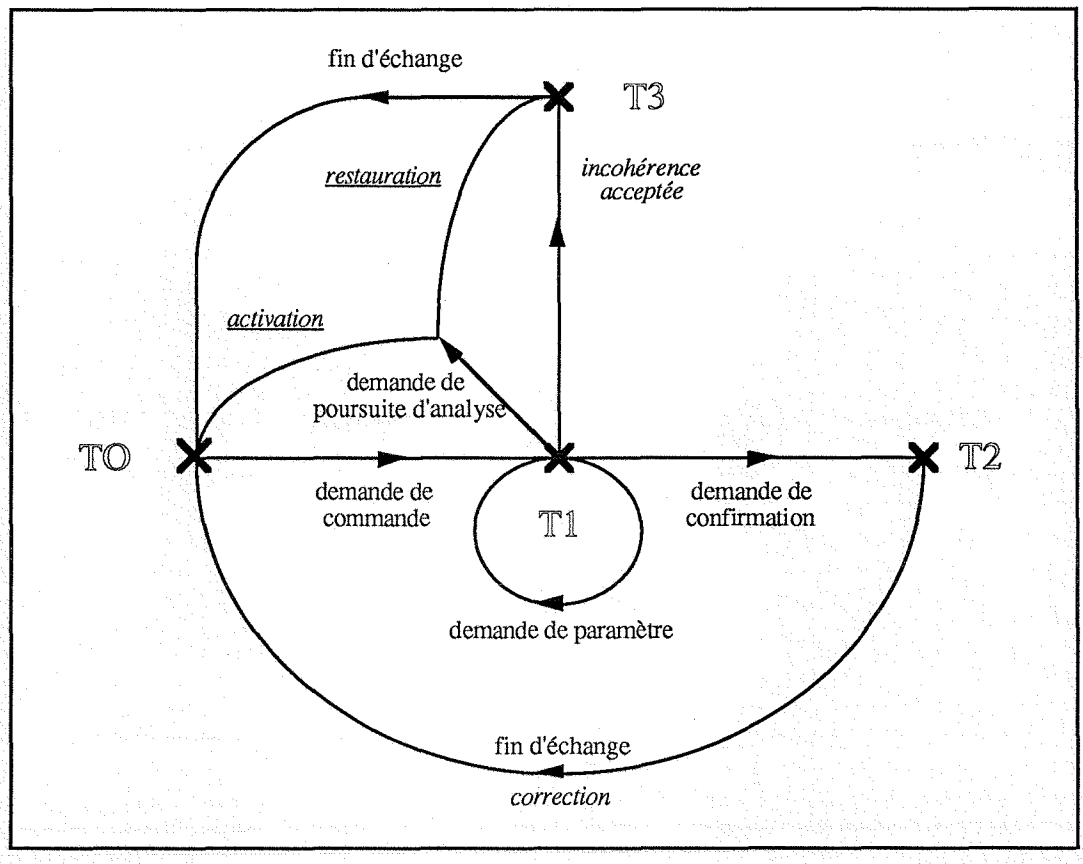


FIGURE 4.5 : STRUCTURE INTERNE DE MOTEUR

- l'arc "incohérence acceptée" qui correspond à l'acceptation de l'incohérence de l'ordre par le système,
- l'arc "poursuite de l'analyse" qui correspond au rejet de l'interprétation courante pour cause d'incohérence et se divise en deux :
  - "activation" lorsqu'il y a une autre interprétation possible et qui remet l'automate dans son état initial,
  - "restauration" lorsqu'il n'y a pas d'autre interprétation possible et qui permet l'acceptation d'une hypothèse incohérente qui avait dans un premier temps été rejetée.

4.3.2.3 Les requêtes formulées à MEMOIRE

Une grande partie des transitions de MOTEUR se fait sur l'envoi de requêtes formulées au sous-module MEMOIRE. Il en existe quatre types qui sont synthétisées dans la [FIGURE 4.6].

La **demande de valeur** est la requête qui permet de récupérer un élément. Elle comporte six champs :

DEMVAL	identifiant de la requête,
paramètre	numéro de paramètre, la commande est considérée comme le paramètre d'ordre 0,
éléments	les éléments précédemment acceptés,

message	question à poser à l'opérateur lorsque l'élément cherché n'est pas trouvé dans la mémoire,
axiomes	axiomes à utiliser si une question est posée,
mémoires	mémoires dans lesquelles est autorisée la recherche. C'est avec ce champ que MOTEUR indique l'existence d'une valeur par défaut.

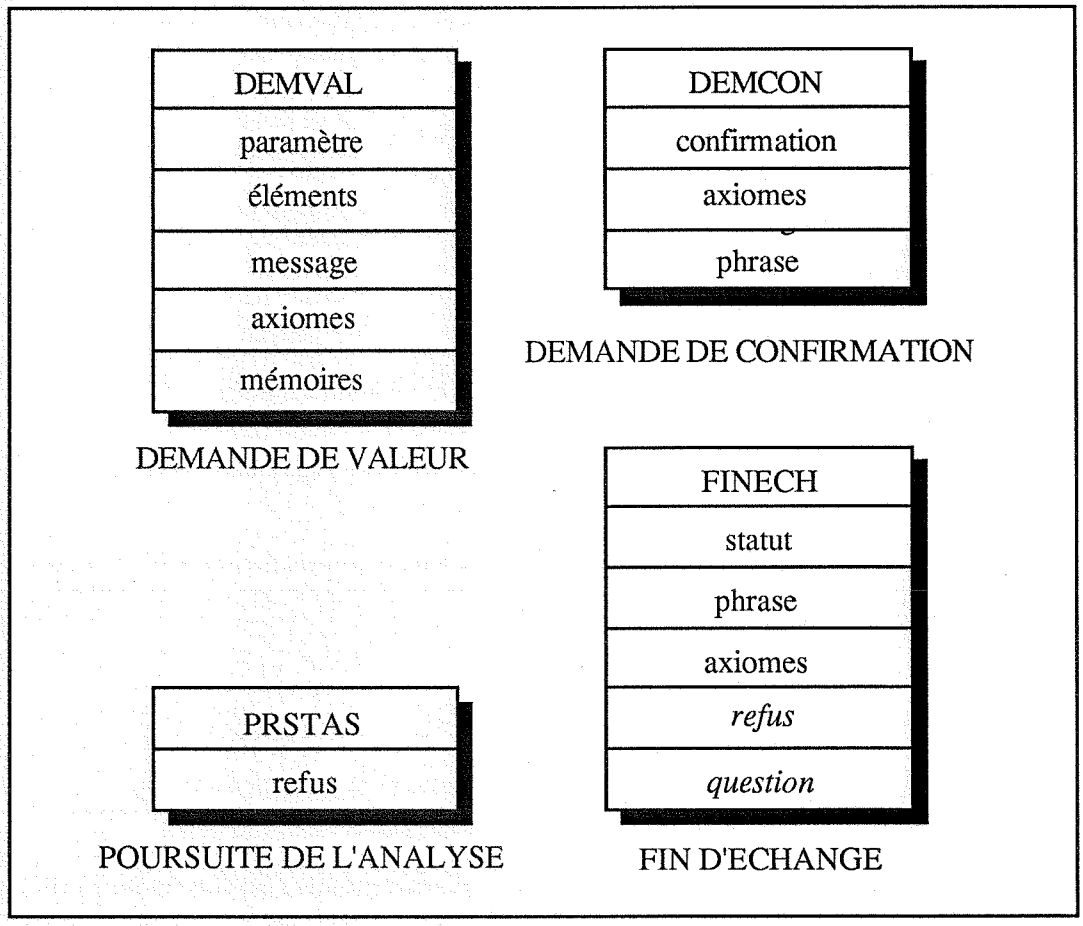


FIGURE 4.6 : LES REQUETES FORMULEES A MEMOIRE

Le **demande de confirmation** est la requête qui est envoyée lorsque MOTEUR possède tous les éléments d'un ordre. Il permet à DIALOGUE de gérer la confirmation ou la correction. La requête comporte quatre champs :

DEMCOM	identifiant de la requête,
confirmation	type de confirmation associé à l'ordre explicite, implicite ou inexistante,
axiomes	axiomes pour les phrases de correction,
phrase	forme syntaxique de l'ordre reconnu.

La **demande de la poursuite de l'analyse** est la requête envoyée lorsque TACHE détecte une incohérence. Elle permet à DIAPASON d'effectuer une

nouvelle interprétation si le niveau reconnaissance a fourni plusieurs hypothèses de phrases. La requête comporte deux champs :

PRSTAS	identifiant de la requête,
refus	indique les éléments ayant provoqué l'incohérence.

Le **signal de fin d'échange** est la requête qui indique à MEMOIRE que l'ordre a été retenu qu'il soit exécutable ou non. Il permet à DIALOGUE de gérer la suite des interventions de l'opérateur. Le nombre de champs est variable selon la conclusion de TACHE : ordre exécuté, ordre incohérent, ordre incohérent mais question posée, annulation de l'ordre précédent. Il existe quatre champs fixes :

FINECH	identifiant de la requête,
statut	conclusion de TACHE,
phrase	forme syntaxique de l'ordre reconnu,
axiomes	axiomes à utiliser pour la suite ;

auxquels s'ajoutent un champ complémentaire pour la conclusion ordre incohérent :

refus	indique les éléments ayant provoqué l'incohérence,

et deux champs complémentaires pour la conclusion ordre incohérent mais question posée :

refus	indique les éléments ayant provoqué l'incohérence,
question	élément sur lequel porte la question.

4.3.2.4 Le traitement de l'incohérence

Nous avons évoqué l'existence de conditions de validité pour le couple (commande, paramètres). Lorsqu'une de ces règles n'est pas respectée on dit qu'il y a incohérence. Elle a deux origines possibles :

- le niveau reconnaissance a proposé une hypothèse ne correspondant pas aux propos du locuteur,
- l'opérateur ne tient pas lui-même des propos cohérents.

Il faut donc que le système soit capable de déterminer si c'est le niveau reconnaissance qui s'est trompé où si c'est l'opérateur, en fait il faut décider si la reconnaissance est bonne ou mauvaise. Une méthode simple est d'utiliser le score de reconnaissance de la phrase et de fixer un seuil. Si le score est au dessus du seuil, on considère qu'il y a une bonne reconnaissance de la phrase prononcée et donc que c'est l'opérateur qui tient des propos incohérents. Si le score est en dessous du seuil, on considère qu'il y a une mauvaise reconnaissance, il faut examiner l'hypothèse suivante pour la phrase prononcée.

Ce mécanisme est implanté au niveau de MOTEUR. Lorsqu'il y a une mauvaise reconnaissance, MOTEUR fait une "demande de poursuite d'analyse" à MEMOIRE. Lorsqu'il n'y a aucune hypothèse cohérente, MEMOIRE envoie à MOTEUR un "signal de restauration", il permet de récupérer la "meilleure" hypothèse incohérente en passant dans l'état de gestion de l'incohérence. Actuellement la "meilleure" hypothèse est la première examinée, on pourrait imaginer une analyse plus complexe de toutes les phrases rejetées.

EXEMPLE 4.3 : Supposons que le niveau reconnaissance présente deux hypothèses de phrase dans l'ordre décroissant de score : "Afficher bruiteur ECHO 1" et "Afficher bruiteur DELTA 1". La première hypothèse est examinée par DIAPASON. Deux cas de figures peuvent alors se présenter :

(1) le bruiteur ECHO 1 existe effectivement et il est effacé, la phrase est acceptée, DIAPASON la propose à l'opérateur :

CAS 1 D : Afficher bruiteur ECHO 1

(2) le bruiteur ECHO 1 n'existe pas ou il est déjà affiché, DIAPASON passe alors à l'hypothèse suivante : "Afficher bruiteur DELTA 1". Deux nouveaux cas de figures se présentent alors :

(2.1) le bruiteur DELTA 1 existe et il est affiché ; la phrase est acceptée, DIAPASON la propose à l'opérateur :

CAS 2.1 D : Afficher bruiteur DELTA 1

(2.2) le bruiteur DELTA 1 n'existe pas ou il est déjà affiché ; DIAPASON passe à l'hypothèse suivante. Il n'y en a pas, MOTEUR reçoit le signal de restauration qui lui fait récupérer la première hypothèse examinée : "Afficher bruiteur ECHO 1". Selon l'origine de l'incohérence, DIAPASON affiche un des deux messages :

CAS 2.2.A D : Il n'y a pas de bruiteur ECHO 1

CAS 2.2.B D : Le bruiteur ECHO 1 est déjà affiché

Une étude statistique des scores a permis de déterminer une frontière entre la mauvaise reconnaissance et la bonne. Mais dans la plupart des cas, lorsque le score est en dessous du seuil, la phrase correcte n'est pas parmi les hypothèses présentées. Implémenter le mécanisme apporterait donc plus d'inconvénients que d'avantages, car des messages d'incohérence intempestifs seraient envoyés. Un filtrage a donc été effectué au niveau de la maquette de reconnaissance, elle ne renvoie que les hypothèses dont le score est supérieur au seuil. Ainsi le sous-module MOTEUR ne gère pas les scores et propose toujours la première hypothèse qui n'est pas incohérente s'il en existe une.

4.3.2.5 Exécution et incohérence

Dès qu'un ordre cohérent est confirmé, il faut l'exécuter. Mais auparavant, il faut vérifier que les objets désignés, si l'ordre en comporte, n'ont pas changé entre le moment où la cohérence est testée et le moment où l'ordre est effectivement exécuté. En cas de modification, il se produit une erreur d'exécution : le champ "statut" du signal fin d'échange [FIGURE 4.6] change et "phrase" contient le message d'erreur.

EXEMPLE 4.4 : L'hypothèse "Afficher bruiteur pointé" a été acceptée par DIAPASON. Au moment du test de la cohérence, il retient le nom du bruiteur. S'il s'avère différent à l'instant où il s'apprête à l'exécuter, il abandonne l'ordre et envoie un message d'erreur :

EXEMPLE 4.4 O : Afficher bruiteur pointé
 # l'opérateur désigne un autre bruiteur #
D : Le bruiteur pointé a changé

4.3.2.6 Le fonctionnement de MOTEUR - algorithme simplifié

La boucle principale du module :
La boucle principale consiste en une itération dont le corps permet de selectionner les traitements concernant l'état courant.

```
initialisations
tantque VRAI faire
    selon état_courant
    cas T0: /* ETAT INITIAL */
        traiter cas T0
    cas T1: /* ETAT DE RECEPTION DES VALEURS */
        traiter cas T1
    cas T2 : /* ETAT DE GESTION DE LA CONFIRMATION */
        traiter cas T2
    cas T3 : /* ETAT DE GESTION DE L'INCOHERENCE */
        traiter cas T3
    finselon
fintantque
```

L'état initial de l'automate :
Le traitement de l'état initial consiste à faire la demande de commande et à passer dans l'état de réception de valeur à la prochaine itération.

```
cas T0: /* ETAT INITIAL */
    on passe dans l'état de réception de valeur T1
    on fait une demande de valeur :
    - paramètre :  commande
    - éléments  :  pas d'élément
    - message   :  "Que dois-je faire?"
    - axiomes   :  toute phrase ou enchaînement sur l'ordre précédent ou sa correction
    - mémoires  :  selon les valeurs par défaut
fincas
```

L'état de réception de valeurs :
Avant de traiter l'arrivée de valeur, le système teste si le sous-module MEMOIRE n'a pas envoyé de signal d'activation. S'il s'agit effectivement d'une arrivée de valeur deux cas peuvent se produire.
Tous les éléments de l'ordre sont réunis, le système teste alors la cohérence totale. Si c'est cohérent il faut passer dans l'état de gestion de la confirmation sinon traiter l'incohérence. Si tous les éléments ne sont pas réunis, il teste la cohérence partielle. Si c'est cohérent il fait une demande de valeur, sinon il traite l'incohérence.
Le traitement de l'incohérence consiste tout d'abord à déterminer s'il accepte ou non l'incohérence. Si le système l'accepte, il passe dans l'état de gestion de

l'incohérence, sinon il effectue une demande de poursuite d'analyse. Selon la réponse de MEMOIRE, "signal d'activation" signifie qu'il y a une nouvelle interprétation donc il repasse dans l'état initial, "restaure" signifie qu'il n'y a pas de nouvelle interprétation, il passe dans l'état de gestion de l'incohérence.

```
cas T1: /* ETAT DE RECEPTION DES VALEURS */
   selon réponse
   cas signal d'activation
      on passe dans l'état initial T0
   fincas
   cas envoi de valeur
      si tous les éléments sont réunis alors
         on teste la cohérence totale
         si c'est cohérent alors
            on passe dans l'état de gestion de la confirmation T2
         sinon
            si l'incohérence est acceptée alors
               on passe dans l'état d'incohérence T3
            sinon
               on envoie une demande de poursuite d'analyse syntaxique
               si la réponse est le signal d'activation alors
                  on passe dans l'état initial T0
               sinon
                  on passe dans l'état initial T3
               finsi
            finsi
         finsi
      sinon
         on teste la cohérence partielle
         si c'est cohérent alors
            on fait une demande de valeur :
            - paramètre :  élément suivant
            - éléments  :  les éléments déjà réunis
            - message   :  question à poser en cas d'absence du paramètre
            - axiomes   :  toute phrase ou réponse à la question
            - mémoires  :  selon les valeurs par défaut
         sinon
            si l'incohérence est acceptée alors
               on passe dans l'état d'incohérence T3
            sinon
               on envoie une demande de poursuite d'analyse syntaxique
               si la réponse est le signal d'activation alors
                  on passe dans l'état initial T0
               sinon
                  on passe dans l'état initial T3
               finsi
            finsi
         finsi
      finsi
   fincas
   finselon
fcas
```

L'état de gestion de la confirmation :
La gestion de la confirmation commence tout d'abord par la demande de confirmation à MEMOIRE. S'il y a effectivement une réponse à la demande de confirmation, si la réponse est OUI, l'ordre est exécuté, un signal de fin d'échange est envoyé et le système passe alors dans l'état initial, si la réponse est NON, le système se met dans l'état initial afin de prendre en compte la nouvelle entrée. S'il n'y a pas de réponse, le système passe dans l'état initial et se met en attente d'une entrée.

```
cas T2 : /* ETAT DE GESTION DE LA CONFIRMATION */
    on fait la demande de confirmation
    on reçoit la réponse de MEMOIRE
    si il y a une réponse alors
        selon réponse
        cas OUI
            on exécute l'ordre
            on envoie fin d'échange :
            - statut     :  EXECUTE
            - phrase :  phrase correspondant à l'ordre
            on passe dans l'état initial T0
        cas NON
            on passe dans l'état initial T0
        finselon
    sinon
        on passe dans l'état initial T0
    finsi
fcas
```

L'état de gestion de l'incohérence :
La première action à effectuer dans l'état de gestion de l'incohérence est de récupérer l'hypothèse incohérente qui a été stockée dans une mémoire spécifique. Deux cas de figures peuvent alors se produire, cela dépend de l'origine de l'élément ayant provoqué l'incohérence. Soit l'élément vient de la mémoire à long terme et le système considère que l'interprétation est mauvaise, donc il pose une question, soit l'élément vient de la mémoire à court terme et le système envoie le message d'incohérence (ce mécanisme est actuellement réalisé au niveau de la tâche).

```
cas T3 : /* ETAT DE GESTION DE L'INCOHERENCE */
    on récupère le message et les axiomes de la première hypothèse incohérente
    si on décide de traiter l'incohérence alors
        on envoie fin d'échange
        - statut     :  INCOHERENT
        - phrase     :  message d'incohérence
        - refus      :  nature de l'incohérence
        - axiomes    :  tous les ordres, correction, enchaînement
    sinon
        on envoie fin d'échange
        - statut     :  IINCOHERENT ET QUESTION
        - phrase     :  question posée
        - refus      :  nature de l'incohérence
        - axiomes    :  tous les ordres, correction, enchaînement, réponse
    finsi
    on passe dans l'état initial T0
fcas
```

4.3.2.7 Fonctionnement de MOTEUR - un exemple

EXEMPLE 4.5 : Cet exemple présente le fonctionnement de MOTEUR pour plusieurs cas de figure. La phrase prononcée est "Graphique annexe LOFAR 1 gamme CD". Chaque cas est décomposé en étapes qui correspondent aux états successifs de l'automate.

1er CAS DE FIGURE :
L'ordre GRAPHIQUE est un ordre à confirmation inexistante.

(1) Au départ MOTEUR est dans son état initial T0
- il passe dans l'état T1,
- il fait une demande de valeur :
  requête(DEMVAL, 0, Ø, "Que dois-je faire ?", axiomes, mémoires)
  - où 0 indique qu'il faut rechercher une commande,
  - où Ø indique qu'il n'y a pas d'éléments,
  - où axiomes contient l'axiome pour reconnaître toute phrase de l'image VPR et l'axiome pour enchaîner sur l'ordre précédent ou le corriger,
  - où "Que dois-je faire ?" est la question à poser si MEMOIRE ne trouve pas d'élément,
  - où mémoires indique qu'il n'y a pas de valeur par défaut.

(2) MOTEUR est dans l'état T1
- il reçoit l'élément : "graphique annexe",
- tous les éléments ne sont pas réunis,
- il teste la cohérence partielle : c'est cohérent,
- il fait une demande de valeur :
  requête(DEMVAL, 1, éléments, "Quel graphique annexe ?", axiomes, mémoires)
  - où 1 indique qu'il faut rechercher le premier paramètre,
  - où éléments contient "garphique annexe" et permet de récupérer un élément compatible avec lui,
  - où axiomes contient contient l'axiome pour reconnaître toute phrase de l'image VPR et l'axiome pour répondre à la question et corriger l'élément déjà reconnu,
  - où mémoires indique qu'il n'y a pas de valeur par défaut.

(3) MOTEUR est dans l'état T1
- il reçoit la valeur : "LOFAR 1",
- tous les éléments ne sont pas réunis,
- il teste la cohérence partielle : c'est cohérent,
- il fait une demande de valeur :
  requête(DEMVAL, 2, éléments, , question, axiomes, mémoires)
  - où 2 indique qu'il faut rechercher le deuxième paramètre,
  - où éléments contient "graphique annexe"+"LOFAR 1",
  - où question est la phrase : "Quelle gamme pour graphique annexe ?",
  - où axiomes contient contient l'axiome pour reconnaître toute phrase de l'image VPR et l'axiome pour répondre à la question et corriger les éléments déjà reconnus,
  - où mémoires indique qu'il n'y a pas de valeur par défaut.

(4) MOTEUR est dans l'état T1
- il reçoit une valeur : "gamme CD",
- tous les éléments sont réunis,
- il teste la cohérence totale : c'est cohérent,
- il passe dans l'état T2.

(5) MOTEUR est dans l'état T2
- il fait une demande de confirmation :
  requête(DEMCON, INEXISTANTE, axiomes, phrase)
  - où INEXISTANTE est le type de confirmation associé à l'ordre,
  - où axiomes contient l'axiome pour reconnaître toute phrase de l'image VPR, les enchaînements sur l'ordre et l'axiome pour le corriger,
  - où phrase contient "Graphique annexe LOFAR 1 gamme CD" ;
- MEMOIRE répond qu'il y a satisfaction,
- TACHE exécute l'ordre "graphique annexe LOFAR 1 gamme CD",
- il envoie un fin d'échange :
  requête(FINECH, EXECUTE, phrase, axiomes)
  - où EXECUTE indique que l'ordre courant a été exécuté,
  - où phrase contient "Graphique annexe LOFAR 1 gamme CD",
  - où axiomes contient l'axiome pour reconnaître toute phrase de l'image VPR, les enchaînements sur l'ordre et l'axiome pour le corriger,
  - où phrase contient "Graphique annexe LOFAR 1 gamme CD" ;
- il revient dans l'état initial T0.

2ème CAS DE FIGURE :
L'ordre GRAPHIQUE est à confirmation explicite, on retrouve tout d'abord les étapes (1) à (4). L'étape suivante est :

(5') MOTEUR est dans l'état T2
- il fait une demande de confirmation :
  requête(DEMCON, INEXISTANTE, axiomes, phrase)
  - où INEXISTANTE est le type de confirmation associé à l'ordre,
  - où axiomes contient l'axiome pour reconnaître toute phrase de l'image VPR, les enchaînements sur l'ordre et l'axiome pour le corriger,
  - où phrase contient "Graphique annexe LOFAR 1 gamme CD",
- MEMOIRE envoie un message de demande de confirmation à l'utilisateur et répond à MOTEUR qu'il n'y a pas confirmation,
- MOTEUR passe dans l'état T0 et se met en attente d'une nouvelle entrée.

Supposons alors qu'une nouvelle entrée réveille MOTEUR.
S'il s'agit de la phrase "ok", les étapes (1) à (5) sont effectuées à nouveau.
S'il s'agit d'une contestation, les nouvelles étapes dépendent du type de contestation. Par exemple si le système avait reconnu LOFAR 2 à la place de LOFAR 1, l'opérateur aurait contesté par "négatif LOFAR 1", les étapes (1) à (4) se reproduiraient suivies de (5')

3ème CAS DE FIGURE :
Supposons maintenant que l'ordre GRAPHIQUE soit impossible au moment du traitement, on retrouve l'étape (1). Les étapes suivantes sont :

(2') MOTEUR est dans l'état T1
- il reçoit l'élément : "graphique annexe",
- tous les éléments ne sont pas réunis,
- il teste la cohérence partielle : c'est incohérent,
- il met l'interprétation dans la mémoire d'incohérence,
- il fait une demande de poursuite d'analyse,
- la réponse est "signal de restauration" (pas d'autre hypothèse),
- il passe dans l'état T3.

(3') MOTEUR est dans l'état T3
- il récupère dans la mémoire l'hypothèse incohérente,
- il fait une demande de fin d'échange
  requête(FINECH, INCOHERENT, refus, axiomes)
  - où INCOHERENT indique que l'ordre est incohérent,
  - où phrase contient "L'ordre graphique annexe est impossible",
  - où refus indique que c'est la commande qui provoque l'incohérence,
  - où axiomes contient l'axiome pour reconnaître toute phrase de l'image VPR, les enchaînements sur l'ordre graphique annexe, les corrections sur graphique annexe ;
- il passe dans l'état T0,
- il se met en attente d'une nouvelle entrée.

4ème CAS DE FIGURE :
Si pour des raisons indépendantes de l'opérateur la fin de sa phrase n'est pas reconnue "Graphique annexe LOFAR 1 ...", les étapes (1) à (3) restent identiques, mais la demande de valeur ne peut pas être satisfaite. La question "Quelle gamme pour garphique annexe LOFAR 1" est posée par le niveau dialogue et les axiomes correspondants sont envoyés à la maquette de reconnaissance. Le système attend une nouvelle entrée et à son réveil par la phrase "Gamme CD" , il effectue les étapes (1) à (5).

4.3.3 Le sous-module MEMOIRE

4.3.3.1 Introduction

Le sous-module de MEMOIRE [FIGURE 4.2] à la page 54, est un intermédiaire entre le module TACHE, chargé de gérer l'application, et le module RECONN, chargé de gérer les entrées écrites et orales du système. Il doit répondre aux requêtes formulées par MOTEUR, pour cela il utilise une stratégie de recherche de l'information dans les sources de connaissances dont il dispose : sa mémoire, des valeurs par défaut, le module de reconnaissance.

4.3.3.2 La structure interne de MEMOIRE

Le module de gestion du dialogue est entièrement subordonné à MOTEUR. Il se compose de quatre états correspondant chacun à une requête du module de gestion de la tâche :

- état de gestion de la demande de valeur,
- état de gestion de la demande de confirmation,
- état de gestion de la demande de poursuite d'analyse,
- état de gestion du signal de fin d'échange.

Chaque état s'efforce de gérer au mieux les appels au module de reconnaissance (détection de la correction, de l'annulation, de la confirmation, etc).

4.3.3.3 Les différents types d'échanges

Le sous-module MEMOIRE communique avec le sous-modules MOTEUR et le module de gestion de la reconnaissance. Il existe donc des échanges de messages entre ces modules.

a) les échanges de MEMOIRE vers MOTEUR

Il existe quatre types de messages que MEMOIRE envoie à MOTEUR, synthétisés [FIGURE 4.7].

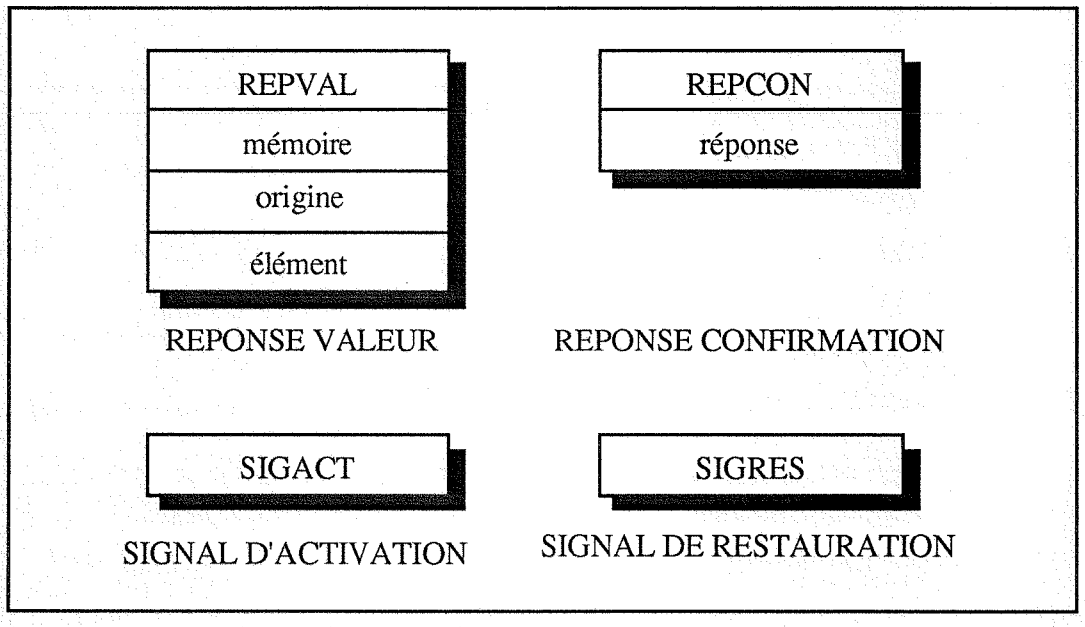


FIGURE 4.7 : LES ECHANGES DE MEMOIRE VERS MOTEUR

On distingue tout d'abord deux réponses directes aux requêtes formulées par MOTEUR.

La **réponse valeur** est la réponse à la demande de valeur. Elle comporte quatre champs :

REPVAL identifiant de la requête,

mémoire	indique dans quelle mémoire a été trouvé l'élément (la structure de la mémoire sera présentée par la suite),
origine	position dans la mémoire,
élément	élément trouvé dans la mémoire.

La **réponse à la confirmation** est la réponse à la demande de confirmation. Elle comporte deux champs :

REPCON	identifiant de la requête,
reponse	deux valeurs possibles pour la réponse : CONFIRMATION_OUI, CONFIRMATION_NON.

La valeur est déterminée en fonction du type de confirmation associé à l'ordre reconnu et de la représentation sémantique de la phrase fournie pour le module de gestion de la reconnaissance.

Il existe ensuite deux requêtes qui permettent la synchronisation entre les deux modules. Elles ne comportent toutes deux qu'un seul champ : l'identifiant de la requête.

Le **signal d'activation** remet l'automate de MOTEUR dans son état initial afin d'effectuer une nouvelle interprétation.

Le **signal de restauration** permet à MOTEUR de traiter l'incohérence lorsqu'il n'y a plus de nouvelle interprétation possible. Il permet de récupérer la "meilleure" interprétation parmi celles qui avaient été auparavant rejetées par le système.

b) les échanges de MEMOIRE vers RECONN

Le module de gestion du dialogue a la possibilité de faire appel au module de reconnaissance. Il existe deux types d'appel différent [FIGURE 4.8].

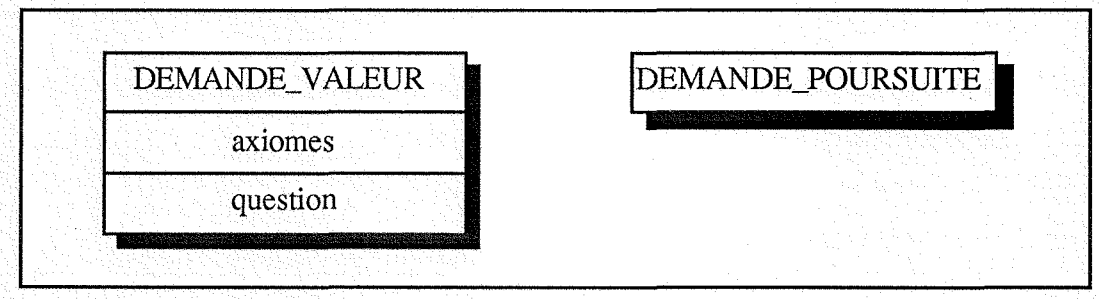


FIGURE 4.8 : LES ECHANGES DE DIALOGUE VERS RECONN

La **demande de valeur initiale** permet de se mettre en attente d'une nouvelle phrase et de récupérer la première hypothèse pour la phrase reconnue. Elle se compose de trois champs :

DEMANDE_VALEUR	est l'identifiant de la demande,
axiomes	les axiomes pour guider la reconnaissance,
question	indique si une question a été posée.

La **demande de poursuite** permet de récupérer l'hypothèse suivante pour la phrase courante. Elle est composée d'un champ unique :

DEMANDE_POURSUITE est l'identifiant de la demande.

4.4.3.4 La mémoire de DIALOGUE

Avant d'étudier la stratégie que MEMOIRE met en œuvre pour trouver la réponse aux requêtes formulées par MOTEUR, nous devons présenter la composante principale du module : l'historique du dialogue ou la mémoire. Il possède deux composantes.

La **mémoire à court terme** conserve toutes les hypothèses étudiées lors de la mise au point de l'ordre courant pour toute phrase prononcée. Le but de DIAPASON est d'arriver dans l'état de satisfaction de MOTEUR en recueillant commande, paramètres et satisfaction. Cette collecte se fait en lien avec des interventions de l'opérateur, lors de la demande de complément d'information ou de la correction d'un élément erroné. La mémoire à court terme conserve tous ces éléments. A chaque hypothèse on associe un **statut** qui est fonction de la réponse fournie par TACHE (refus de la commande ou du paramètre pour cause d'incohérence) ou fonction des corrections du locuteur (négation de commande, négation d'un paramètre). La valeur de *statut* évoluera en fonction de ces interventions.

La **mémoire à long terme** possède une porté plus globale, elle conserve les ordres complets exécutés par le système et les ordres incohérents finalement proposés à l'opérateur.

EXEMPLE 4.6 : cet exemple présente l'évolution du contenu des mémoires pour le dialogue suivant :

EXEMPLE 4.6	O_1	:	Libérer mémoire VPR
	D_1	:	Confirmez : libérer mémoire VPR0
	O_2	:	Négatif VPR
	D_2	:	Confirmez : libérer mémoire VPR
	O_3	:	Ok
	D_3	:	Ok

Initialement : la mémoire à court terme est vide. Supposons que O_1 soit le premier ordre donné par l'opérateur, la mémoire à long terme est donc vide elle aussi :

mémoire à court terme	**mémoire à long terme**
mémoire vide	*mémoire vide*

O_1 : le niveau reconnaissance propose deux hypothèses présentées dans l'ordre croissant des scores :

Hypothèse 1.1 ..... libérer mémoire VPR0

Hypothèse 1.2 ..... libérer mémoire VPR

L'hypothèse 1.1 est examinée en premier, elle se retrouve dans la mémoire à court terme, le champ *statut* prend la valeur AUCUN (ni rejet, ni négation) :

mémoire à court terme		**mémoire à long terme**
libérer mémoire VPR0	AUCUN	*mémoire vide*

Supposons que l'ordre soit cohérent, le système demande la confirmation D1 et l'hypothèse 1.2 n'est donc pas examinée.

O_2 : L'opérateur corrige l'erreur. Supposons qu'il n'y ait alors qu'une seule hypothèse délivrée par le niveau reconnaissance :

Hypothèse 2.1 ..... négatif VPR

Elle se retrouve dans la mémoire à court terme. Le champ *statut* de la phrase précédente est NIEPAR, c'est à dire que l'opérateur a nié le premier paramètre :

mémoire à court terme		**mémoire à long terme**
négatif VPR	AUCUN	*mémoire vide*
libérer mémoire VPR0	P1 nié	

O_3 : l'opérateur confirme, la mémoire à court terme est vidée, l'ordre est rangé dans la mémoire à long terme :

mémoire à court terme	**mémoire à long terme**	
mémoire vide	libérer mémoire VPR	EXECUTE

4.4.3.5 La recherche de valeur

Pour répondre à la demande de valeur, MEMOIRE dispose de quatre sources de connaissance :

- la mémoire à court terme,
- la mémoire à long terme,
- les valeurs par défaut,
- le module de reconnaissance.

Il les explore dans l'ordre de présentation. Il consulte tout d'abord la mémoire à court terme. Si la recherche échoue, il parcourt la mémoire à long terme. S'il n'a toujours pas obtenu de valeur, il regarde s'il existe une valeur par défaut. Enfin en dernier recours, il fait appel au module de reconnaissance.

a) la recherche dans les mémoires

La recherche dans les mémoires se fait à l'aide des champs *éléments* et *paramètre* de la requête "demande de valeur" formulée par MOTEUR :

éléments liste des éléments déjà réunis,
paramètre ordre du paramètre recherché.

Pour l'application SONAR, la profondeur de recherche est différente suivant que DIALOGUE explore l'une ou l'autre des mémoires. La mémoire à court terme est parcourue dans sa totalité, tous les éléments sont significatifs en fonction de la valeur de *statut*. Le recherche est restreinte au dernier ordre pour la mémoire à long terme. Dans un premier temps, elle n'était pas limitée. Mais pendant la phase de test du système, il arrivait que le système propose des ordres comportant des éléments trop anciens pour être valides. De plus l'opérateur n'avait plus forcement à l'esprit la phrase qui avait permis à la machine de proposer un tel élément, ce qui provoquait un certain désagrément. Nous avons donc réduit la profondeur de la recherche à 1, c'est-à-dire au dernier ordre reconnu, qu'il soit cohérent ou non. Pour l'application DIVA, nous n'avons pas, à l'heure actuelle, limité la profondeur par manque d'étude suffisante.

Lorsque MEMOIRE trouve un élément correspondant à la demande, il renvoie la "réponse de valeur" et affecte les champs mémoire, origine et élément.

b) la valeur par défaut

La valeur par défaut est déterminée à partir de la requête "demande de valeur" formulée par MOTEUR. Le champ *mémoires* de la requête indique l'existence ou non d'une valeur par défaut. Cette valeur est conservée par MOTEUR, le rôle de MEMOIRE est de lui indiquer s'il doit l'utiliser ou non. Le champ *mémoire* de "réponse de valeur" est dans le cas de l'existence d'une valeur affecté à VALEUR_PAR_DEFAUT.

c) l'appel au module de reconnaissance

L'appel au module de reconnaissance se fait en utilisant les champs *axiomes* et *paramètre* de la "demande de valeur" lorsqu'il s'agit de l'appel initial. Lorsqu'il s'agit d'une "demande de poursuite", il n'y a pas à relancer le module de reconnaissance pour une nouvelle acquisition.

Une des caractéristiques de DIAPASON est de ne pas proposer deux fois le même ordre lors de la mise au point. Ainsi le système ne pourra boucler indéfiniment sur la même erreur de reconnaissance. Cela est réalisé à l'aide du champ *statut* : MEMOIRE ne propose plus une commande ou un paramètre qui a été nié par le locuteur ou refusé par le système pour cause d'incohérence.

EXEMPLE 4.7 : cet exemple reprend le dialogue présenté dans l'EXEMPLE 4.5. :

EXEMPLE 4.7	O_1	:	Libérer mémoire VPR
	D_1	:	Confirmez : libérer mémoire VPR0
	O_2	:	Négatif VPR
	D_2	:	Confirmez : libérer mémoire VPR
	O_3	:	Ok
	D_3	:	Ok

La différence se situe au niveau de O_2, on suppose à ce moment que le système de reconnaissance commet la même erreur qu'en O_1, il propose deux hypothèses :

Hypothèse 2.1 ..... négatif VPR0

Hypothèse 2.2 ..... négatif VPR

L'hypothèse 2.1 est examinée en premier, l'élément VPR0 est refusé à cause du *statut* de VPR0 dans la phrase précédente. Il regarde donc l'hypothèse 2.2, l'ordre "Libérer mémoire VPR" est cohérent, il est proposé à l'opérateur.

4.4.3.6 Incohérence et mémoire

Lorsque nous avons traité l'incohérence, nous n'avions pas encore indiqué que les valeurs pouvaient avoir deux origines : la mémoire à court terme ou la mémoire à long terme. Le message envoyé à l'opérateur était jusqu'à présent, un constat d'échec : MEMOIRE proposait des valeurs erronées et on n'y pouvait rien. Pour remédier à cet inconvénient en cas d'incohérence, un traitement supplémentaire est effectué qui permet de revenir sur un choix. Le principe est le suivant : si une valeur provient de la mémoire à long terme et provoque une incohérence, c'est qu'il ne fallait pas l'envisager. Il faut poser une question au lieu d'envoyer un message d'erreur.

EXEMPLE 4.8	O	:	Afficher bruiteur pointé
	D	:	Afficher bruiteur pointé
sans			*# exécution de afficher bruiteur pointé #*
traitement	O	:	Afficher
	D	:	Le bruiteur pointé est déjà affiché
	O	:	Afficher bruiteur pointé
	D	:	Afficher bruiteur pointé
avec			*# exécution de afficher bruiteur pointé #*
traitement	O	:	Afficher
	D	:	Afficher quel bruiteur ?

4.4.3.7 Le traitement des phrases de gestion du dialogue

Le système DIAPASON dispose de quatre phrases de gestion du dialogue, deux pour l'annulation, une pour la confirmation et une pour la répétition.

La première version de DIAPASON, pour l'application SONAR, traitait ces phrases au niveau du module de gestion du dialogue pour l'annulation et la confirmation (la répétition n'était pas gérée). Cela donnait un statut particulier à ces ordres, la cohérence étant gérée au niveau de MOTEUR, DIAPASON ne pouvait remettre en cause ces phrases en cas d'incohérence et explorer ainsi les éventuelles hypothèses parallèles.

La seconde version de DIAPASON, pour l'application DIVA, traite ces ordres de manière identique aux ordres de l'application. Cela resoud le problème rencontré pour la gestion de l'incohérence. On remarquera que pour cette application, tous les

ordres sont à confirmation inexistante et que l'ordre OK est considéré comme un ordre de l'application, ce qui nécessitait de toute façon, cette révision dans le fonctionnement de DIAPASON.

4.4 LE MODULE DE GESTION DE LA TACHE

Le module de gestion de la tâche TACHE assure deux fonctions. D'une part il gère les informations spécifiques à l'application à l'aide d'un réseau syntaxico-sémantique (syntaxe et contexte de la tâche). D'autre part il assure la gestion de la communication avec l'application.

4.4.1 Le réseau syntaxico-sémantique

Le réseau syntaxico-sémantique conserve les informations syntaxiques, sémantiques et liées à la tâche qui sont spécifiques à l'application. Ses nœuds et ses arcs représentent l'ensemble des structures syntaxico-sémantiques valides. Les nœuds correspondent aux éléments de l'ordre, on leur associe un identifiant identique au code de l'élément. Il existe deux types d'arcs : les arcs marquant les alternatives sur les nœuds/éléments et les arcs marquant les suivants sur les nœuds/éléments.

Lorsque MOTEUR voudra accéder aux informations, il activera un chemin dans le réseau en lui fournissant les éléments qu'il a déjà réunis. En retour le réseau affectera une structure qui contiendra les informations demandées et qui servira à affecter les champs des requêtes faites à MEMOIRE.

La [FIGURE 5.9] présente un extrait du réseau de l'application SONAR et son instanciation avec l'ordre "graphique annexe fréquence bruiteur alpha 1" (voir ANNEXE A à la page A16 pour la définition de l'ordre). Les nœuds sont représentés par des ovoïdes (leur identifiant est indiqué à coté), les arcs alternatives sur les éléments par des flèches verticales et les arcs suivants par les flèches horizontales. A l'intérieur des nœuds instanciés on trouve les mots prononcés par l'opérateur.

Les principales actions effectuées par le réseau et les informations qu'il contient sont les suivantes :

- il indique si l'ordre est complet ou non (MOTEUR a réuni tous les éléments qui composent l'ordre),
- il teste la cohérence partielle des éléments lorsque l'ordre n'est pas encore complet (il consulte le contexte de la tâche),
- il teste la cohérence globale de l'ordre lorsqu'il est complet (il consulte le contexte de la tâche),
- il indique si l'élément courant est compatible avec les éléments déjà réunis (chemin valide entre les nœuds),
- il donne la valeur par défaut de l'élément suivant en fonction des éléments déjà réunis,
- il détermine les séquences de codes à exécuter lorsque l'ordre est confirmé, corrigé, annulé ou simulé. Une séquence de codes est un suite d'actions élémentaires à effectuer sur l'application,

- il exécute la séquence de codes qui lui est donnée (il modifie ainsi le contexte de la tâche).

Le réseau sémantique est implémenté de manière procédurale, chaque opération sur le réseau sémantique correspond à un appel de fonction. Ces fonctions sont en partie créées par les outils de l'environnement DIAPASON.

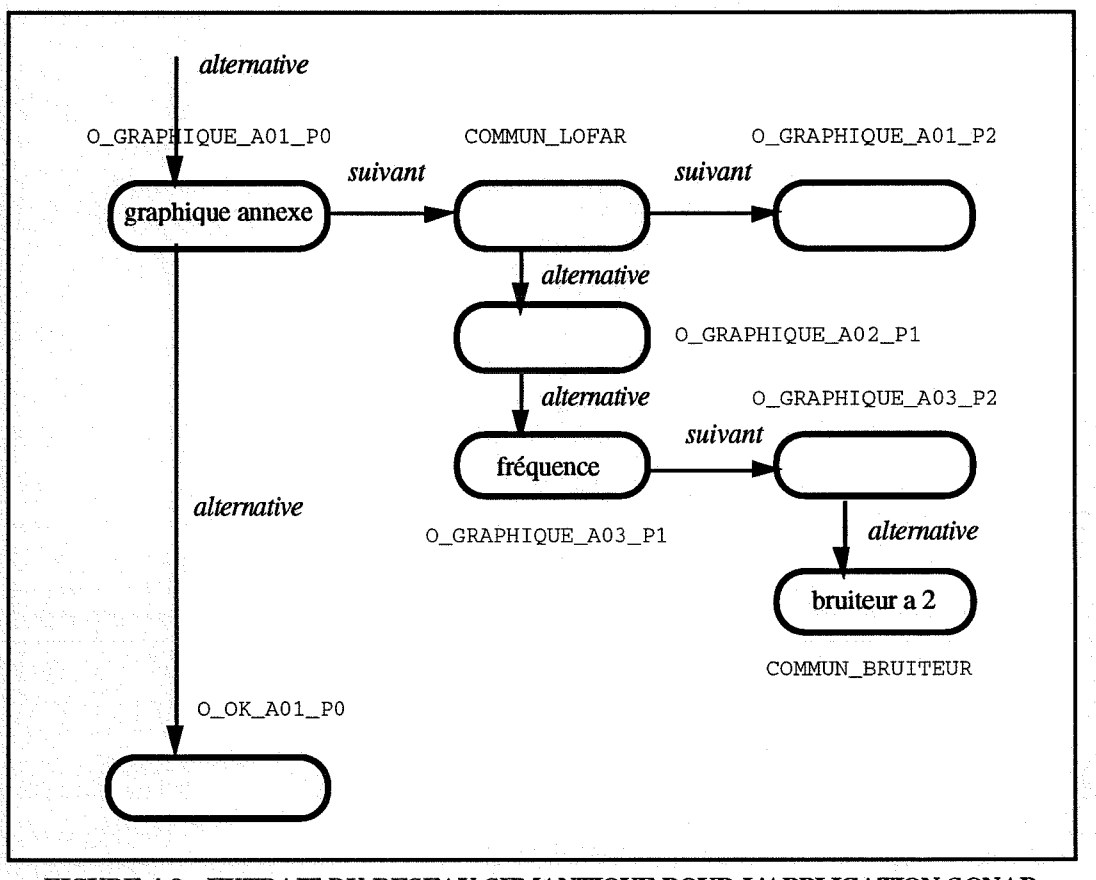


FIGURE 4.9 : EXTRAIT DU RESEAU SEMANTIQUE POUR L'APPLICATION SONAR

4.4.2 Le sous-module INTERFACE

Le sous-module INTERFACE [FIGURE 4.10] assure la communication entre le système de dialogue et l'application dans le but de maintenir un contexte de la tâche identique.

Un ordre vocal est traduit en séquences d'actions élémentaires sur l'application.

Diapason_Envoi_Application les envoie les unes à la suite des autres.

Application_Reception_Diapason reçoit une séquence qui provoque l'appel de procédures de l'application. Ces procédures modifient les affichages de l'application et lorsqu'elles modifient en plus le contexte géré par l'application, elles sont traduites en séquences.

Application_Envoi_Diapason les envoie une à une à DIAPASON.

Diapason_Envoi_Application reçoit une séquence qui provoque l'appel de fonctions qui vont modifier le contexte de l'application géré par DIAPASON.

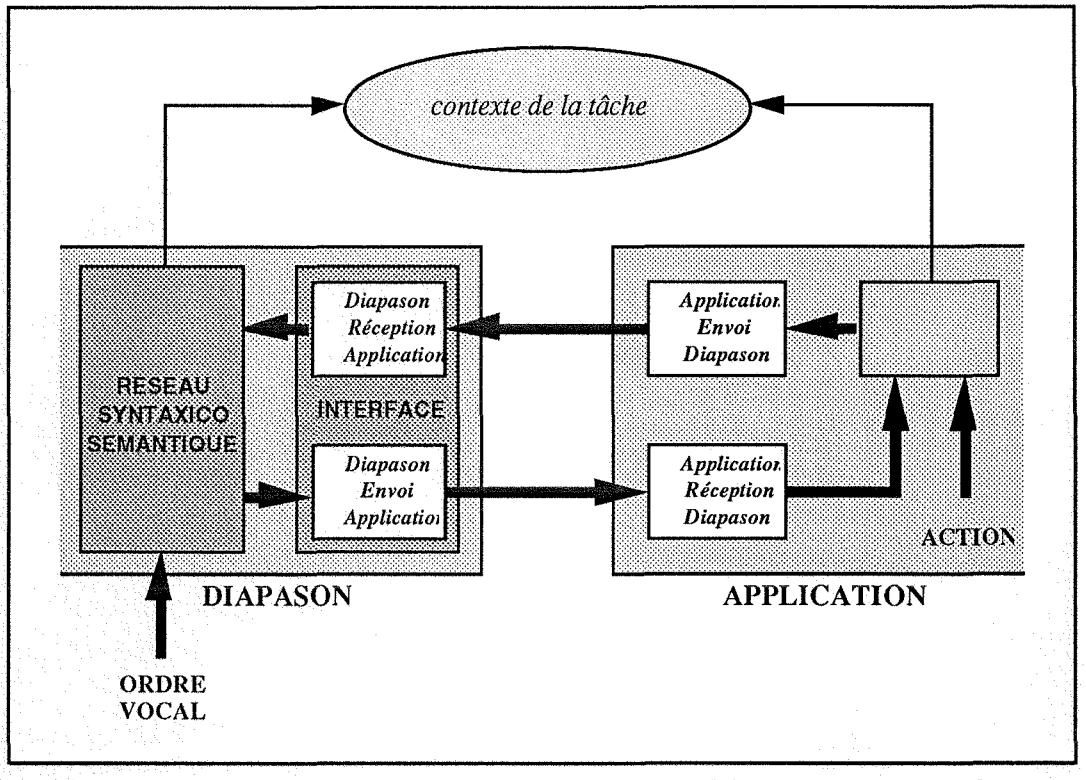


FIGURE 4.10 : LA COMMUNICATION ENTRE DIAPASON ET L'APPLICATION

Une action de l'opérateur sur l'application ou un événement déclenché par celle-ci (disponibilité de résultats par exemple) et qui modifie le contexte de l'application, est traduite en séquences.
Application_Envoi_Diapason les envoie une à une à DIAPASON.
Diapason_Envoi_Application recoit une séquence qui provoque l'appel de fonctions qui vont modifier le contexte de l'application géré par DIAPASON et traduire ces séquences en phrases qui vont être ajoutées aux mémoires du système de dialogue.

4.5 LE MODULE DE GESTION DE LA RECONNAISSANCE

Le module de gestion de la reconnaissance est chargé de gérer les entrées écrites et orales du système de dialogue DIAPASON. Il communique en interne avec le module de gestion du dialogue. Il peut échanger des informations avec des modules extérieurs tel que la maquette de reconnaissance de la parole développée par THOMSON SINTRA DASM. Il travaille à partir des réprésentations sémantiques des phrases reconnues.

4.5.1 La structure interne de RECONN

Le module RECONN est composé de plusieurs sous-modules. Il y a le sous-module principal chargé du contrôle général et des sous-modules dédiés chacun à une entrée.

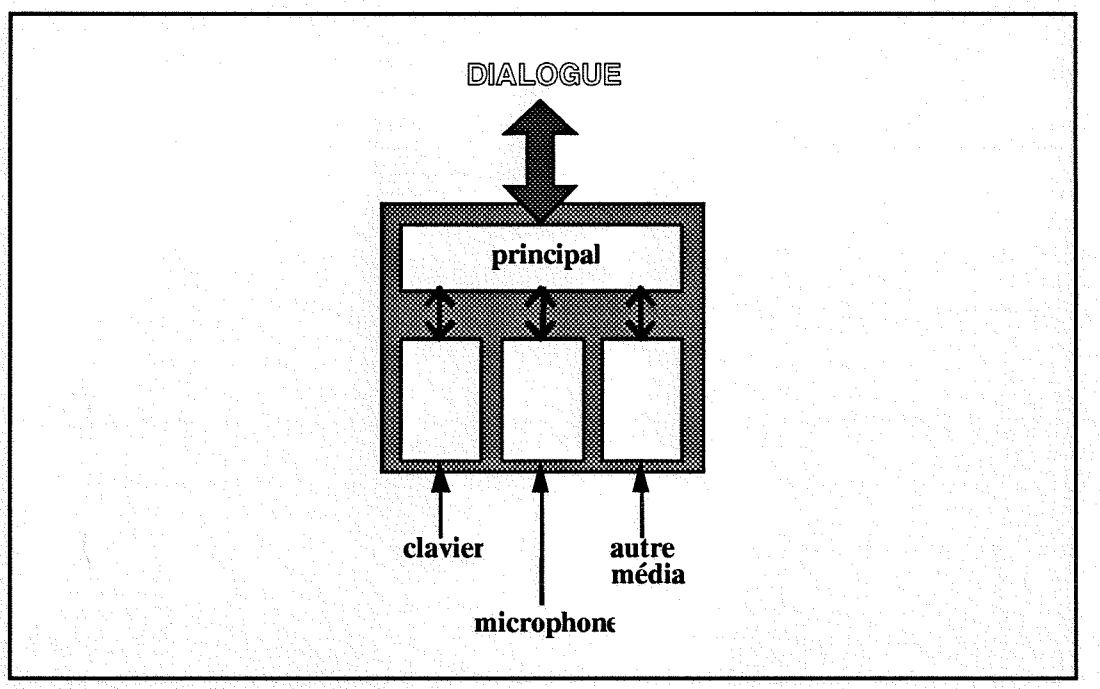
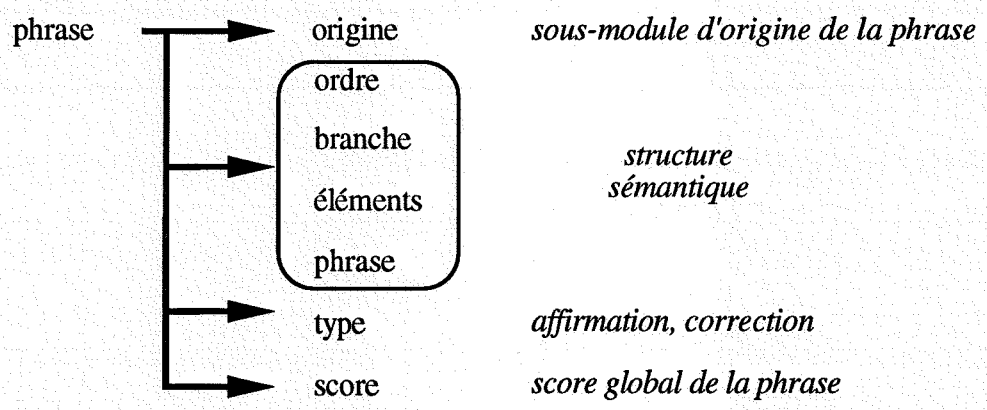


FIGURE 4.11 : LE MODULE RECONN

4.5.2 La représentation sémantique de la phrase

Quel que soit le média dont se sert l'utilisateur final du système de dialogue, le sous-module dédié à cette entrée fournit au sous-module principal une représentation sémantique de la phrase reconnue. Elle contient la structure sémantique présentée § 4.2 et des informations complémentaires sur la phrase.



EXEMPLE 4.9 :
Réprésentation sémantique de la phrase "graphique annexe LOFAR 1 gamme CD"

```
origine :          MICROPHONE
ordre :            20
branche :          1
elements[0] ->     code            O_GRAPHIQUE_A01_P0
                   nombre_mots     2
                   mot[0]          graphique
                   mot[1]          annexe
elements[1] ->     code            O_GRAPHIQUE_A01_P1
                   nombre_mots     2
                   mot[0]          lofar
                   mot[1]          un
elements[2] ->     code            O_GRAPHIQUE_A01_P2
                   nombre_mots     2
                   mot[0]          gamme
                   mot[1]          cd
phrase :           "graphique annexe LOFAR 2 gamme CD"
type :             AFFIRMATION
score :            322
```

4.5.3 Le sous-module principal

Le sous-module principal est chargé de gérer les sous-modules dédiés à une entrée et de transmettre à DIALOGUE les structures sémantiques des phrases reconnues.

4.5.3.1 la gestion des sous-modules

La gestion d'un sous-module consiste à transmettre les paramètres qui doivent guider la reconnaissance de phrase. Ils permettent de choisir la grammaire à utiliser pour faire l'analyse syntaxique de la phrase prononcée au microphone, tapée au clavier ou exprimée à l'aide d'un autre média.

Quelle que soit l'application à diriger, RECONN conserve trois paramètres :

- le code du dernier ordre reconnu,
- le numéro de branche du dernier ordre reconnu,
- si une question est posée, le numéro de paramètre sur lequel porte la question.

Selon l'application, il peut y avoir des paramètres supplémentaires, par exemple pour l'application SONAR, il y a un code qui indique l'image courante, pour l'application DIVA, il y a un code qui indique le mode (exploitation ou configuration).

A partir de ces paramètres RECONN déduit les informations qu'il doit transmettre à chaque sous-module.

4.5.3.2 les échanges avec DIALOGUE

RECONN repond aux requêtes formulées par DIALOGUE "demande de valeur" et "poursuite d'analyse". S'il trouve une valeur ou s'il existe une autre hypothèse pour la phrase, il répond VRAI et transmet la représentation sémantique de la phrase, sinon il répond FAUX.

4.5.3.3 la résolution des ambiguïtés

Le langage utilisé pour les ordres est de type artificiel, à un ordre complet ne correspond qu'une seule interprétation sémantique. Il ne peut donc y avoir ambiguïté. Cependant des phrases incomplètes, dans le cas d'ellisions, peuvent avoir des interprétations différentes. Dans ce cas la levée de l'ambiguïté est possible en tenant compte du contexte de la tâche et des paramètres de reconnaissance. Ce traitement est effectué au niveau de RECONN, les structures syntaxico-sémantiques ambiguës concérnées subissent un traitement qui supprime l'ambiguïté.

4.5.4 L'entrée clavier

L'entrée clavier permet à l'opérateur de donner un ordre à l'application via le système de dialogue, en utilisant le clavier. Ce mode de fonctionnement est plus un mode de test pour la mise au point du système pour une application. Dans un premier temps nous pensions qu'il pourrait suppléer aux éventuelles défaillances du système de reconnaissance vocal (bruit ambiant trop important, rejet d'un locuteur), mais dans la pratique l'opérateur peut toujours utiliser l'entrée classique de l'application par menus.

Deux types d'implémentations de ce module ont été réalisés.
La première est la reprise directe des travaux de [RICHARD 1987] [ALINAT 1987] sur lequel nous avons ajouté une couche dialogue. Il reprend les principes développés pour MYRTILLE 1 [PIERREL 1978]. Il s'agit d'un système guidé par la syntaxe qui à chaque étape, émet des hypothèses sur les mots à reconnaître et les valide par un sous-module de reconnaissance de mots. Une telle implémentation ne s'avère pas forcément utile dans le cadre de phrases tapées au clavier où il n'existe aucune incertitude sur les terminaux. La description du langage est simple, mais son implémentation nécessite de définir un ensemble d'objets et une procédure de conversion pour l'interface avec le sous-module principal.
La seconde implémentation a été réalisée à l'aide du générateur d'analyseur syntaxique UNIX "YACC" [JOHNSON 1978]. La description du langage pour chaque application demande plus d'attention que dans l'implémentation précédente mais la structure sémantique est directement réalisée.

Ces deux implémentations sont très contraignantes du point de vue syntaxique, elles n'autorisent aucune faute de frappe, en effet notre but est de faire fonctionner DIAPASON en mode vocal. Néanmoins la première implémentation permettrait d'introduire une certaine incertitude sur les terminaux. Des études ont été réalisées au CRIN-CNRS & INRIA-LORRAINE sur des analyseurs écrits qui tolèrent des erreurs sur les terminaux [MORIN 1987] [GAIFFE 1991].

Une troisième implémentation est en cours, elle est toujours syntaxiquement contraignante, mais elle sera compatible avec les outils de l'environnement

DIAPASON. Il restera au concepteur du système de dialogue à saisir les structures syntaxico-sémantiques, utiliser le compilateur et le formateur pour obtenir une description des phrases compréhensibles directement par l'analyseur syntaxique écrit et par le système de compréhension.

4.5.5 L'entrée vocale

Le sous-module de gestion de l'entrée vocale est une interface entre le système de dialogue DIAPASON et la maquette de reconnaissance développée par THOMSON SINTRA DASM [FIGURE 4.12]. Dans le sens DIAPASON vers maquette de reconnaissance, le sous-module envoie les paramètres de reconnaissance présentés plus haut, dans le sens inverse la maquette de reconnaissance envoie la liste des hypothèses pour une phrase prononcée. Les hypothèses sont décrites à l'aide d'un format spécifique et le sous-module est chargé de les convertir en structures sémantiques compréhensibles par le sous module principal.

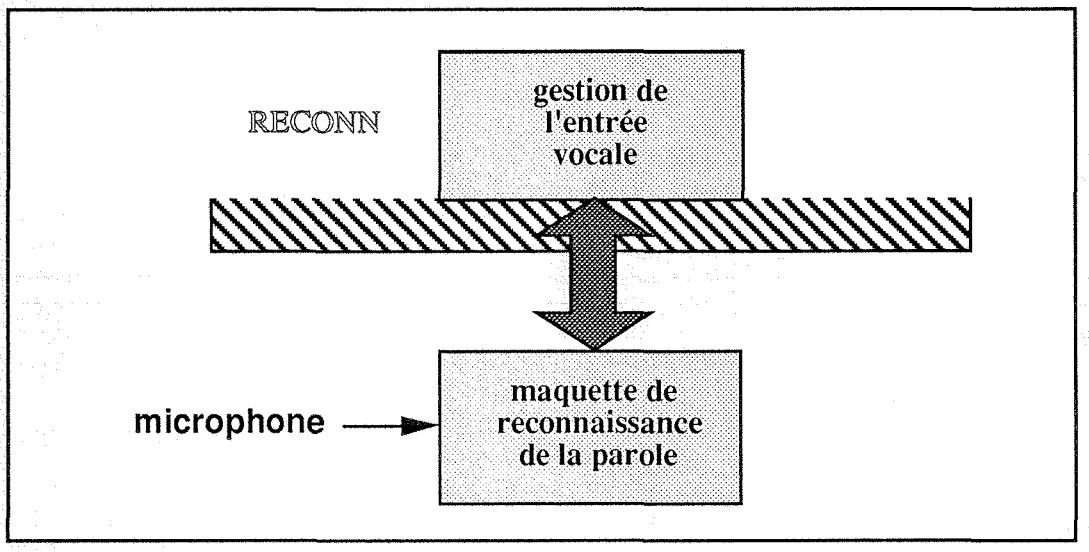


FIGURE 4.12 : INTERFACE ENTRE DIAPASON ET LE SYSTEME DE RECONNAISSANCE DE LA PAROLE

Exemple de phrase au format de l'application SONAR

```
*RES*  NBHYP=1
*RES*  HYP    nor=  315  tot=  170  GLO=(25, 3)   FP= 39  NBMOTs=5
*RES*  MOT    nor=  57   tot=  54   LOC=(C, )   AFFICHER
*RES*  MOT    nor=  45   tot=  38   LOC=(P, )   PISTE
*RES*  MOT    nor=  40   tot=  16   LOC=(p,)    RAIE
*RES*  MOT    nor=  14   tot=  16   LOC=(d, )   BRUITEUR
*RES*  MOT    nor=  44   tot=  47   LOC=(D, )   POINTE
```

Le message *RES* en première colonne indique qu'il s'agit d'un message destiné au système de dialogue. On distingue trois types de lignes.
La ligne NBHYP indique le nombre total d'hypothèses.
La ligne HYP correspond à une description générale de l'hypothèse avec :

`nor`	le score de l'hypothèse après normalisation ∈ [0 , 512]
`tot`	le score brut de l'hypothèse ∈ [0 , 512]
`GLO`	le code global indique les paramètres numéro d'ordre et numéro de branche
`FP`	une note qui estime la fin de parole
`NBMOTs`	le nombre de mots de l'hypothèse

La ligne MOT présente un mot de l'hypothèse :

`nor`	le score du mot après normalisation, il tient compte de la longueur du mot ∈ [0 , 64]
`tot`	le score brut du mot ∈ [0 , 64]
`LOC`	le code local est une codification du numéro de paramètre, la lettre majuscule est mise sur un mot pertinent
`MOT`	mot reconnu

Exemple de phrase au format de l'application DIVA

Le second format reprend uniquement les informations les plus pertinentes pour le système de dialogue et de manière plus condensée afin de réduire la longueur des échanges. L'habillage des lignes a disparu, les lignes commencent par le caractère @ identifiant la maquette de reconnaissance de la parole, seule la note normée est envoyée, les codes locaux donnent directement le numéro de paramètre et sont placés avant le score. Si la même phrase existait pour l'application DIVA cela donnerait :

```
@1
@25 3    315 39   5
@0  57   AFFICHER
@1  45   PISTE
@1  40   RAIE
@2  14   BRUITEUR
@2  44   POINTE
```

4.5.6 Autres entrées

Il est envisageable d'introduire une autre entrée dans le système de dialogue. Il pourrait s'agir d'une entrée souris par exemple, dans le cas où nous serions amenés à gérer celle-ci au niveau de RECONN et non plus au niveau de l'application comme c'est le cas actuellement. On peut aussi envisager l'utilisation d'un gant de désignation, une autre maquette de reconnaissance de la parole... Pour introduire ce nouveau media il faudrait définir les échanges qu'il y a à effectuer entre le sous-module principal et le nouveau media. Il faudrait aussi définir les structures syntaxiques propres à ce media. On peut noter qu'à ce niveau une telle étude est en cours au CRIN pour la désignation (souris ou gant) [BELLALEM 1991].

Chapitre 5

LES APPLICATIONS DEVELOPPEES

5.1 L'APPLICATION SONAR

5.1.1 Le contexte de l'étude

En vue d'étudier l'intégration d'une entrée vocale aux autres moyens d'interface homme-machine d'une console et de pouvoir se rendre compte des avantages et des inconvénients de ce mode d'entrée, nous nous sommes volontairement placés dans le cas réaliste d'une console d'un matériel existant, en l'occurrence un sonar. L'introduction d'une entrée vocale sur une console nécessite des modifications très importantes du système d'interaction avec l'opérateur et il était donc hors de question de modifier directement le matériel. Compte tenu de cela, nous avons préféré simuler, sur un PC connecté au système de reconnaissane, les images et commandes de la console sonar réelle. En disposant le sonar réel et le PC côte à côte dans le même local, il a été possible de faire des comparaisons entre l'interaction avec commande vocale sur la simulation et celle existant sur le sonar réel, pour les mêmes sous-tâches élémentaires.

Dans cette expérience la restitution orale de la phrase reconnue n'est pas possible. En effet l'opérateur est déjà à l'écoute du signal qu'il analyse, une synthèse vocale des réponses du système serait venue parasiter son écoute. L'interprétation de la phrase reconnue est donc affichée dans une zone réservée de l'écran.

5.1.2 Le choix des ordres donnés en vocal

Les commandes du sonar réel (sans entrée vocale) sont essentiellement du type menu hiérarchique avec une profondeur maximum de 4. L'opérateur dispose pour cela d'un clavier logiciel et de deux commutateurs. Il dispose également d'une boule pour déplacer un curseur. Sur le sonar choisi, il y a deux images principales : l'image "Initialisation" et l'image "Veille".

L'image "Initialisation" permet la gestion du contenu des mémoires de visualisation et la gestion des paramètres de base du sonar.

L'image "Veille" est composée d'une image (azimut, temps) sur laquelle sont représentées les pistes, c'est-à-dire les détections successives dans le temps d'un

même bruit, avec plus ou moins de détails et d'une image annexe grâce à laquelle l'opérateur peut avoir des renseignements complémentaires sur les pistes. Sur cette image de Veille, l'opérateur peut régler les échelles, décider des informations à visualiser, pointer des traitements particuliers dans des directions déterminées, régler les paramètres de certains traitements et envoyer des renseignements à l'extérieur.

L'ensemble des ordres est décrit en ANNEXE A à l'aide de leur représentation syntaxico-sémantique.

La suite de ce paragraphe présentera l'émulation des commandes sonar que nous avons réalisée. En effet un des premiers travaux réalisés fut la mise au point d'une simulation réaliste de la console sonar.

5.1.2.1 Les écrans

L'écran est porteur d'informations relatives à la tâche ou au système de compréhension. Elles étaient disposées dans un premier temps en des endroits différents de l'écran. Nous verrons par la suite que nous avons été amenés à introduire une contre-réaction visuelle du système de compréhension dans la zone réservée à la tâche.

a) la structure générale de l'écran

On distingue quatre zones dans un écran [FIGURE 5.1] :

- la **zone titre** contient des indications sur le programme : version et origine ;

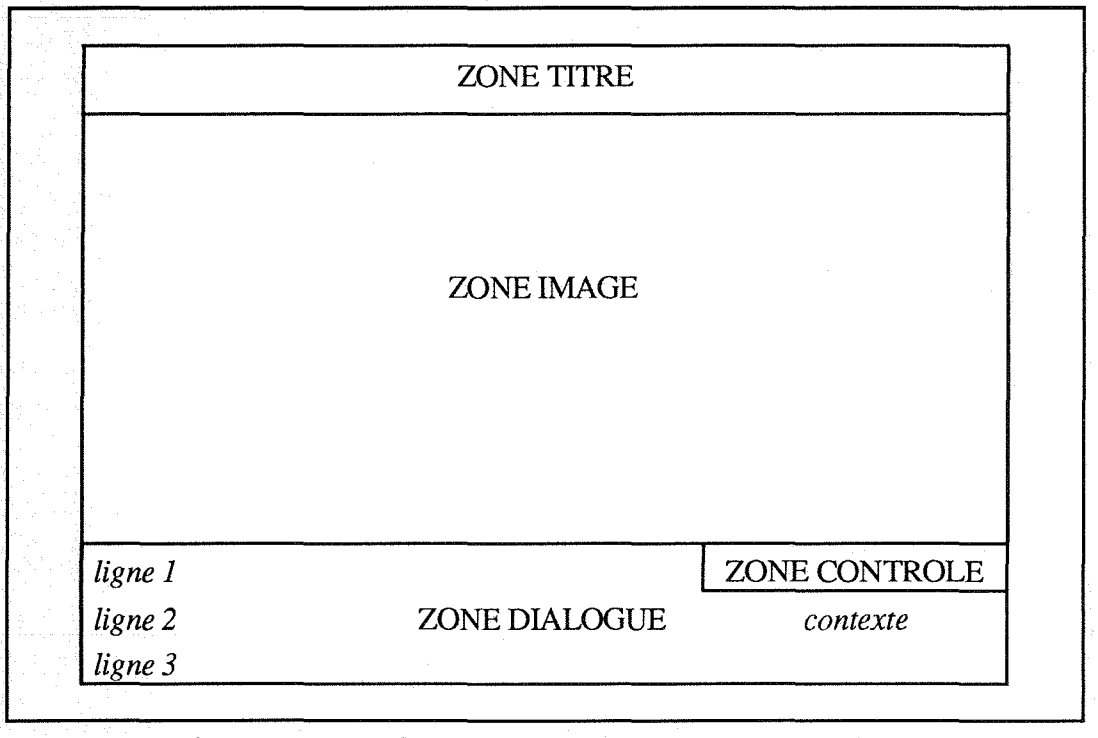


FIGURE 5.1 : LES DIFFERENTES ZONES DE L'ECRAN

- la **zone image** concerne tous les éléments spécifiques à une image SONAR. Le module APPLICATION effectue sa mise à jour lors de l'exécution d'un ordre. La souris permet de désigner des objets dans cette zone,
- la **zone dialogue** présente les informations relatives au dialogue. La partie droite de la zone contient le contexte du dialogue en affichant la dernière commande reconnue. Elle permet en un coup d'œil si nécessaire et si l'opérateur a une pratique suffisante du système, de connaître le contexte d'enchaînement des ordres. La partie gauche de la zone comporte trois lignes où s'affichent les phrases issues du système de reconnaissance et celles déduites par la compréhension. Lorsqu'il y a cohérence d'un ordre, il y a sur la première ligne la phrase provenant de RECONN et sur la deuxième son interprétation contextuelle issue de DIALOGUE. En cas d'incohérence il y a sur la première l'interprétation et sur la deuxième le message d'incohérence. La troisième ligne est utilisée pour indiquer que le système est en attente d'une entrée ou est en cours de traitement. Le module SYNTAXE y envoie l'écho de la phrase tapée au clavier. La FIGURE 5.2 présente le contenu des deux premières lignes en fonction des phrases prononcées et du contexte du dialogue. A l'utilisation nous avons remarqué qu'il était gênant pour l'opérateur d'être obligé de porter le regard sur cette zone pour vérifier que l'ordre a été correctement reconnu et interprété. L'idéal serait qu'il n'ait pas à quitter des yeux l'endroit où se passe l'action,

O : Libérer mémoire VPR
D : Confirmez : libérer mémoire VPR 0
ligne 1 : Libérer mémoire VPR
ligne 2 : Confirmez : libérer mémoire VPR 0

O : Négatif VPR
D : Confirmez : libérer mémoire VPR
ligne 1 : Négatif VPR
ligne 2 : Confirmez : libérer mémoire VPR

O : Ok
D : Ok
ligne 1 : Libérer mémoire VPR
ligne 2 : Ok

O : LOFAR 3
D : La fonction LOFAR 3 n'est pas présente dans la MIV
ligne 1 : Libérer mémoire LOFAR 3
ligne 2 : La fonction LOFAR 3 n'est pas présente dans la MIV

FIGURE 5.2 : LE CONTENU DE LA ZONE DIALOGUE

- la **zone contrôle** modifie le comportement de DIAPASON lorsque l'on clique avec la souris une des trois cases qui la composent. Elles permettent successivement l'accès au module AIDE, l'affichage d'informations internes au système, l'arrêt de DIAPASON. Lorque le module AIDE a la main, elles permettent l'accès aux trois types de syntaxe (standard, correction, enchaînement), la sortie du

module, l'arrêt de DIAPASON. En ce qui concerne les informations internes, elles sont plus utiles pour la mise au point du système, mais permettent de présenter numériquement des informations affichées analogiquement à l'écran (position d'un bruiteur, d'un marqueur, contenu du SAT...).

b) l'image initialisation

L'image INIT est l'image qui s'affiche lors de la mise en route du système [FIGURE 5.3] et [FIGURE 5.4]. On distingue trois parties dans la zone image :

- la **zone nom image** où s'affiche le nom de l'image,
- la **zone MIV** présente la répartition des différentes fonctions dans la MIV (sur la ligne i , il y a la liste des images accessibles depuis la console secondaire i),
- la **zone paramètres** présente la valeur de chaque paramètre SONAR.

c) l'image veille

L'image VEILLE se décompose en six zones [FIGURE 5.5] et [FIGURE 5.6] :

- la **zone nom image** où s'affiche le nom de l'image,
- la **zone graphique annexe** où est affiché le graphique annexe demandé avec son nom à la gauche du dessin. Afin de ne pas augmenter inutilement la taille d'APPLICATION quatre types de graphiques ont été programmés : large bande, histogramme, un seul pour la fréquence et un commun à tous les LOFAR,
- la **zone marqueurs** comporte quatre lignes correspondant successivement de haut en bas aux marqueurs AUDIO, LOFAR, DEMON et LOUPE. Le symbole du marqueur est affiché à la verticale de l'azimut correspondant (A pour AUDIO, L pour LOFAR, D pour DEMON et V pour LOUPE),
- la **zone échelle azimut** présente les azimuts de 0 à 360 ou de -180 à +180. Elle sert de repère pour les marqueurs, les bruiteurs et le SAT (centre de commande chargé de centraliser les informations provenant de différentes sources : autres sonars, vigie, position des navires amis...),
- la **zone bruiteurs** est divisées en cinq parties :
  - la ***zone échelle temps*** dans laquelle apparaissent une echelle de temps et l'heure courante. Selon la cadence d'affichage (quatre ou seize secondes), elle présente huit ou trente deux minutes d'enregistrement des positions des bruiteurs (sonar réél respectivement 1/4h ou 1h). L'instant présent est situé en haut,
  - la ***zone sat*** dans laquelle apparaissent les contacts connus du SAT,
  - la ***zone identification*** où est affiché le nom d'identification du bruiteur. Ils est positionné selon le lever de doute (le contact vient de la gauche ou de la droite de la flûte),
  - la ***zone baptême*** où est affiché le nom de baptême du bruiteur. Il est positionné de la même façon que le nom d'identification,
  - la ***zone trajectoire*** où apparaissent les positions des bruiteurs et le cap du navire. Le trait du bruiteur est continu si le contact est affiché, pointillé s'il est effacé ou si le coté ne correspond pas au lever de doute. Les raies forment un nuage de points de part et d'autre du bruiteur, leur nombre est limité à un maximum de quatre. Par soucis de simplification, cette image n'évolue pas dans le temps.

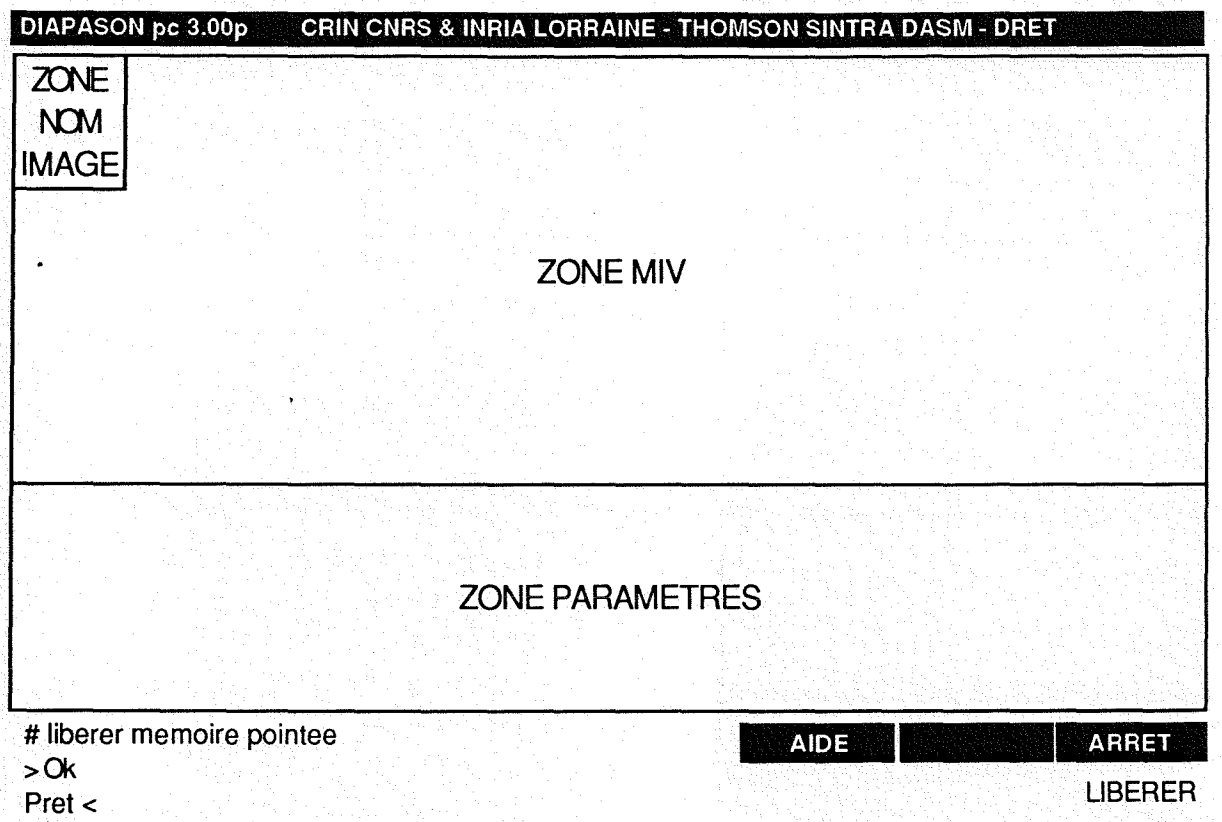


FIGURE 5.3A : LES ZONES DE L'IMAGE INIT

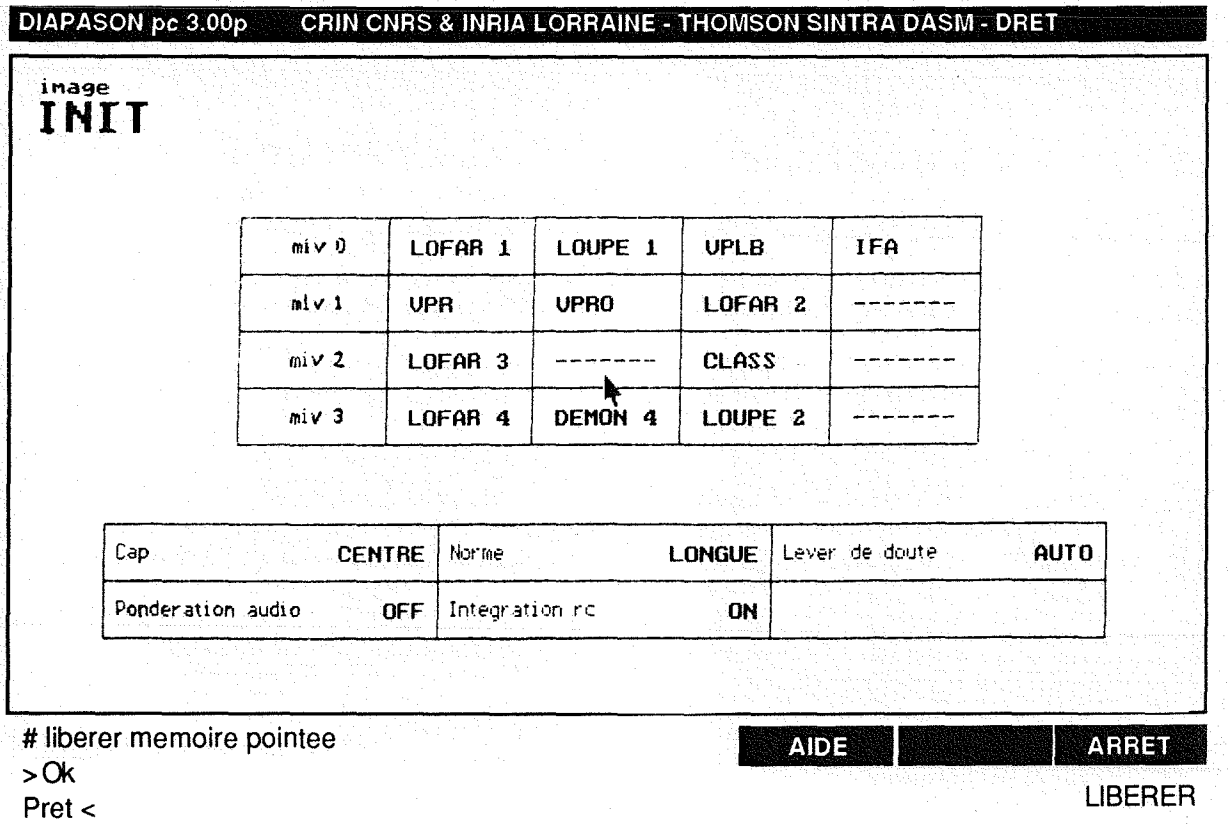


FIGURE 5.3B: L'IMAGE INIT

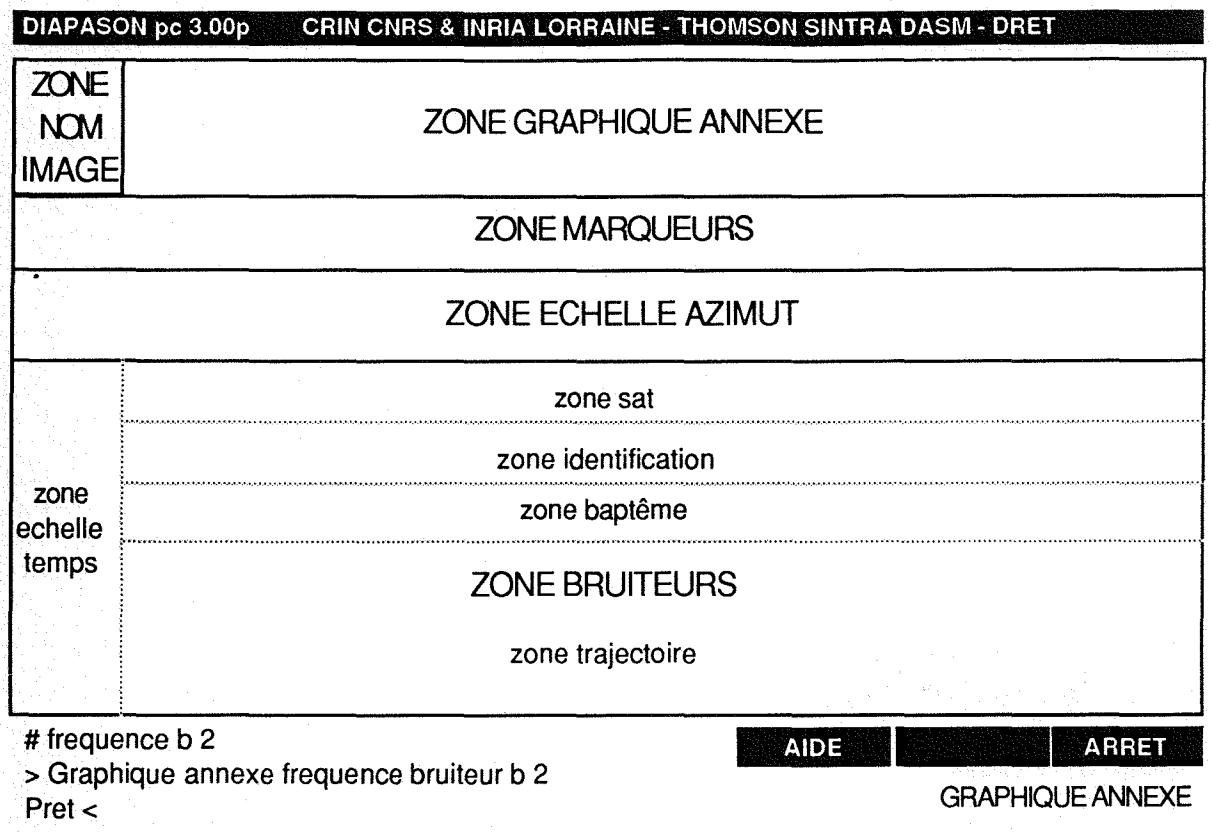


FIGURE 5.4A : LES ZONES DE L'IMAGE VPR

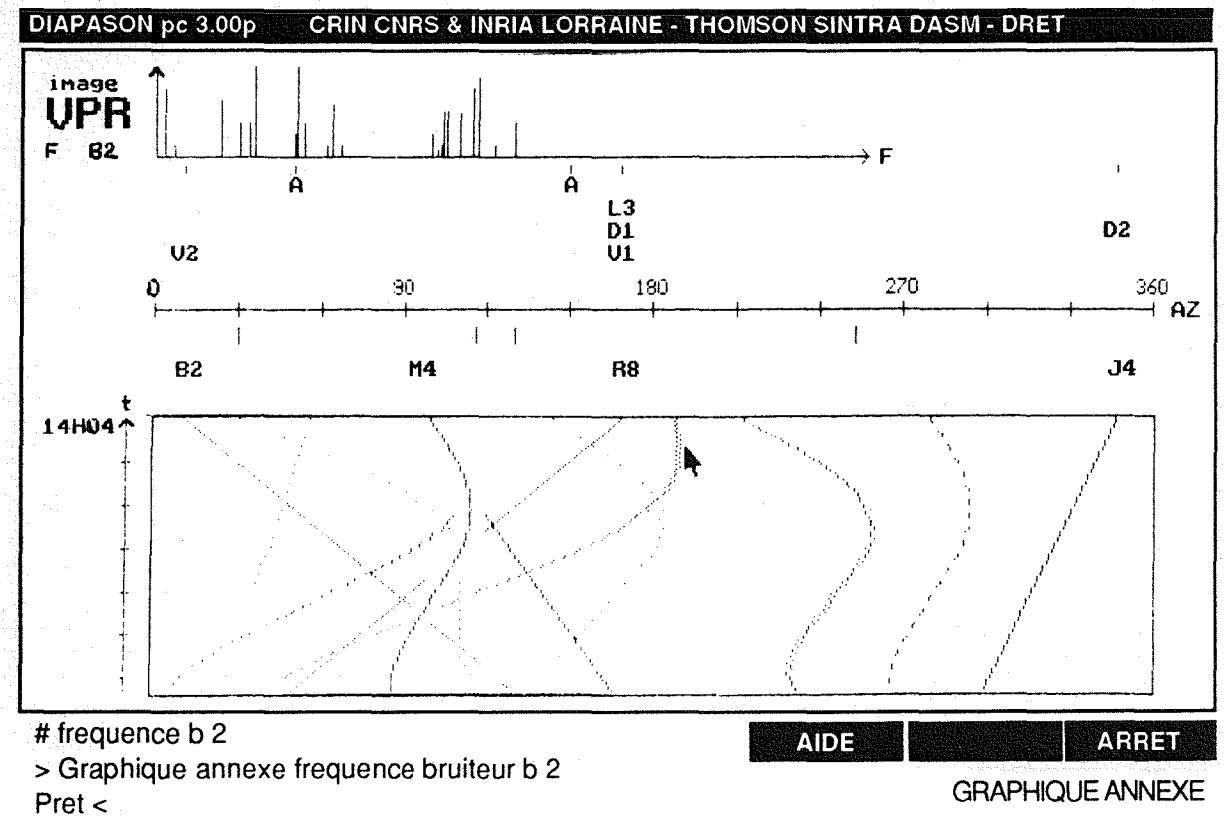


FIGURE 5.4B : L'IMAGE VPR

d) les autres images sonar

Les autres images sonar sont réduites à la seule **zone nom image**. Le fait de simuler ces images n'apportait rien de plus au système :

- du point de vue de l'application, il suffit d'ajouter un dessin de LOFAR, de LOUPE, etc, selon l'image.
- du point de vue des ordres, le vocabulaire n'augmente pas, les ordres à donner pour les images semblent être un sous-ensemble des ordres de l'image VEILLE.

e) les écrans d'AIDE

Le module AIDE présente la syntaxe des ordres que l'opérateur peut donner au système pour l'image courante. Après un menu, l'ordre choisi est affiché sous forme d'un graphe, dans un cadre posé sur la zone image. A titre d'exemple, nous développerons ci-dessous, l'aide mise en place pour l'ordre de libération d'une mémoire : ordre LIBERER.

Le premier écran d'AIDE qui apparaît est un menu général qui présente tous les ordres possibles depuis l'image courante. La première ligne du menu présente la syntaxe de la commande en cours, la deuxième l'ordre pour le changement d'image et les lignes suivantes les ordres spécifiques à l'image courante. Les cases de la zone contrôle jouent alors le rôle suivant :

- la première permet de choisir le type de syntaxe que l'on veut étudier : STANDARD pour l'ordre complet, ENCHAINE pour les enchaînements (ellipse de la commande et ellipse d'un paramètre pour une autre commande) et CORRIGE pour la correction. Le choix est possible quel que soit l'écran AIDE, il suffit de cliquer la case pour obtenir l'affichage suivant la permutation circulaire : STANDARD -> ENCHAINE -> CORRIGE -> STANDARD,
- la deuxième permet de quitter le mode AIDE depuis l'écran menu, de revenir à ce menu depuis les autres écrans,
- la troisième permet d'arrêter le programme.

Le type de confirmation associé à l'ordre est indiqué en bas à gauche du cadre de l'aide.

Les écrans auquels l'opérateur peut accéder pour l'ordre de libération d'une mémoire sont présentés FIGURE 5.7 à FIGURE 5.11. Ils utilisent les conventions suivantes :

- un mot en majuscule est obligatoire,
- un mot en minuscule est facultatif,
- un mot entouré de < et > est un terminal lexical, en cliquant dessus on obtient sa définition.

Les cinq figures qui suivent présentent :

- FIGURE 5.7, tous les ordres valables en image INIT, image dont dépend l'ordre LIBERER,
- FIGURE 5.8, la syntaxe complète de l'ordre LIBERER,
- FIGURE 5.9, la syntaxe du terminal-lexical <FONCTION>,
- FIGURE 5.10, les enchaînements possibles après l'ordre LIBERER,
- FIGURE 5.11, les syntaxes de correction de l'ordre LIBERER.

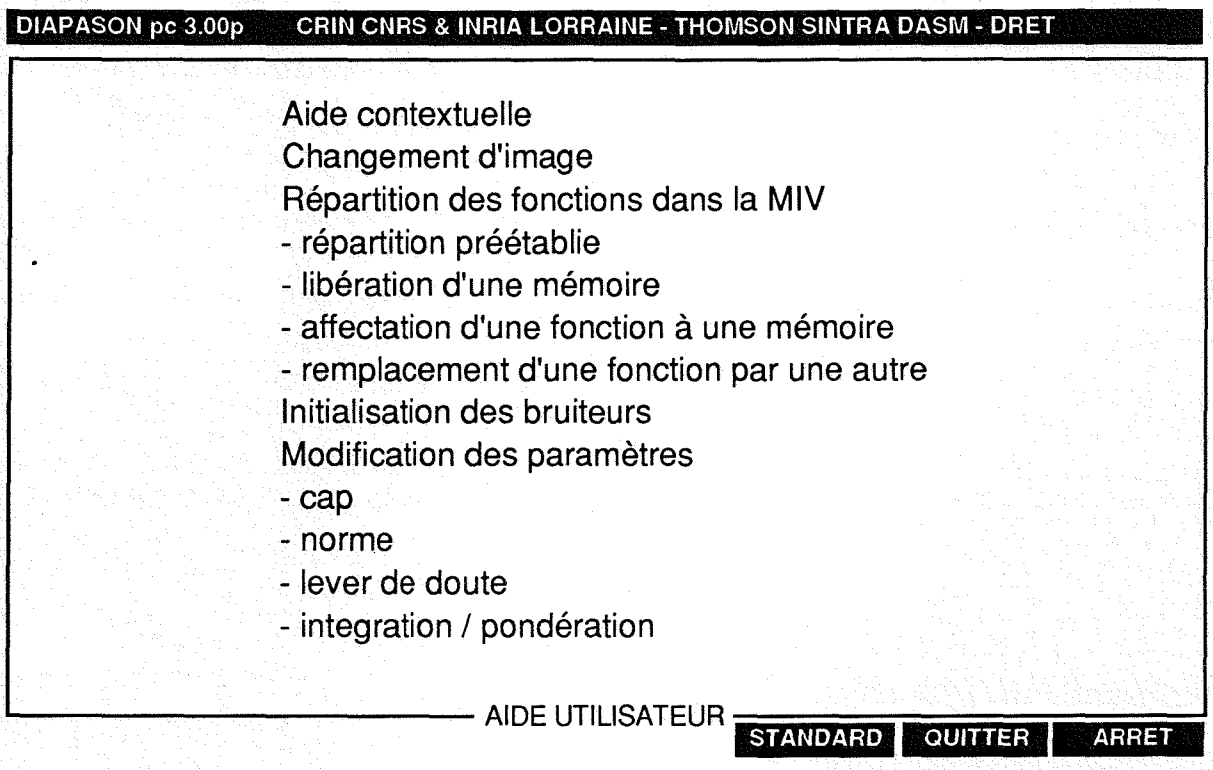


FIGURE 5.5A : AIDE - ECRAN MENU image INIT

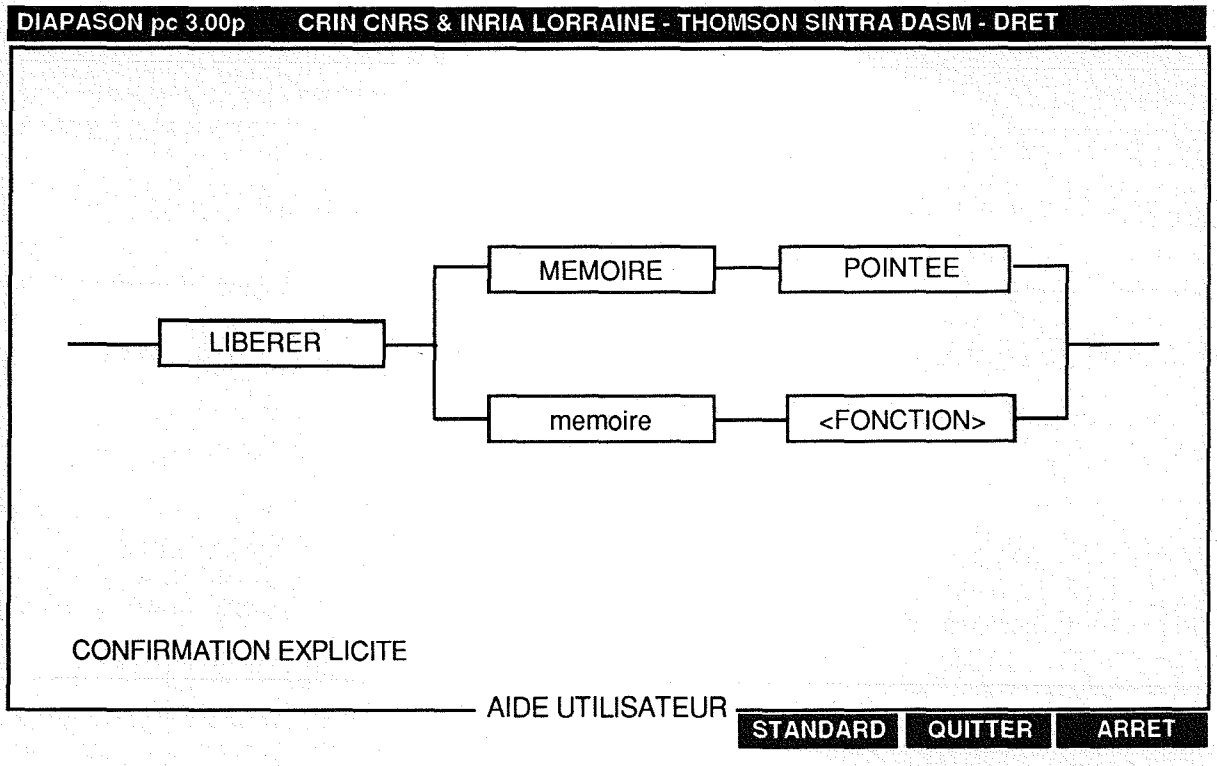


FIGURE 5.5B : AIDE - SYNTAXE COMPLETE ordre LIBERER

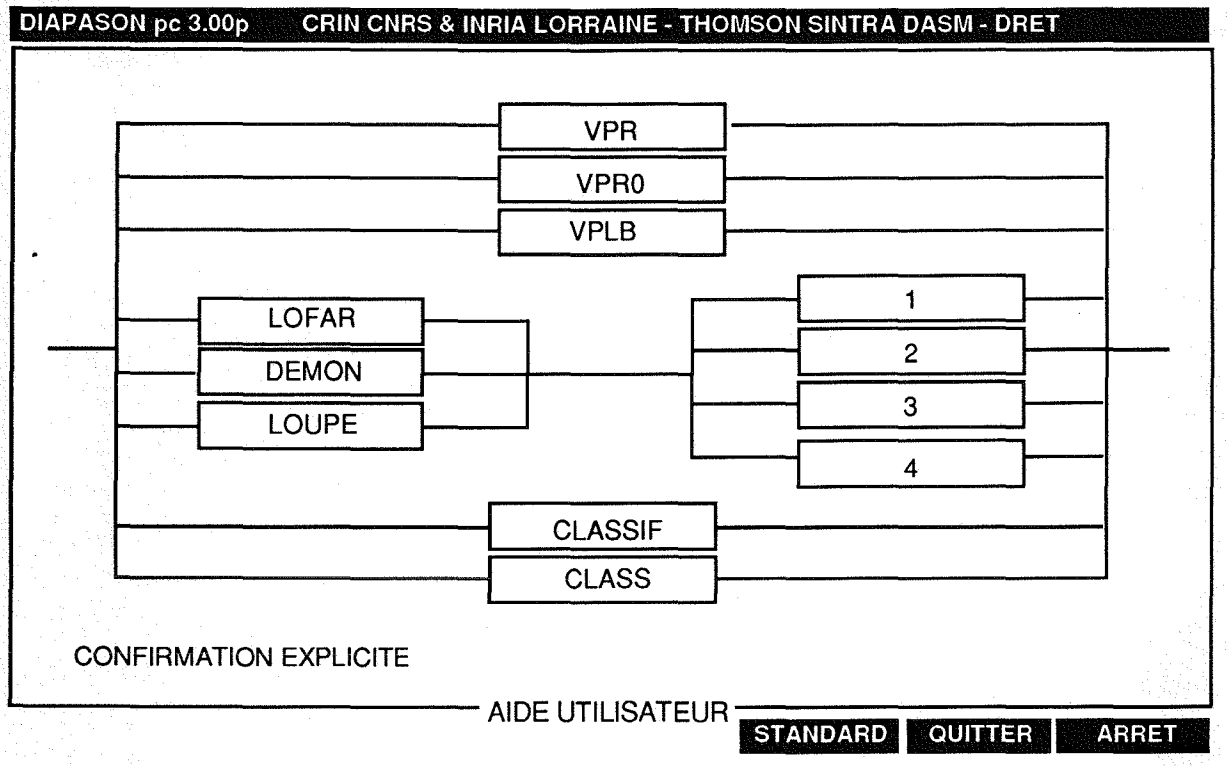


FIGURE 5.5C : AIDE - SYNTAXE DE <FONCTION> ordre LIBERER

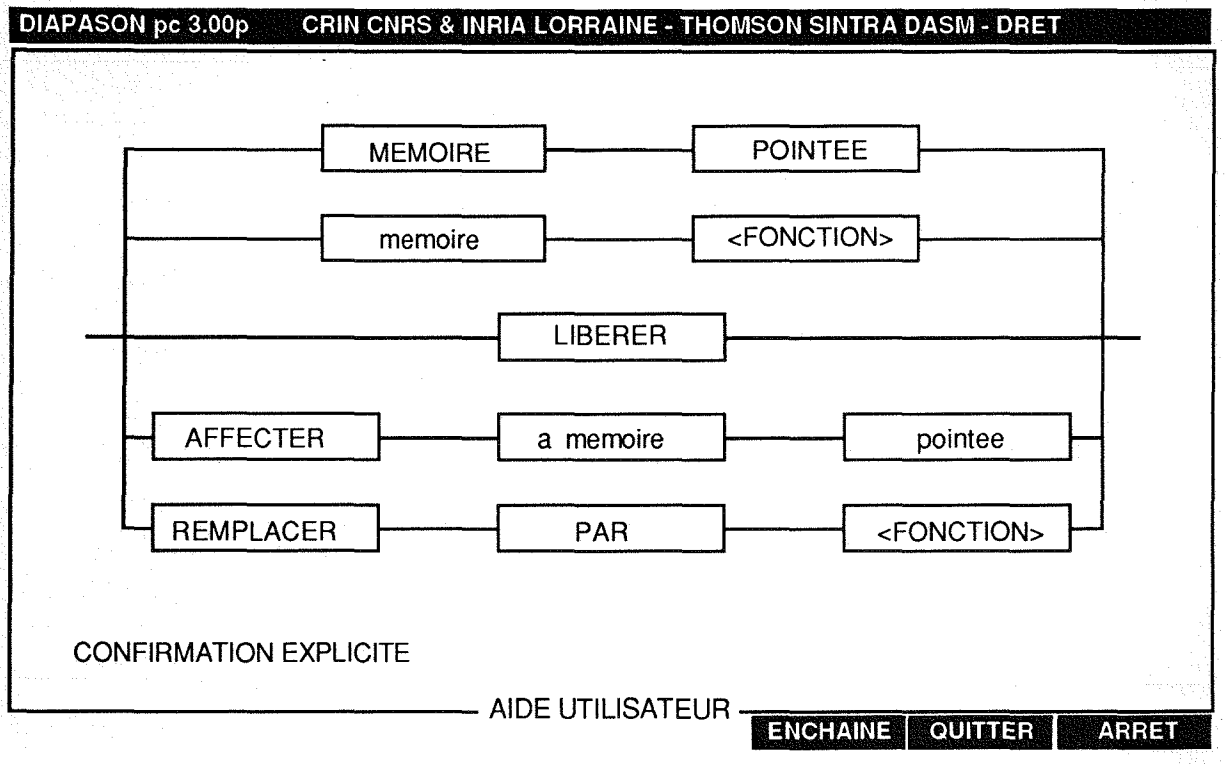


FIGURE 5.5D : SYNTAXE DES ENCHAINEMENTS ordre LIBERER

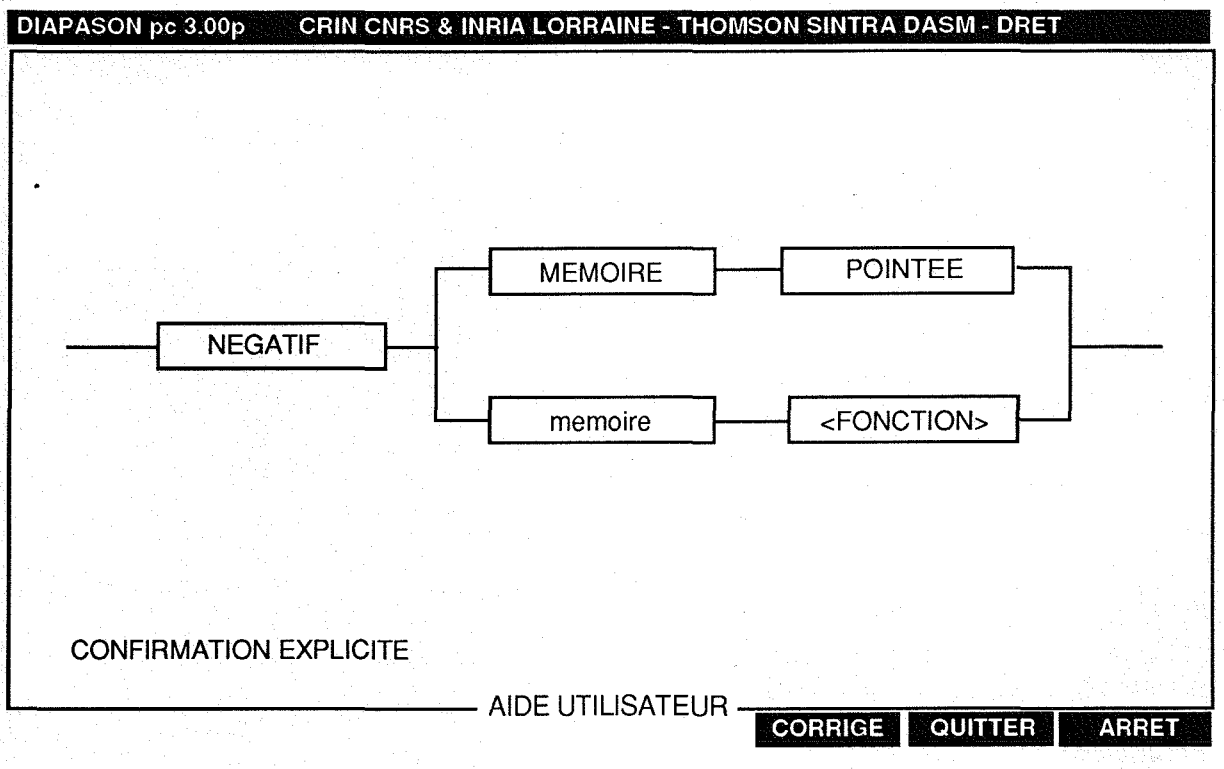


FIGURE 5.11 : AIDE - SYNTAXE DES CORRECTIONS ordre LIBERER

5.1.2.2 Les pointages

Il est possible de désigner des objets à l'aide de la souris. Pour chaque type d'objet l'opérateur ne peut en désigner qu'un seul. Les différents types d'objets sont les suivants :

- les **cases de la MIV**, l'opérateur peut en selectionner une dans la partie IFA, et une dans l'autre partie. Un clic dans une case la marque et annule le choix précédent,
- les **bruiteurs**, l'opérateur selectionne un contact en désignant son nom d'identification, son nom de baptême, ou un point de la trajectoire proche de l'instant présent,
- un **contact SAT**, l'opérateur sélectionne un contact en désignant le trait vertical qui représente le bruiteur,
- un **azimut**, l'opérateur marque un azimut en désignant un point de l'échelle azimut. Un trait vertical apparaît à l'endroit, l'azimut pointé précédent est effacé.

La désignation d'objets joue un rôle important dans les performances opératoires du système. Elle apporte un plus grand naturel aux commandes, en permettant de travailler sur un objet sans avoir à le nommer explicitement. Elle permet au système de reconnaissance d'obtenir de meilleurs résultats, par exemple pour faire référence à un objet on utilise le mot "pointé" qui est plus facile à reconnaître qu'une suite de chiffre ou un nombre :

"azimut pointé"	à la place de	"azimut 245"
"bruiteur pointé"	à la place de	"bruiteur sierra six".

5.1.2.3 Les contrôles

Il existe deux instants où DIAPASON effectue des contrôles : le premier se situe au moment de la vérification de la cohérence, le second à l'exécution de l'ordre.

a) les contrôles de cohérence

Les contrôles de cohérence ont une grande importance dans le fonctionnement de DIAPASON, comme nous l'avons montré dans le chapitre précédent. Ce sont ces contrôles qui vont permettre de rejeter des hypothèses incorrectes fournies par le niveau reconnaissance.

Les principes généraux mis en œuvre au cours de la validation sont les suivants :

- lorsque l'ordre reconnu provoque un changement de valeur, DIAPASON vérifie que la nouvelle valeur est différente de la précédente.
  exemple :
  si la phrase reconnue est "Norme courte", DIAPASON vérifie que la valeur courante de la norme n'est pas courte,
  autre exemple :
  si la phrase reconnue est "Afficher bruiteur ALPHA 1", DIAPASON vérifie que le bruiteur ALPHA 1 est effacé.
- lorsque l'opérateur fait référence à un objet pointé, DIAPASON vérifie qu'il y a effectivement un objet de ce type qui est désigné.
  exemple :
  si la phrase reconnue est "Libérer mémoire pointée", il vérifie qu'il existe bien une mémoire pointée.
- lorsque l'opérateur fait référence à un objet dans un énoncé, DIAPASON vérifie que cet objet est bien défini.
  exemple :
  si la phrase reconnue est "Afficher bruiteur ALPHA 1", DIAPASON vérifie que le contact ALPHA 1 est connu du sonar.

La liste des contrôles effectués par DIAPASON est donnée en ANNEXE A avec la représentation syntaxico-sémantique de l'ordre.

b) les contrôles à l'exécution

Au moment de l'exécution d'un ordre, il reste un dernier contrôle à effectuer, vérifier que les objets pointés n'ont pas changé depuis l'instant où le contrôle de cohérence a été effectué. Cela peut arriver plus facilement dans le cas d'un ordre à confirmation explicite où l'intervalle de temps qui sépare les deux instants n'est plus négligeable. Dans le cas où cela se produit, un message d'erreur est envoyé à l'opérateur et l'ordre n'est pas exécuté.

EXEMPLE 5.1

 # désignation d'une case #
O : Libérer mémoire pointée
 # contrôle de cohérence # 'pas de problème
D : Confirmez : libérer mémoire pointée
 # désignation d'une nouvelle case #
O : Ok
 # contrôle à l'exécution # 'problème
D : Attention la mémoire pointée a changé !

5.1.3 Evaluation du système

Dès le début de l'étude il a été prévu de présenter le système à des personnes connaissant le SONAR afin d'avoir une idée réelle de l'apport du mode oral dans une application de type commande de machine. Nous avons alors réalisé une évaluation ergonomique. Les tests ont été effectués sur la plate-forme d'intégration du sonar chez THOMSON. Nous en avons tiré un certain nombre de résultats relatifs à l'emploi et aux avantages de la commande vocale. Ils nous ont permis dans un premier temps d'entreprendre des améliorations et devraient par la suite permettre de réaliser un nouveau système.

5.1.3.1 la complexité du langage

Afin de pouvoir évaluer les performances du système de reconnaissance, il faut connaître quelques caractéristiques du langage qu'il reconnait. Le vocabulaire/syntaxe de l'application SONAR est donné en ANNEXE A.

Le premier critère pour évaluer cette complexité est la **taille du vocabulaire**. Le nombre de mots du lexique est 122.

Le second critère à retenir est le **facteur de branchement**.

Le facteur de branchement initial, c'est-à-dire le nombre de mots pouvant se trouver en début de phrase, varie suivant le contexte du dialogue :

- en image initiale : 13 à 30,
- en image veille : 17 à 27.

Le facteur de branchement maximum à l'intérieur d'une commande varie :

- en image initiale : 2 à 9 en moyenne 5.8,
- en image veille : 1 à 32 en moyenne 19.

Le facteur de branchement dans les grammaires de correction, c'est-à-dire les phrases introduites par le mot négatif, est plus important encore car il est la somme des facteurs de branchement de chacun des constituants de la phrase à corriger et peut atteindre :

- en image initiale : 12,
- en image veille : 18.

Le **nombre de mots** par phrases est très variable, au minimum il est de 1, au maximum il est de 7 (ordre diriger branches 2 et 4). Le nombre moyen de mots

par phrase est difficile à calculer étant donné que le facteur de branchement initial important et qu'il est fonction du contexte du dialogue, mais il doit être proche de 3.

5.1.3.2 les conditions de l'expérimentation

La comparaison entre le dialogue de commande du SONAR réel et celui de la maquette PC a eu lieu sur la plate-forme d'intégration du SONAR dans le centre THOMSON-SINTRA d'ARCUEIL (91).

Les deux systèmes étaient côte à côte. Le SONAR fonctionnait en simulation, les tests n'ont donc pu être effectués en environnement opérationnel sur une tâche opérationnelle. Nous avons alors étudié des **sous-tâches simulées** que nous présentons dans le paragraphe qui suit.

Le **bruit d'ambiance** qui régnait dans la salle était important. Il y avait d'une part six baies d'électronique et d'autre part une dizaine de personnes présentes d'où des conversations fréquentes de 0,5m à 2m du microphone omnidirectionnel pendant l'expérimentation. Nous n'avons pas pris de mesures chiffrées de ce bruit ambiant.

Les **opérateurs** qui se sont succédés étaient entraînés soit sur le SONAR soit sur la maquette. Nous ne disposions pas de suffisament de temps pour les préparer sérieusement sur les deux systèmes. Les opérateurs habitués au SONAR étaient des techniciens et ingénieurs travaillant à la mise au point des sonars et non pas utilisateurs finaux.

Le **système PC** utilisait une version de grammaires où la référence au bruiteur se faisait en désignant le bruiteur avec la souris et non pas à l'aide de son baptême lettre + chiffre. Cela aide à la reconnaissance mais ne diminue en rien le potentiel de commande du système.

5.1.3.3 les mesures ergonomiques

Etant donné le peu de temps de disponibilité du SONAR et le fait que les tests ne pouvaient être faits sur une tâche réelle, M. AMALBERTI (médecin principal au CERMA (Centre d'Etude et de Recherche de la Medecine Aéronautique)) nous a conseillé de concentrer nos efforts de comparaison sur quelques sous-tâches exécutées par quelques opérateurs en gardant bien présent à l'esprit que seules les comparaisons sur tâches réelles en ambiance réelle permettent des conclusions quasi définitives.

Trois scénarios ont été effectués par six opérateurs sur le SONAR, puis sur le PC. Pour chaque sous-tâche on a mesuré :

- la durée totale,
- le nombre d'opérations effectuées classées en différentes catégories : appui de touche, réglage de commutateur, déplacement du regard, pointage avec la boule, pointage avec la souris, phrase prononcée...

Pour des raisons de confidentialité la suite des opérations sur le sonar ne sera pas détaillée. Une présentation plus complète des scenarii est faite dans [GALLAIS 1989].

a) le premier scénario

BUT

Regrouper LOFAR 1..4 en bas à droite des mémoires présentées sur l'image INIT.

SUITE DES OPERATIONS SUR LE SONAR

Nombre d'opérations

19 appuis de touches (+ 8 appuis successifs)
8 réglages commutateur 4 positions
environ 20 déplacements du regard

Durée opérateurs très entraînés sur le SONAR : 40s à 50s
opérateurs avec entraînement pour cette séquence : 60s à 70s

SUITE DES OPERATIONS SUR PC (exemple pour l'un des opérateurs)

Répartition préétablie 2	(click)	<LOFAR 1>		
Libérer mémoire pointée	(click)	<LOFAR 2>		pointages
Libérer	(click)	<LOFAR 3>	<---	avec la
Libérer	(click)	<DEMON 4>		souris
Remplacer par LOFAR 1	(click)	<DEMON 3>		
Par LOFAR 2	(click)	<case MIV>		
Affecter LOFAR 2	(click)			

Nombre d'opérations

7 ordres et 7 clicks (on pourrait avoir un click commun)
6 pointages avec la souris (on peut supprimer les pointages)

Durée Opérateurs entraînés : 50s à 60s
avec léger entraînement : 70s à 90s

Remarque : le temps comporte environ 21s d'attente due au temps de réponse trop important du système de reconnaissance de parole utilisé, ceci sera amélioré dans la version suivante de la maquette. Entre la version support de ce test et la nouvelle version du système de reconnaissance, ce temps d'attente devrait être divisé par un facteur compris entre 2 et 4.

Le regard reste sur l'objet si la confirmation est bien faite

b) le deuxième scenario

BUT

En image VPR asservir LOFAR 3 au bruiteur 01, pointer LOFAR 1 dans l'azimut 90. Visualiser le LOFAR 3 gamme EF puis le graphe LB pour la gamme EF en graphique annexe.

SUITE DES OPERATIONS SUR LE SONAR

Nombre d'opérations

31 appuis de touches
3 réglages commutateur 4 positions
2 pointages par boules
environ 30 déplacements de regard

Durée : 85s à 120s

SUITE DES OPERATIONS SUR PC (exemple pour l'un des opérateurs)

Image VPR	(click)
Diriger LOFAR 3 vers bruiteur pointé	(Ø)
Diriger LOFAR 3 vers bruiteur pointé	(click)
LOFAR 1 vers azimut pointé	(click)
VPLB gamme EF	
Graphique annexe LOFAR 3 gamme EF	
Large bande	

Nombre d'opérations

6 ordres et 3 clicks
1 répétition (Ø)
2 pointages avec la souris

Durée : 70s à 90s
dont environ 24s d'attente de réponse

c) le troisième scenario

BUT

En image VPR mettre l'échelle en azimut -180/+180, afficher la poursuite des bruiteurs. Lever de doute sur le bruiteur 01 : choisir comme bruiteur réel le gauche. Donner le baptême 8.7.3 au bruiteur 02. Alerter le SAT.

SUITE DES OPERATIONS SUR LE SONAR

Nombre d'opérations

24 appuis de touches
2 pointages par boules
environ 24 déplacements de regard

Durée : 95s à 125s

SUITE DES OPERATIONS SUR PC (exemple pour l'un des opérateurs)

Image VPR	(click)
Changer échelle	
Afficher bruiteurs	(click)
Lever de doute bruiteur pointé gauche	(Ø)
Négatif	
Lever de doute bruiteur pointé gauche	(click)
Baptisé 873	(click)
Alerte SAT	(click)

Nombre d'opérations

6 ordres et 5 clicks
1 répétition (Ø)
1 négatif
1 pointage avec la souris

Durée : 80s à 95s
dont environ 30s d'attente de réponse

d) analyse des résultats

Les trois sous-tâches étudiées peuvent se décomposer en treize sous-sous-tâches élémentaires. On a compté pour chaque tâche le nombre d'opérations élémentaires effectuées par l'opérateur (appui de touche, click, réglage de commutateur, prononciation d'une phrase, pointage, pointage ...). Il a été possible d'obtenir les histogrammes [FIGURE 5.12]. Le premier présente les résultats pour le SONAR, le deuxième pour le PC solution par pointage des MIV et le troisième solution par nom de fonction. On peut remarquer une nette diminution du nombre d'opérations élémentaires réalisées par l'opérateur entre la version SONAR et la version PC. Si on se limite dans le cas du PC aux phrases prononcées pour chaque sous-sous-tâche élémentaire, il est en moyenne de 1,8 (minimum 1, maximum 3).

5.1.3.4 Démonstration

En plus des essais précédents sur les trois sous-tâches, une vingtaine de locuteurs non entraînés ont pu essayer plus ou moins longuement le système de commande vocale montrant ainsi les qualités du système :

- multilocuteur sans apprentissage,
- résistance aux variations de prononciation :
  + accents régionaux,
  + rapidité d'élocution,
  + silence ou liaison entre mots,
  + hésitations,
- résistance au bruit d'ambiance et aux conversations à proximité.

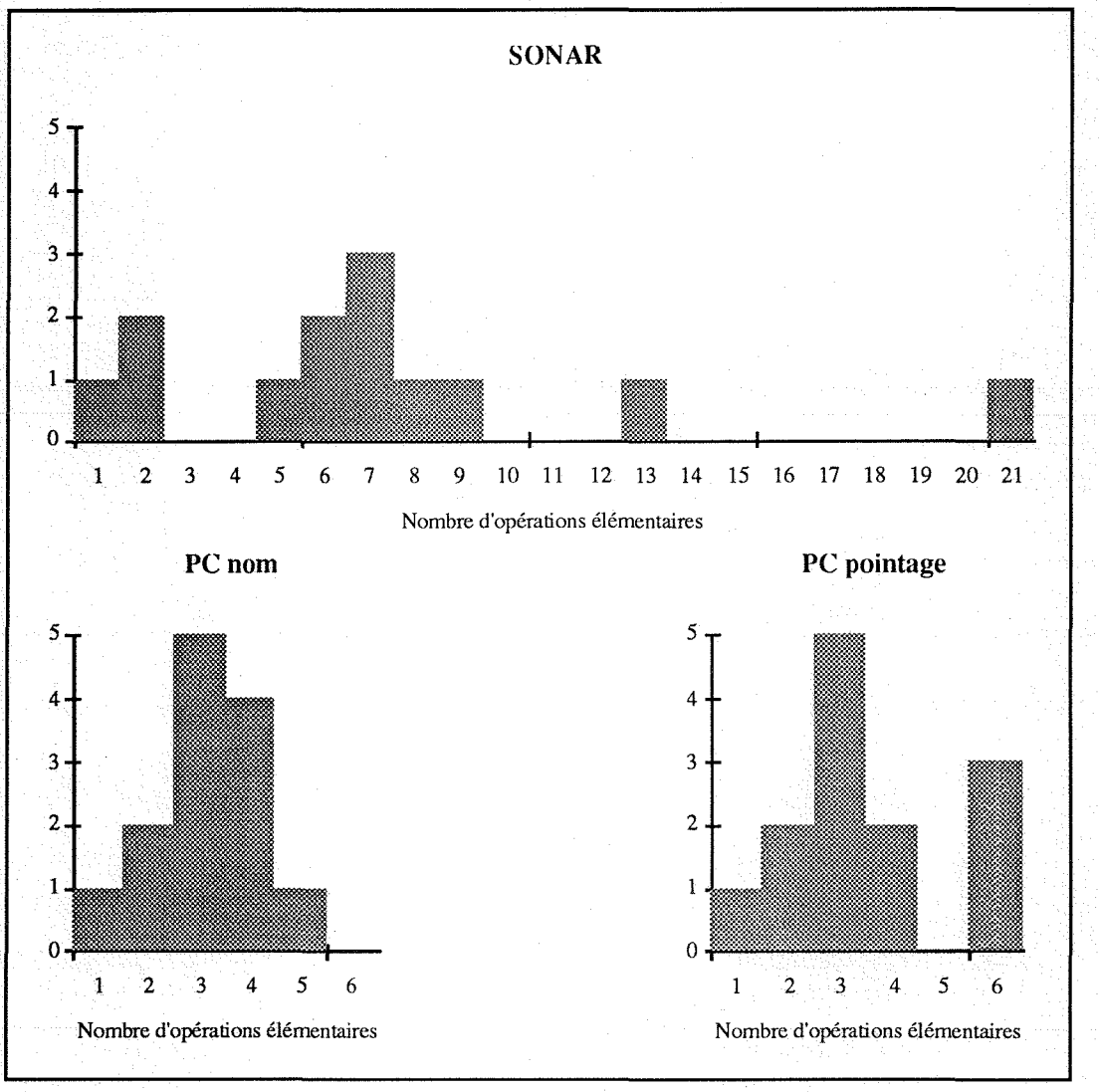


FIGURE 5.12 : HISTOGRAMMES

5.1.4 Les résultats acquis

Grâce à la comparaison entre la console du SONAR (sans commande vocale) et sa simulation sur PC (avec l'adjonction d'une commande vocale) la présente étude a permis d'acquérir les résultats suivants relatifs à l'emploi et aux avantages de la commande vocale pour les consoles.

5.1.4.1 mode d'emploi de l'entrée

Les **temps de réponse** cumulés des systèmes de reconnaissance de parole et de compréhension de l'ordre ne doivent pas dépasser **1 seconde**. Au-delà l'opérateur perd du temps à attendre.

Les **demandes de confirmation** d'ordre doivent le plus souvent possible être faites "en place" c'est-à-dire se matérialiser par une modification de l'image proche de l'endroit que l'opérateur est censé observer à cet instant. Autrement dit la zone dialogue (en haut ou en bas de l'écran) doit être utilisée le moins souvent possible pour ne pas perdre en partie un des avantages principaux de la parole qui est que le regard reste fixé sur l'objet manipulé

L'utilisation d'un commutateur (**pédale micro**) permettant à l'opérateur d'indiquer quand il donne un ordre vocal s'avère un moyen simple, pratique à utiliser (s'il est monté sur la souris ou la boule) et très efficace pour se protéger de la reception d'ordres parasites dus aux conversations se tenant près du microphone. Les opérateurs apprécient de pouvoir confirmer leurs ordres quand nécessaire en faisant un "click" sur le commutateur ci-dessus. L'emploi d'un "double-click" est encore plus sûr.

Bien évidemment les ordres doivent pouvoir être donnés sous forme de "**phrases**" (par exemple mots enchaînés) prononcées de façon naturelle c'est-à-dire continue. Sinon il faut restreindre l'emploi de la commande vocale à des ordres composés d'un seul mot.

Pour les ordres vocaux l'opérateur n'est plus guidé dans son choix par des possibilités affichées à l'écran. Dans ce cas les **moyens d'aide** deviennent très utiles, surtout pour les opérateurs débutants. En particulier une aide contextuelle devient indispensable.

Dans les systèmes à commande par mots enchaînés il faut **limiter la taille** du vocabulaire et la richesse de la syntaxe de façon que l'opérateur puisse l'apprendre. La taille maximale semble être de 150 à 200 mots, ce qui est suffisant pour les applications de type console.

5.1.4.2 aspect cognitif de l'avantage à employer une entrée vocale

Les ordres données au PC à la voix étaient nettement plus naturels que la suite d'abréviations du clavier logiciel du SONAR. En effet ils comportent en général une structure de type :

verbe - complément direct - complément circonstanciel

EXEMPLE 5.2

SONAR	"MENU" "LOFAR 1"	(EXEC)
	"FONC" "LOFAR" "AZ"	(EXEC)

soit 7 appuis de touches et un réglage de bouton

Le même ordre donné vocalement	"Diriger LOFAR 1 vers azimuth pointé"

Cet avantage est d'autant plus net que l'opérateur dispose pour ses ordres vocaux de plus de liberté : élisions, inversions, mots outils qui n'existent pas sur le SONAR sans commande vocale.

5.1.4.3 aspects ergonomiques de l'avantage à employer une entrée vocale

Le **regard de l'opérateur** peut rester fixé sur l'objet manipulé (piste, échelle, fenêtre de présentation d'informations). Autrement dit, il n'est pas nécessaire, pour un ordre vocal, de regarder le clavier, un clavier logiciel ou des menus affichés sur l'écran. En moyenne la présente étude a montré que, même avec des demandes de confirmations faisant trop appel à la zone dialogue (en bas de l'écran), l'opérateur distrait son regard 4 fois moins souvent en utilisant le système PC avec commande vocale qu'en utilisant la console du SONAR sans commande vocale. On estime qu'avec des demandes de confirmation faites de façon optimale l'emploi d'une commande vocale permettrait à l'opérateur de distraire son regard environ 15 fois moins souvent qu'avec les autres modes de commande pour les ordres concernés.

L'usage d'une commande vocale permet de **concentrer en une seule "phrase"** prononcée de nombreuses manipulations élémentaires de curseur (souris) et du clavier logiciel. Par exemple les trois sous-tâches étudiées en détails sont au total composées de treize sous-tâches élémentaires. Dans le cas de la console SONAR pour chaque sous-tâche élémentaire le nombre d'opérations (appui de touche ...) est en moyenne de 7,5 alors que dans le cas du PC le nombre de phrases prononcées est de 1,8 en moyenne.

L'usage de la commande vocale permet des **échanges plus rapides**. En corrigeant les mesures de façon à se ramener au cas d'un système de reconnaissance répondant en une seconde (au lieu de deux à quatre secondes du matériel utilisé dans cette manip) on obtient alors pour l'ensemble des opérateurs et des sous-tâches, une amélioration de la vitesse de travail de 1,66 en faveur de la commande vocale (entre 1,1 et 2 selon le cas). Cet avantage est d'autant plus net que les ordres correspondent à des phrases longues ($\geq$ 3mots).

Les menus occupent de la **place sur l'écran**, que ce soit sous forme de clavier logiciel ou sous forme de menu pointable par curseur. Cette occupation peut être permanente (environ 10% de la surface) ou temporaire. Dans tous les cas c'est de la surface perdue pour l'affichage des informations proprement dites. La commande vocale peut permettre de récupérer cette surface.

La commande vocale ne nécessite pas un **éclairage de clavier** pour fonctionner en ambiance nocturne.

5.1.5 Modification à l'issue des tests

Nous avons fait remarquer dans le paragraphe précédent l'importance de la confirmation visuelle de l'ordre reconnu.

Nous avons entrepris, à l'issue des tests, de mettre en place un mécanisme de simulation de l'ordre éxécuté.

5.2 L'APPLICATION DIVA

5.2.1 Le contexte de l'étude

L'objectif de cette seconde étude est de mettre en place une démonstration dans le domaine ASM (Activité Sous-Marine) d'une entrée vocale sur console qui soit dans la mesure du possible appliquée à une tâche réelle pouvant être confiée à des utilisateurs opérationnels. Cette démonstration devrait permettre de mesurer le gain de productivité sur tâches réelles avec et sans emploi de l'entrée vocale.

Pour effectuer cette étude, nous avons choisi un système d'aide à la classification des bruiteurs sous-marins. Il s'agit du système DIVA développé au CERDSM du BRUSC (83), et plus précisément de la première maquette du système [TASSET 1991].

5.2.2 Présentation de l'application

Le système actuel se présente sous la forme de deux postes de travail séparés et complémentaires :

- un poste TS/TI (Traitement Signal et Information),
- un poste SE (Système Expert).

Notre étude est relative à l'interaction avec le poste TS/TI. Il est constitué de deux écrans :

- un écran IHM[1] Dialogue (une station DEC GPX sous ULTRIX et OSF-MOTIF),
- un écran IHM Graphique (un écran de visualisation graphique GOULD).

Grâce à ces deux écrans disposés l'un au dessus de l'autre, une souris, une boule et le clavier de la GPX [FIGURE 5.13], l'opérateur peut :

- choisir une partition de l'écran GOULD en 1 à 4 zones,
- définir des traitements correspondant à chaque zone et leurs paramètres,
- définir ce qui doit être visualisé,
- manipuler un certain nombre d'outils pour prendre des décisions ou faire des mesures.

---

[1]IHM Interface Homme-Machine

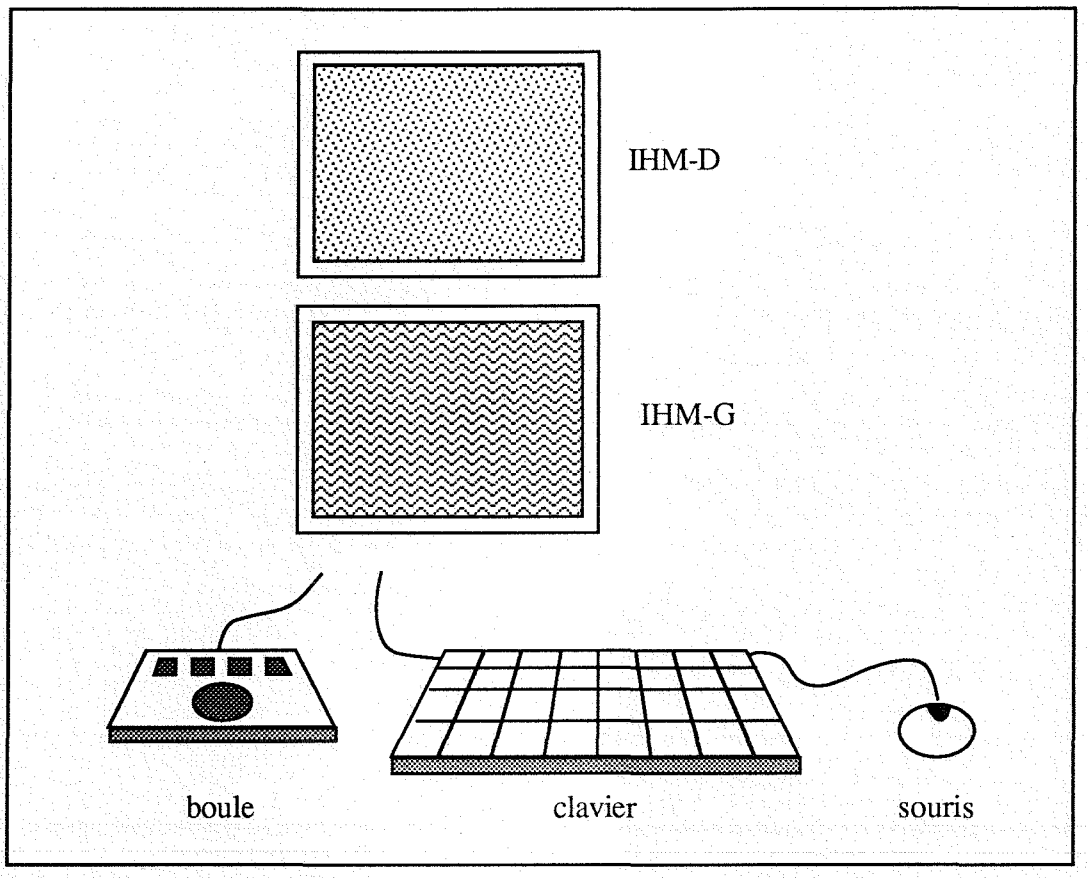


FIGURE 5.13 : L'IHM DIVA

L'opérateur prend connaissance des résultats des différentes analyses par leur affichage sur l'écran GOULD.

5.2.3 Le choix des ordres donnés en vocal

Une première sélection des ordres susceptibles de présenter un intérêt sous forme vocale a été faite par la personne du CERDSM connaissant le mieux l'interaction DIVA au point de vue ergonomie. Il s'agit généralement d'ordres pouvant être donnés tout en observant directement l'écran GOULD.

Ensuite les ordres vocaux ont été bâtis dans leur forme complète en fonction des ordres existant dans l'IHM actuelle.

La syntaxe obtenue est présentée en ANNEXE B. Le nombre de mots est légérement supérieur à une centaine compte tenu de l'énumération.

La figure qui suit [FIGURE 5.14] présente une copie d'écran de DIAPASON pour l'application DIVA. Il s'agit en fait d'une simulation réaliste des écrans IHM-D et IHM-G. Elle a été développée au CRIN-CNRS & INRIA-Lorraine dans le but de mettre au point le dialogue et l'outil plate-forme de développement de l'environnement DIAPASON.

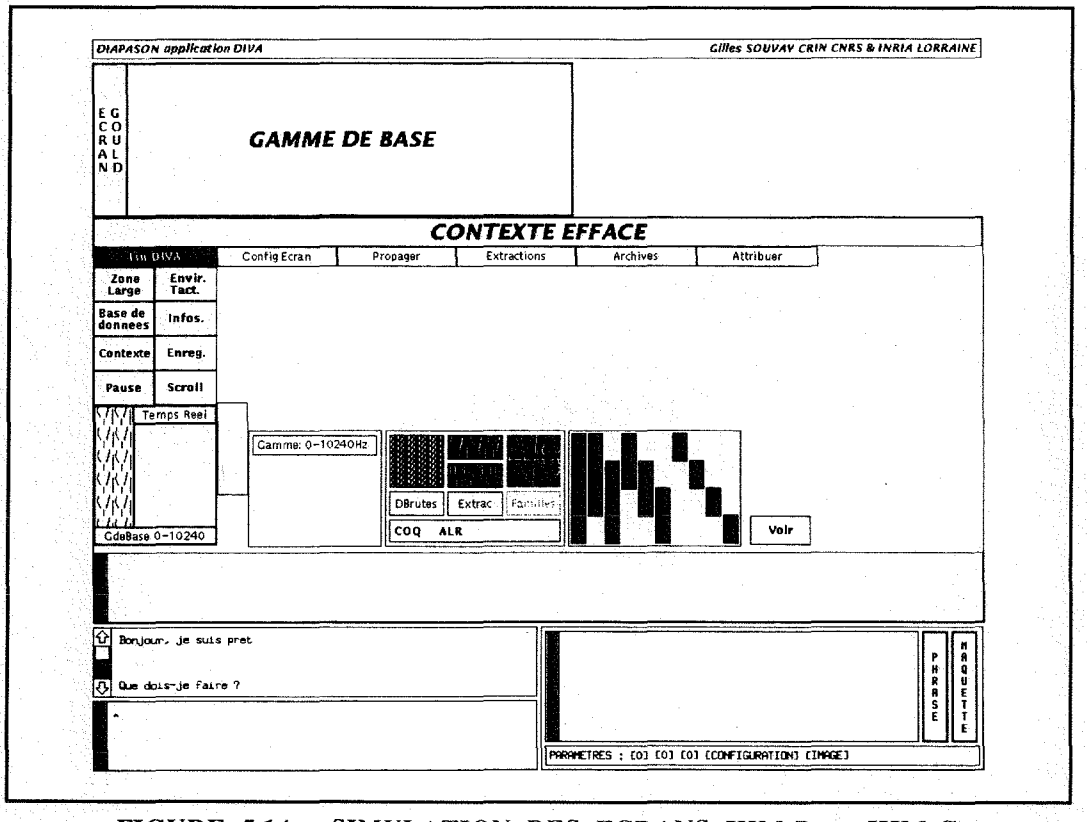


FIGURE 5.14 : SIMULATION DES ECRANS IHM-D et IHM-G

5.2.4 Interfaçage avec le système de dialogue

5.2.4.1 l'organisation logique

L'organisation générale de l'IHM DIVA avec l'entrée parole est donnée par la FIGURE 5.14. On distingue trois composantes principales :

- la maquette de reconnaissance de la parole,
- le système de dialogue,
- l'IHM DIVA.

Pour nos partenaires du CERDSM, l'ensemble porte le nom de DIVA parole, pour le CRIN, l'ensemble est appelé DIAPASON application DIVA.

Ces différentes composantes échangent des informations :

- la maquette de reconnaissance de la parole envoie des hypothèses pour la phrase prononcée. Le système de dialogue envoie au système de reconnaissance le contexte de dialogue courant afin de guider la reconnaissance.
- le système de dialogue et DIVA partagent le même contexte de l'application. Il est déclaré explicitement dans le système de dialogue et implicitement dans DIVA. Chaque modification de l'un des contextes est répercutée sur le second par l'envoi d'un message à l'autre.

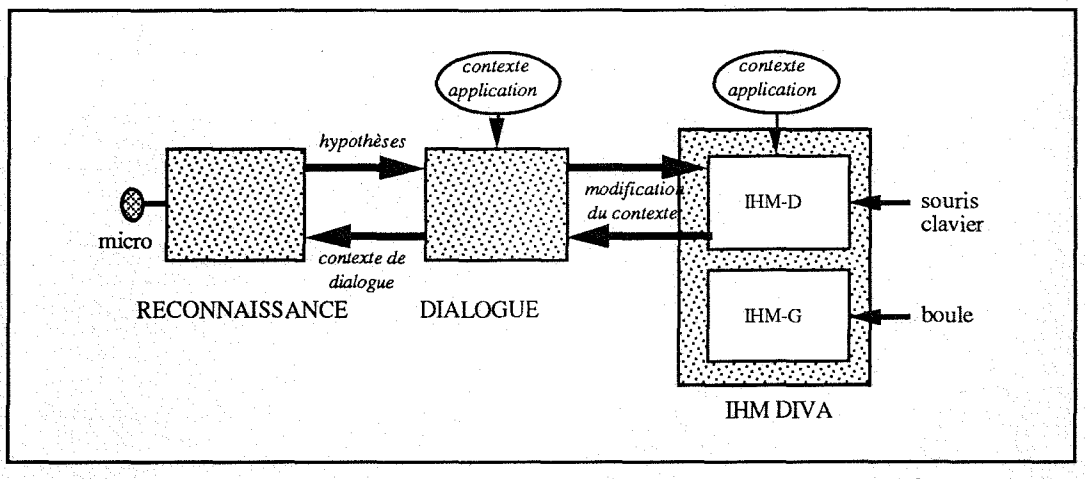


FIGURE 5.15 : ORGANISATION LOGIQUE DE DIVA PAROLE

5.2.4.2 l'organisation physique

L'implémentation physique de ce schéma logique est donné par la FIGURE 5.16. L'IHM-D ayant été développée sous OSF-MOTIF, il n'y a pas eu de difficulté à intégrer DIAPASON dans l'IHM-D. L'appel au système de dialogue se fait par des *Callbacks*.

Les différents éléments de l'IHM DIVA communiquent en utilisant un réseau DECnet. Il n'a pas été possible de connecter directement la sortie RS232 de la maquette de reconnaissance de la parole à l'IHM-D. Il a donc été nécessaire de créer un nouveau noeud sur le réseau qui est géré par un PC auquel est connectée la sortie RS232 de la maquette de reconnaissance. Les échanges entre la maquette de reconnaissance et le système de dialogue passent donc par l'IHM-D.

L'inconvénient majeur de cette implémentation, où la communication entre le système de reconnaissance de la parole et le système de dialogue est obligée de transiter par le réseau, est que le temps de réponse de l'ensemble se trouve fortement augmenté compte tenu de la charge du réseau.

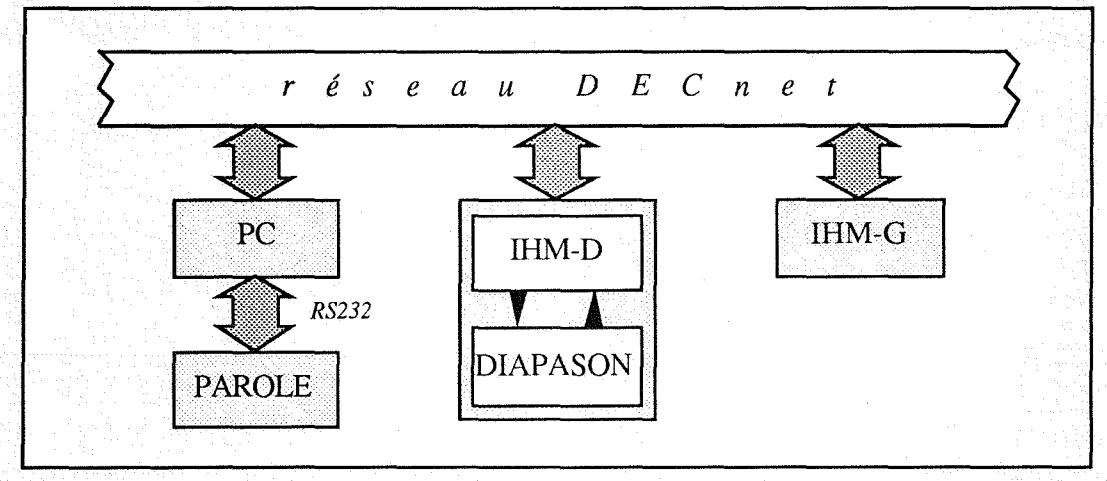


FIGURE 5.16 : ORGANISATION PHYSIQUE DE DIVA PAROLE

5.2.5 L'évaluation de DIVA Parole

5.2.5.1 les difficultés de l'évaluation

Un des objectifs de cette démonstration était d'effectuer une mesure de gain de productivité. Cet objectif à l'heure actuelle ne semble pas pouvoir être atteint. Les raisons de cette impossibilité se trouvent aux trois niveaux du système.

a) les difficultés au niveau du système de reconnaissance de la parole

Pour le vocabulaire-syntaxe de DIVA, le système de reconnaissance de la parole ne présente pas de bons taux de reconnaissance. La raison principale est que le facteur de branchement a une valeur moyenne de 30 avec de nombreux mots courts alors que la maquette est conçue pour un facteur de l'ordre de 10. L'algorithme de recherche de l'analyseur syntaxique a alors l'inconvénient d'être pris en défaut par les mots courts qui se retrouvent à de nombreux endroits et qui écrasent les hypothèses correctes.

La mise au point de la description phonétique des mots semble insuffisante comparée aux taux de reconnaissance des mots pour la démonstration SONAR.

De nouvelles règles de coarticulation entre les mots, qui n'avaient pas été vues jusqu'à présent sont apparues. La mémoire de la maquette étant occupée à 99,9%, il n'a pas été possible de modifier les algorithmes pour remédier à ces problèmes.

D'un point de vue général, on peut dire que la maquette de reconnaissance de la parole, qui date de 1982, est du point de vue technologique inadaptée à cette démonstration.

b) les difficultés au niveau de l'IHM DIVA

L'IHM DIVA qui a été connectée au système de dialogue est une maquette du système. De plus elle n'est plus maintenue (logiciellement et matériellement).

Le CERDSM, qui n'est pas lié contractuellement, souhaitait avoir un minimum de modifications à réaliser. De toute façon, il aurait été impossible d'effectuer des modifications importantes, les personnes qui ont développé le système ayant quitté le centre. Les seules interventions possibles ont été au niveau de l'IHM-D. La démonstration a donc perdu une grande partie de son intérêt dès le départ car il n'était pas possible alors d'introduire de contre-réactions graphiques sur l'IHM-G, écran sur lequel l'opérateur est censé observer les résultats des traitements demandés.

L'IHM DIVA a subi de nombreuses pannes qui ont entraîné la suppression de certains traitements dans le cas de pannes logicielles et retardé le projet lorsqu'il s'agissait de pannes matérielles.

c) les difficultés au niveau du système de dialogue

Le système de dialogue n'a pas été suffisamment mis au point, ce qui peut amener à des lourdeurs au niveau des dialogues de correction par exemple. Elles s'avèrent même impossibles si on les combine au mauvais taux de reconnaissance des phrases.

d) rapport de synthèse

Cette démonstration donnera lieu à un rapport de synthèse final pour la DRET [ALINAT 1992] à paraître et a fait l'objet d'un rapport interne au niveau du CERDSM [MAIS 1992].

5.2.5.2 le scénario

Malgré ces difficultés nous avons mis au point un scénario représentatif de la tâche à réaliser par un opérationnel et évitant les principales faiblesses rencontrées. Le texte du scénario est donné en ANNEXE F.

5.2.5.3 conclusion

On peut dire que cette démonstration doit plutôt être présentée comme un échec dont les raisons principales sont :

- les mauvais taux de reconnaissance du système de reconnaissance,
- les pannes logicielles et matérielles de l'IHM DIVA,
- l'insuffisance de mise au point du dialogue,
- le mauvais temps de réponse de l'ensemble.

Elle n'a pu conduire à une évaluation réelle de l'introduction d'une composante orale dans un système de commande de machine complexe. Une démonstration du système dans l'état actuel de ses performances ne pourrait jouer qu'en défaveur de l'utilisation de la parole.

Il existe néanmoins quelques points positifs :

- une maquette de reconnaissance plus moderne est en cours de réalisation et devrait permettre de tenir compte des enseignements de cette expérience,
- nous avons sensibilisé le CERDSM au fait qu'introduire une entrée vocale dans un système n'est pas simplement introduire une entrée supplémentaire, mais que les aspects liés au contexte de dialogue et de la tâche sont très importants et qu'ils doivent être pris en compte,
- cette application a permis de spécifier et réaliser les outils de dialogue nécessaires à l'implémentation d'une nouvelle application (plate-forme de développement, éditeur, compilateur...) mais ils demandent néanmoins une validation,
- elle montre bien que l'intégration d'une entrée parole pour être efficace et fonctionnelle doit être prise en compte dès la phase de spécification d'un système complexe.

Chapitre 6

LES OUTILS POUR LE DIALOGUE

6.1 INTRODUCTION

Lorsque l'on conçoit un système d'interprétation et de gestion de dialogue oral, il ne faut pas se contenter de le connecter à une application donnée, il faut rester le plus générique possible. Si ce principe n'est pas respecté, le concepteur d'un tel système peut être amené à réécrire un système complet lorsqu'une nouvelle application est à développer. Il convient donc de mettre en place, en plus du système d'interprétation et de gestion du dialogue, des outils qui devront permettre de minimiser le travail lors de la mise en place de la nouvelle application. L'ensemble formera un environnement de travail qui dans notre cas tire son nom du système de dialogue : l'environnement DIAPASON.

La première étape pour réaliser les outils a consisté à séparer les connaissances générales à mettre en action pour toute application et celles spécifiques à une application donnée. Les premières vont permettre de construire le système de dialogue (algorithmes et structures manipulées, interfaces entre les différentes composantes), les secondes de remplir les structures ainsi définies (représentation des connaissances).

Ce travail réalisé permet de passer à la seconde étape, la réalisation des différents programmes qui forment l'environnement DIAPASON [FIGURE 6.1] c'est-à-dire :

- d'une plate-forme de développement qui est constituée du système de dialogue, des interfaces visuelles pour la mise au point, des fichiers source de l'application,
- d'une interface d'édition des structures syntaxico-sémantiques manipulées par le système de dialogue,
- d'un compilateur qui transforme les données issues de l'éditeur en données compréhensibles par les composantes du système de dialogue oral,
- de logiciels de mise au format d'une description syntaxique générale en une description particulière à un système de reconnaissance de la parole donné,
- d'outils d'accès spécifiques à l'historique du dialogue,
- d'outils de représentation de l'état de la tâche.

L'ensemble des ces outils a été développé sur station de travail sous UNIX et dans un environnement X Window System.

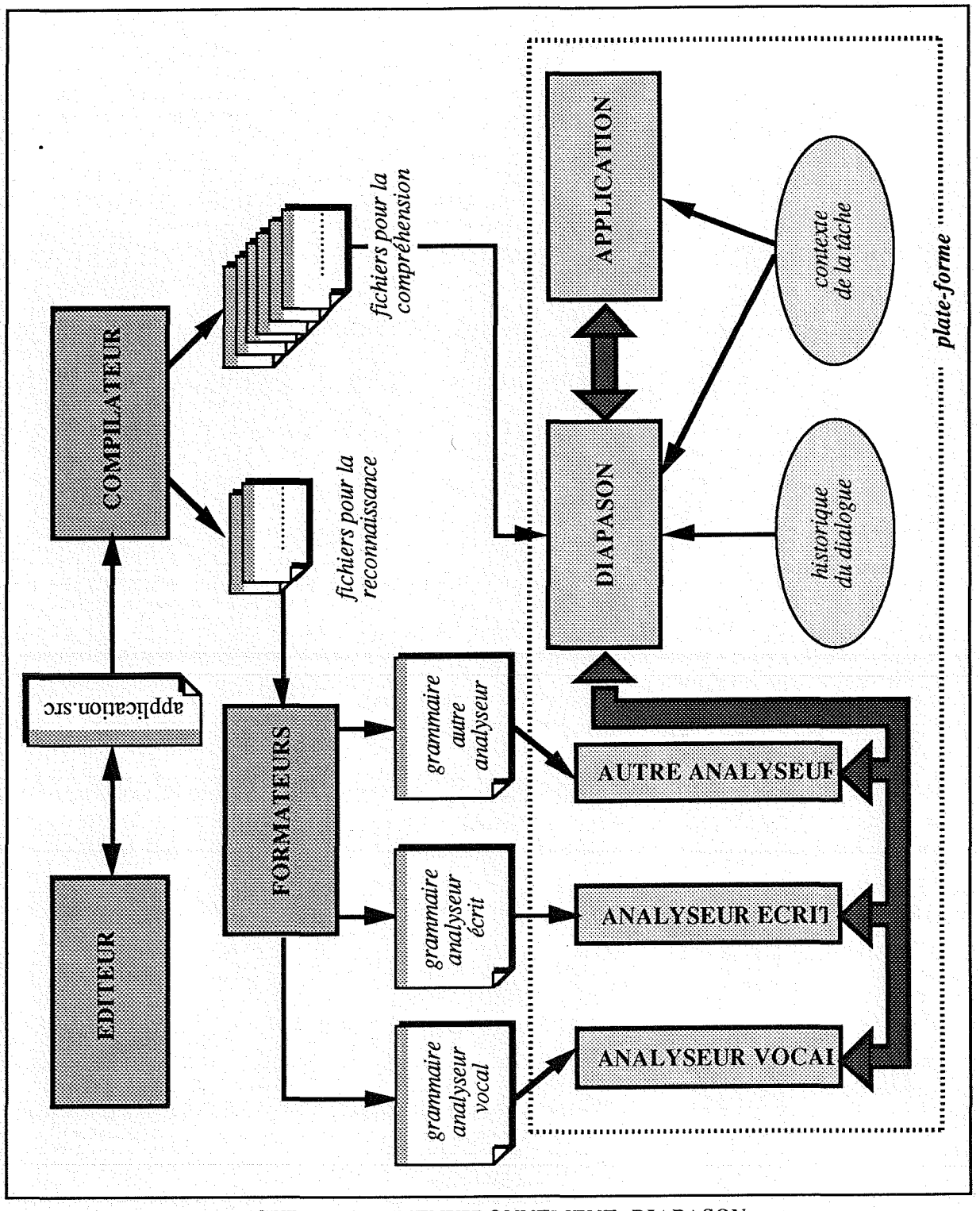


FIGURE 6.1 : L'ENVIRONNEMENT DIAPASON

6.2 LA PLATE-FORME DE DEVELOPPEMENT

6.2.1 Introduction

Les deux expériences que nous avons menées avec l'application SONAR et l'application DIVA, nous ont amené à réaliser une plate-forme de développement. Elle permet de créer et de tester une nouvelle application sans avoir à réécrire le système de dialogue. Il reste à réaliser les visualisations spécifiques à l'application et l'interface avec le système de dialogue. Les aspects liés au langage de commande sont traités par d'autres outils que nous présentons plus loin dans ce chapitre. Les spécifications complètes de la plate-forme de développement sont données en annexe.

6.2.2 Aspects visuels de la plate-forme

6.2.2.1 aspect visuel général

La plate-forme de développement de DIAPASON est composée de cinq fenêtres principales : la fenêtre titre, la fenêtre application, la fenêtre dialogue, la fenêtre clavier et la fenêtre hypothèses [FIGURE 6.2].

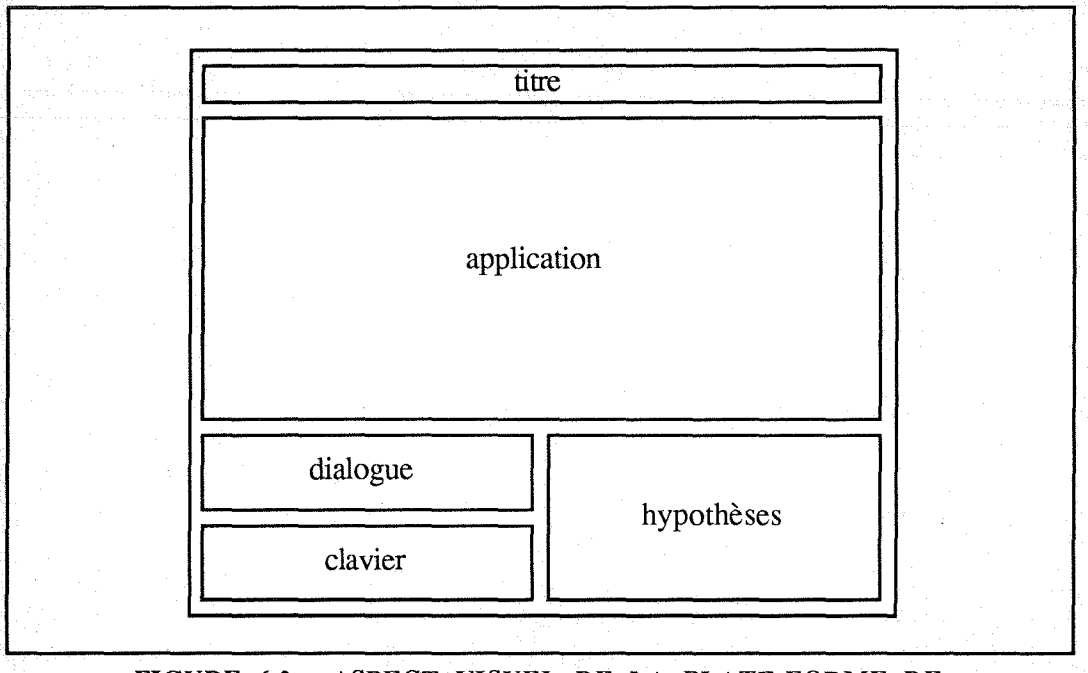


FIGURE 6.2 : ASPECT VISUEL DE LA PLATE-FORME DE DEVELOPPEMENT

6.2.2.2 la fenêtre titre

La fenêtre titre a deux composantes : le nom de l'application à gauche et le nom du concepteur à droite.

6.2.2.3 la fenêtre application

Le contenu de la fenêtre application est à définir par le concepteur du système de dialogue. Elle contiendra toutes les informations visuelles relatives à la tâche.

6.2.2.4 la fenêtre dialogue

La fenêtre dialogue est composée d'un ascenseur et de trois lignes de texte [FIGURE 6.3]. Chaque ligne a sa fonction. La première affiche la phrase reconnue par le module de reconnaissance. Le contenu de la deuxième et de la troisième est fonction de l'interprétation du système. Si l'ordre est cohérent, la deuxième ligne est vide et la troisième contient la phrase interprétée. Si l'ordre est incohérent, la deuxième affiche la phrase interprétée et la troisième le message d'incohérence. Losque DIAPASON pose une question la deuxième ligne est vide et la troisième contient le texte de la question. L'ascenseur permet de faire défiler le texte par groupe de trois lignes.

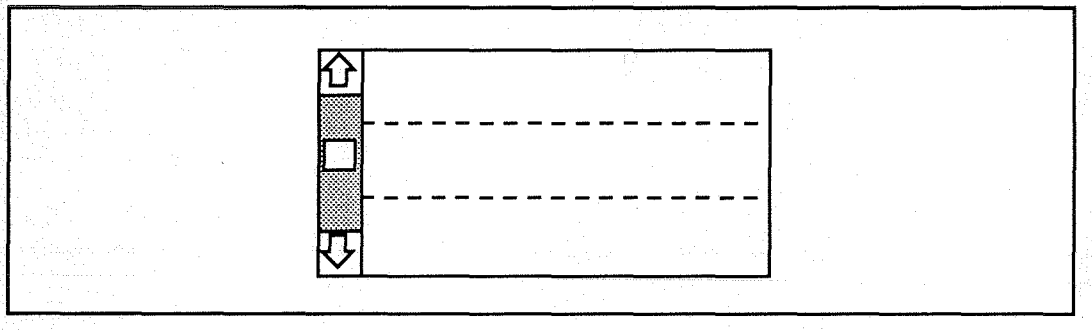


FIGURE 6.3 : LA FENETRE DIALOGUE

6.2.2.5 la fenêtre clavier

La fenêtre clavier est une simple fenêtre texte munie d'une barre de scroll. Après chaque retour chariot la ligne courante est envoyée au module de reconnaissance de texte.

6.2.2.6 la fenêtre hypothèses

La fenêtre hypothèses est composées de quatre sous-fenêtres [FIGURE 6.4]. Elles permettent de consulter l'ensemble des hypothèses envoyées par le module de reconnaissance pour la phrase courante et d'afficher les paramètres de reconnaissance. La sous-fenêtre texte affiche les informations demandées à l'aide du bouton affichage. Ce dernier permet la sélection entre le texte des phrases reconnues, la structure envoyée par la maquette de reconnaissance et la structure sémantique correspondante. Le second bouton mode, permet de choisir entre deux modes de fonctionnement : entrée en mode maquette de reconnaissance simulée ou en mode structure sémantique.

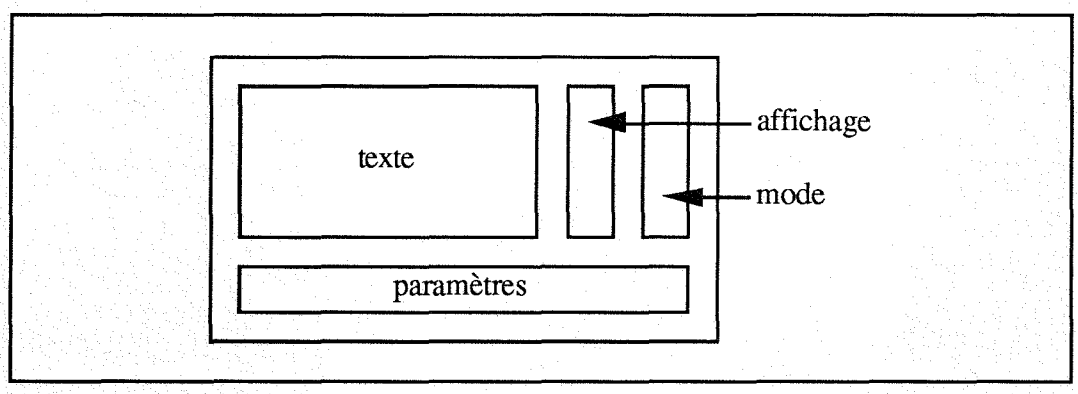


FIGURE 6.4 : LA FENETRE HYPOTHESES

6.2.2.7 exemples

Le premier exemple présente l'aspect visuel de la plate-forme d'intégration pour une application vide [FIGURE 6.5].

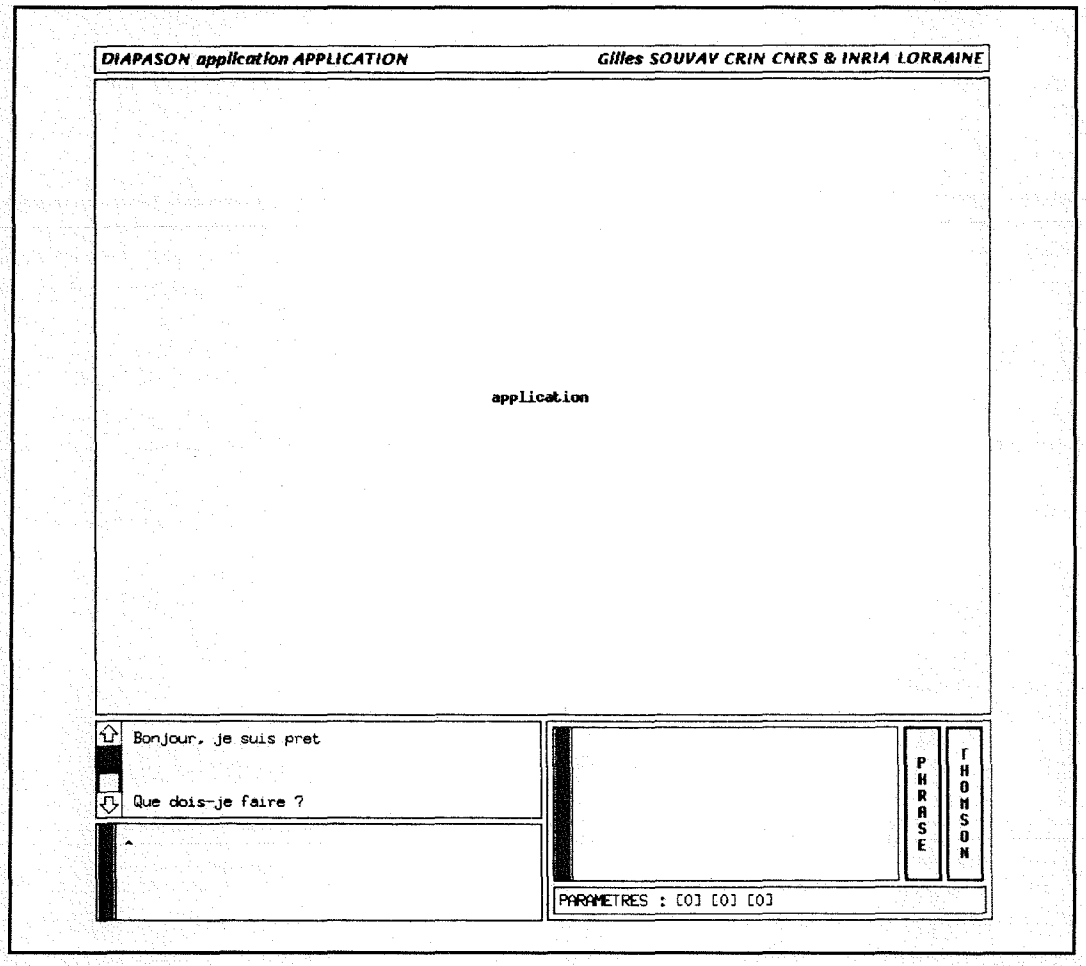


FIGURE 6.5 : ASPECT VISUEL DE LA PLATE-FORME DE DEVELOPPEMENT

Les exemples qui suivent [FIGURE 6.6], [FIGURE 6.7] et [FIGURE 6.8] présentent le contenu de la fenêtre hypothèse pour la phrase "image VPR" de l'application SONAR.

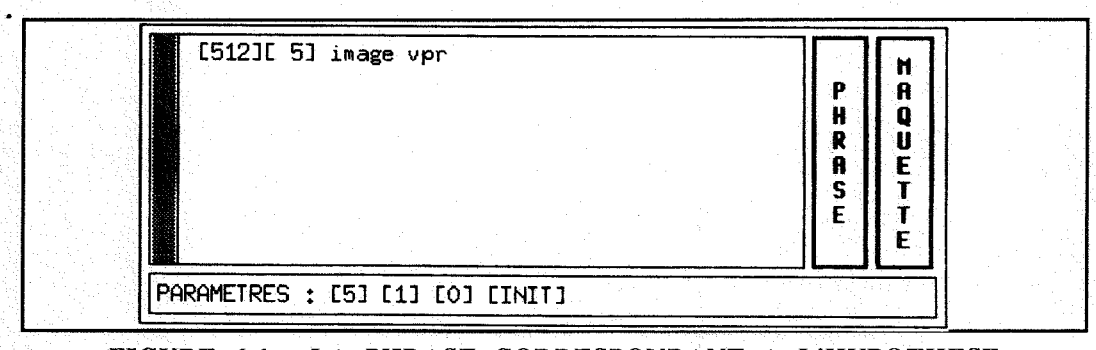


FIGURE 6.6 : LA PHRASE CORRESPONDANT A L'HYPOTHESE

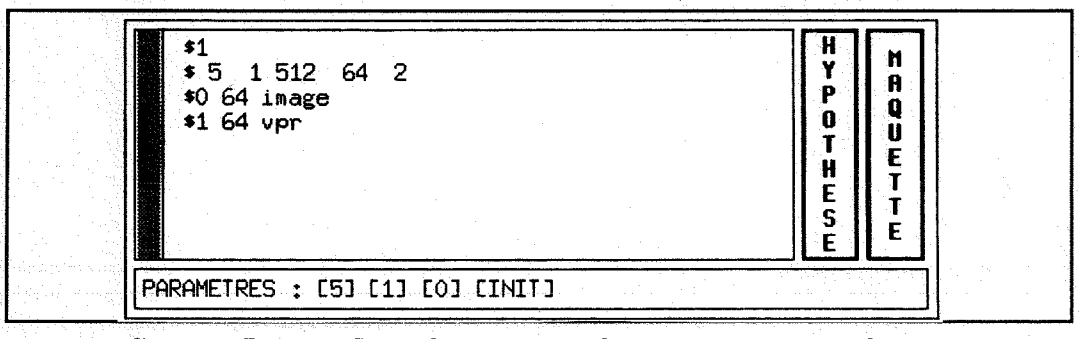


FIGURE 6.7 : LA STRUCTURE ENVOYEE PAR LA MAQUETTE

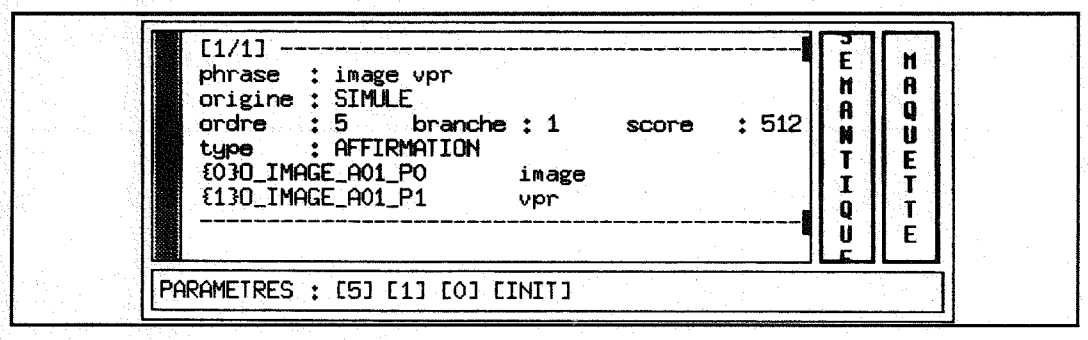


FIGURE 6.8 : LA STRUCTURE SEMANTIQUE CORRESPONDANTE

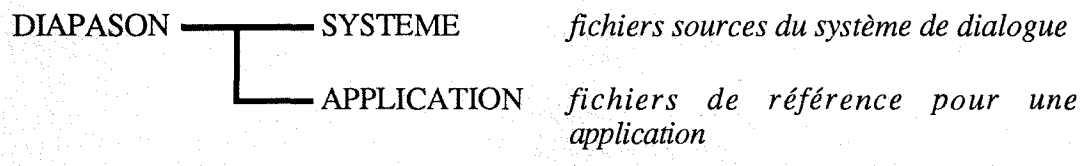
6.2.3 Organisation des fichiers sources

6.2.3.1 installation minimale du système de dialogue

L'installation minimale correspond à l'ensemble des fichiers nécessaires pour faire fonctionner le système de dialogue avec une application vide. Cette application vide porte le nom d'APPLICATION.

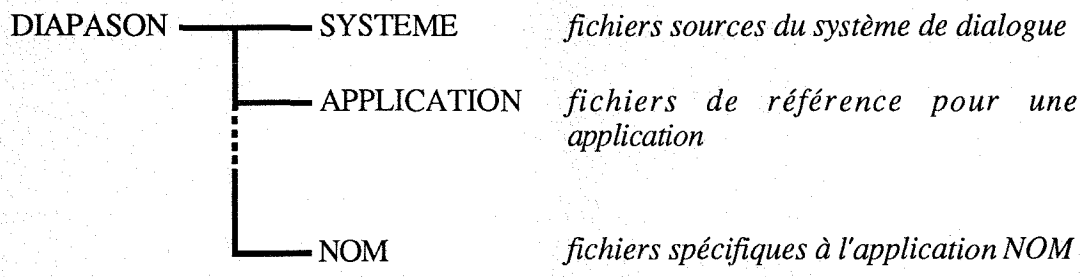
Elle se fait sous le répertoire DIAPASON qui contient les deux sous-répertoires SYSTEME et APPLICATION. Le premier regroupe tous les fichiers du système de

dialogue indépendants de l'application, le second tous les fichiers de référence pour une application.



6.2.3.2 installation d'une nouvelle application

L'installation d'une nouvelle application consiste à créer un sous-répertoire et à y copier les fichiers de référence situés sous APPLICATION. Il reste ensuite à remplir les structures et fonctions spécifiques à l'application. Par convention le nom du sous-répertoire sera identique au nom de l'application.



6.2.3.3 Installation particulière

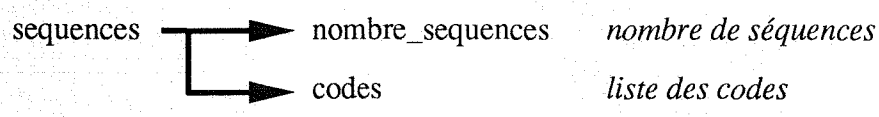
Il est possible d'installer le système de dialogue et l'application sous un répertoire unique. Cette installation se fait si le concepteur de système de dialogue est amené à ne travailler que sur une seule application ou pour l'installer sur la machine où fonctionne l'application réelle.

6.2.4 La communication avec l'application

DIAPASON et l'application partagent le même contexte de la tâche. Il faut donc que si un ordre de DIAPASON provoque un changement de contexte, l'application en soit informée et inversement.

La communication entre DIAPASON et l'application passe actuellement par l'envoi de séquences de codes.

Structure des séquences :



action élémentaire sur l'application

réponse attendue après le code aller

paramètres de l'action

Le contexte de la tâche interviendra une fois de plus au moment de l'affectation des séquences afin de minimiser le nombre de codes envoyés.

Exemple de séquences pour l'application DIVA

Soit l'ordre : **image 1 temps début 10**

Il y a deux actions élémentaires à effectuer sur l'application
1) sélection de l'image
2) envoi de la valeur de temps début
La séquence élémentaire pour la sélection de l'image porte le numéro 7 et possède un paramètre : le numéro de l'image - 1.
La séquence élémentaire pour l'envoi de temps début porte le numéro 59 et possède trois paramètres : le temps décomposé en heures, minutes, secondes.

Deux cas de figures se présentent alors pour l'envoi des séquences

a) l'image 1 n'est pas sélectionnée

nom séquence	*aller*	*retour*	*suite*
sélection image	7	8	0
temps début	59	60	0 10 0

b) l'image 1 est déjà sélectionnée

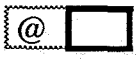
nom séquence	*aller*	*retour*	*suite*
temps début	59	60	0 10 0

6.2.5 Intégration de la maquette de reconnaissance

6.2.5.1 la communication de THOMSON vers DIAPASON

La maquette de reconnaissance envoie à DIAPASON les hypothèses pour la phrase prononcée. Il s'agit d'une série de lignes de caractères contenant toutes les informations nécessaires pour la conversion en structure sémantique. L'arrangement de ces informations est variable pour une application. Le format présenté en exemple est celui que nous utilisons pour l'application DIVA.

type de ligne contenu

NBHYP: 

GLOBAL: 

LOCAL: 

NBHYP	nombre d'hypothèses,
GLOBAL1	code syntaxique de l'ordre,
GLOBAL2	le numéro de branche,
NORME	score normé de l'hypothèse $\in [0, 512]$,
SILENCE	note de silence $\in [0, 64]$,
NBMOTS	nombre de mots de l'hypothèse,
LOCAL1	numéro du paramètre (même valeur pour tous les mots d'un élément) ou identification d'un élément commun,
SCORE	score du mot $\in [0, 64]$,
MOT	mot en toute lettre.

Les codes GLOBAL1 et GLOBAL2 sont des entiers. L'élément neutre est 0.
Le code LOCAL1 est un caractère. L'élément neutre est le caractère espace.

6.2.5.2 la communication de DIAPASON vers THOMSON

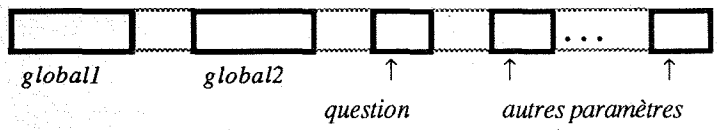
La communication de DIAPASON vers THOMSON consiste à envoyer une série de nombres qui permettent d'ouvrir les grammaires valables pour un état de la tâche donné. Quelle que soit l'application, on trouve :

- en première position GLOBAL 1,
- en deuxième position GLOBAL 2,
- en troisième position un indicateur pour les questions. Lorsqu'il n'y a pas de question posée la valeur est 0, sinon il s'agit du numéro du paramètre augmenté de 1.

On trouve ensuite une série de nombres particuliers à chaque application, par exemple :

- pour l'application SONAR, un paramètre supplémentaire le type de l'image,
- pour l'application DIVA, un paramètre supplémentaire le mode de traitement.

Les codes sont à envoyer chaque fois qu'il y a eu interprétation d'une phrase ou une action opérateur sur l'application. Il faut envoyer les valeurs initiales à l'initialisation du système de dialogue.



6.2.6 L'analyseur syntaxique écrit ASE

La plate-forme de développement inclut un analyseur syntaxique pour l'entrée écrite. Il s'agit plus d'un outil de remplacement de l'entrée vocale que d'une véritable entrée clavier. En effet il impose une syntaxe rigoureuse et ne tolère pas les fautes de frappe.

Cet analyseur est paramètrable par la grammaire de l'application. Pendant le phase d'initialisation du système de dialogue, il y a chargement de la grammaire contenue dans le fichier "application.ase". Ce fichier est construit de manière automatique par l'outil formateur présenté plus loin dans ce chapitre.

6.3 L'EDITEUR DE STRUCTURES SYNTAXICO-SEMANTIQUES

6.3.1 Introduction

L'éditeur de structures syntaxico-sémantiques est un outil qui va permettre au concepteur de système de dialogue de saisir l'ensemble des ordres pour une application donnée. La saisie sera la plus proche possible du formalisme présenté dans le paragraphe précédent.

Une première version de l'éditeur a été développée dans un environnement X Window System en utilisant la bibliothèque de XLIB [LONG 1991]. Une version plus conviviale, avec une révision des spécifications, est en phase finale de développement dans un environnement X MOTIF [DRUART 1992].

6.3.2 Saisie de la grammaire

6.3.2.1 présentation de l'écran de saisie

L'écran de saisie est composé de deux types de fenêtres : le fenêtres de commande et les fenêtres d'affichage [FIGURE 6.9][FIGURE 6.10].

Les fenêtres de commande sont :		Les fenêtres d'affichage sont :
grammaires	gestion des grammaires des applications	la liste des grammaires
ordres	gestion des ordres de la grammaire	la liste des ordres
X	fenêtres de défilement des ordres	l'ordre courant
Aide	appel à l'aide en ligne	
Objet	génération du fichier objet	
Sortie	sortie du logiciel	

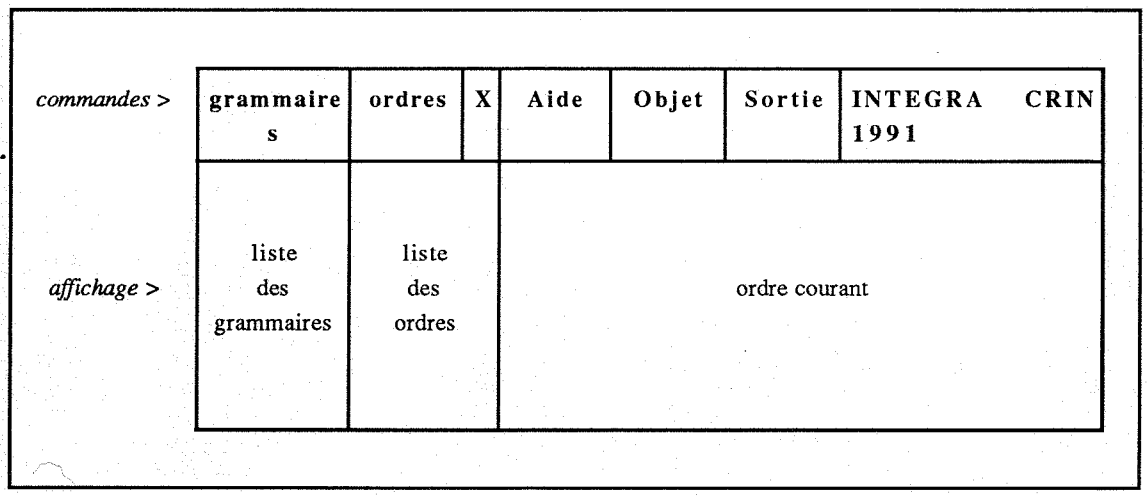


FIGURE 6.9 : LES ZONES DE L'ECRAN DE SAISIE D'INTEGRA

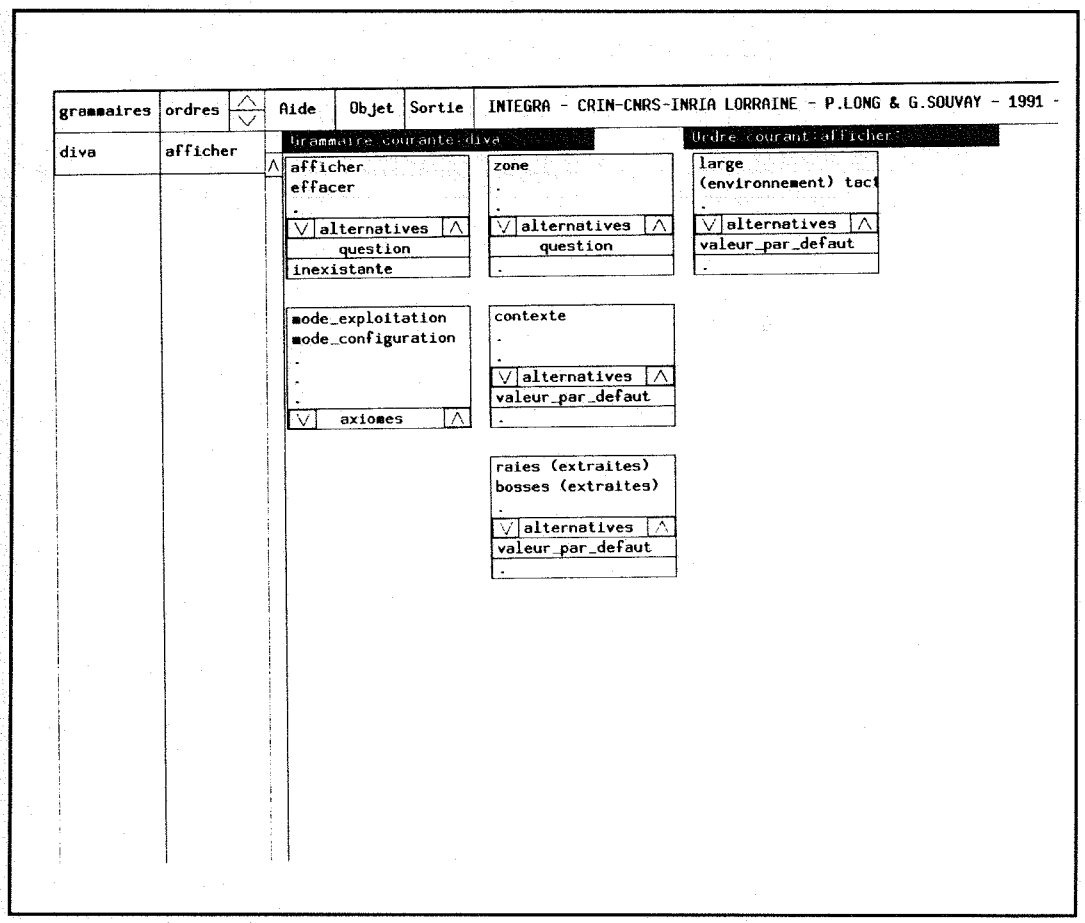


FIGURE 6.10 : L'ECRAN DE SAISIE D'INTEGRA

6.3.2.2 principes de la saisie

Il existe trois niveaux de saisie :

- la saisie de la grammaire de l'application,
- la saisie d'un ordre,
- la saisie d'un élément.

A chaque niveau, trois types d'opérations sont possibles :

- l'adjonction,
- la modification,
- la suppression.

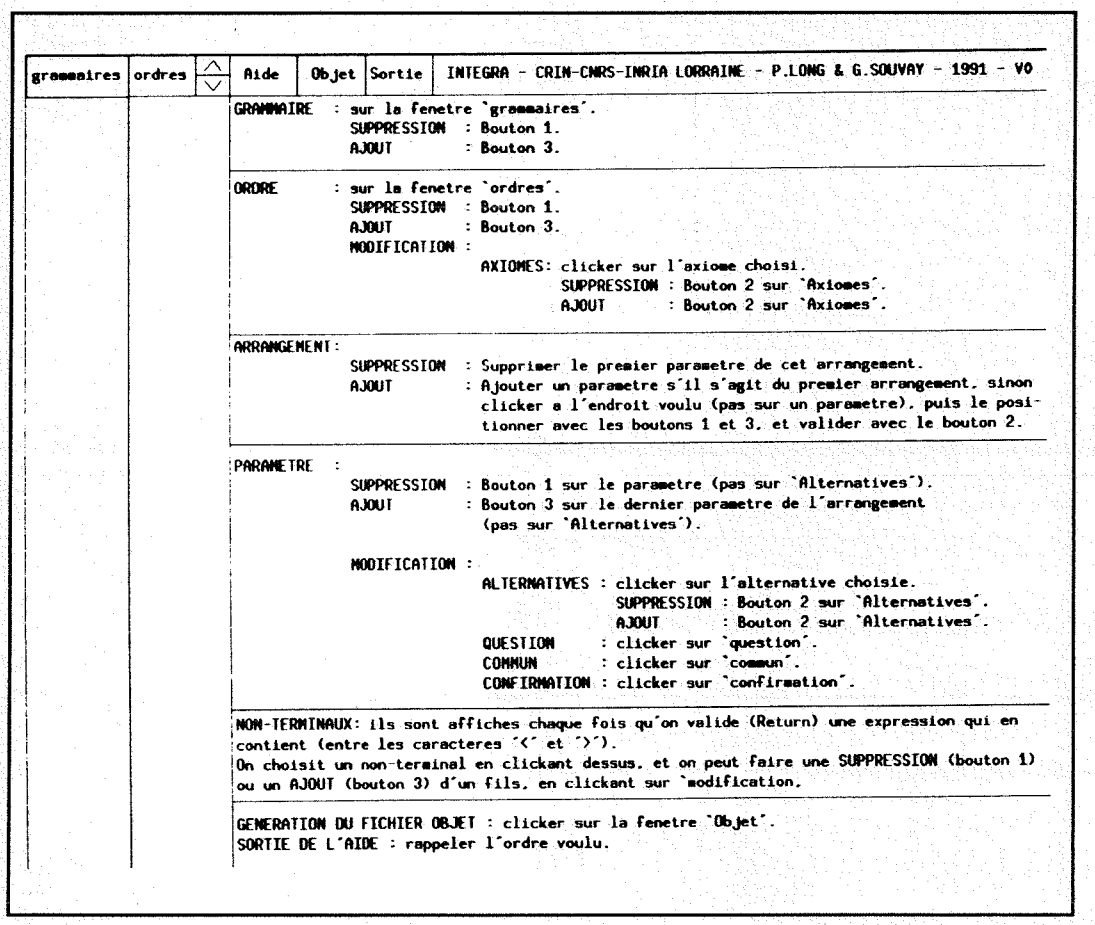


FIGURE 6.11 : L'ECRAN D'AIDE D'INTEGRA

L'opérateur doit choisir une grammaire et un ordre de travail. Il peut ensuite travailler sur les éléments de l'ordre. Lors de la création d'un élément, l'opérateur doit répondre à une série de questions qui permettent de saisir les alternatives de l'ordre, le type de confirmation associé à la branche en cours, la question à poser à l'opérateur si l'élément en cours de définition vient à manquer, le caractère commun. Une fois la création terminée, ll peut à son gré modifier les caractéristiques de l'élément. L'écran qui suit est l'aide en ligne fournie par INTEGRA. Un exemple de saisie est donné en annexe.

6.3.2.3 la représentation graphique des ordres

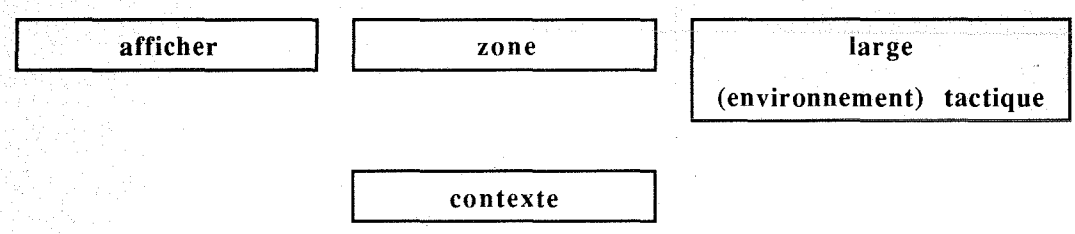
La représentation graphique des ordres permet de définir d'une manière précise tous les éléments qui composent un ordre.

On trouve en ligne, les alternatives sur les éléments et en colonne les suivants. Ils permettent de composer un ordre complet.

A l'intérieur d'un élément :

- le saut à la ligne correspond aux alternatives,
- les parenthèses indiquent les mots ou parties de mots facultatifs,
- le mots encadrés par < et > sont des non-terminaux lexicaux,
- les valeurs par défaut sont encadrées par les crochets [et].

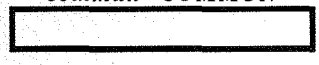
L'exemple qui suit donne quatre possibilités :



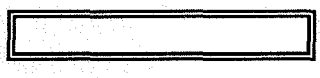
1- afficher zone large,
2- afficher zone environnement tactique,
3- afficher zone tactique,
4- afficher contexte.

On trouve trois types de cadre pour marquer le contour d'un élément. Ils correspondent à des informations sémantiques :

cadre trait plein simple,
aucune information sémantique supplémentaire,



cadre trait plein double,
élément commun à plusieurs ordres, le nom de l'élément apparaît au dessus du cadre,



cadre double trait,
élément sémantique qui n'a pas de formulation syntaxique, le système de compréhension les trouve dans sa mémoire ou ses connaissances. Il peut posséder un caractère commun.

6.3.2.4 Le fichier objet

Le fichier objet contient toutes les informations saisies à l'éditeur. elles sont destinées aux systèmes de reconnaissance et au système de dialogue. La structure interne de ce fichier respecte une syntaxe très précise qui est donnée avec les spécifications de l'outil compilateur en ANNEXE E.

EXEMPLE : Cet exemple présente un exemple de fichier pour l'ordre AFFICHER introduit dans le paragraphe précédent.

```
grammaire=ESSAI;
langue=F;
ordre=AFFICHER , 1
confirmation=INEXISTANTE
axiomes=
rang=STANDARD
{
  ELEMENT_1_0  : afficher
               | effacer
               ;
               question="%0 quoi ?"
               ;
  ELEMENT_1_1  : zone
               ;
               question="%0 quelle zone ?"
               commun=FONCTION
               ;
  ELEMENT_1_2  : large
               | (environnement) tactique
               ;
               commun=MEMOIRE_POINTEE
               ;
  ELEMENT_2_1  : contexte
               ;
}
```

6.4 LE COMPILATEUR

Le compilateur des structures syntaxico-sémantiques va permettre de fournir au système de dialogue l'ensemble des connaissances spécifiques à l'application qui auront été saisies à l'aide de l'éditeur.

A partir du fichier issu de l'éditeur, le compilateur va construire deux types de fichiers [FIGURE 6.11] :

- les fichiers destinés aux systèmes de reconnaissance qui seront connectés à DIAPASON,
- les fichiers destinés au système de compréhension.

Les spécifications complètes du compilateur de structures syntaxico-sémantiques sont données en ANNEXE E.

6.4.1 Les fichiers pour la reconnaissance

6.4.1.1 le fichier grammaire

Il contient la syntaxe de toutes les phrases qui peuvent être reconnues par le système :

- les phrases de gestion du dialogue,
- les ordres complets,
- la correction des ordres,

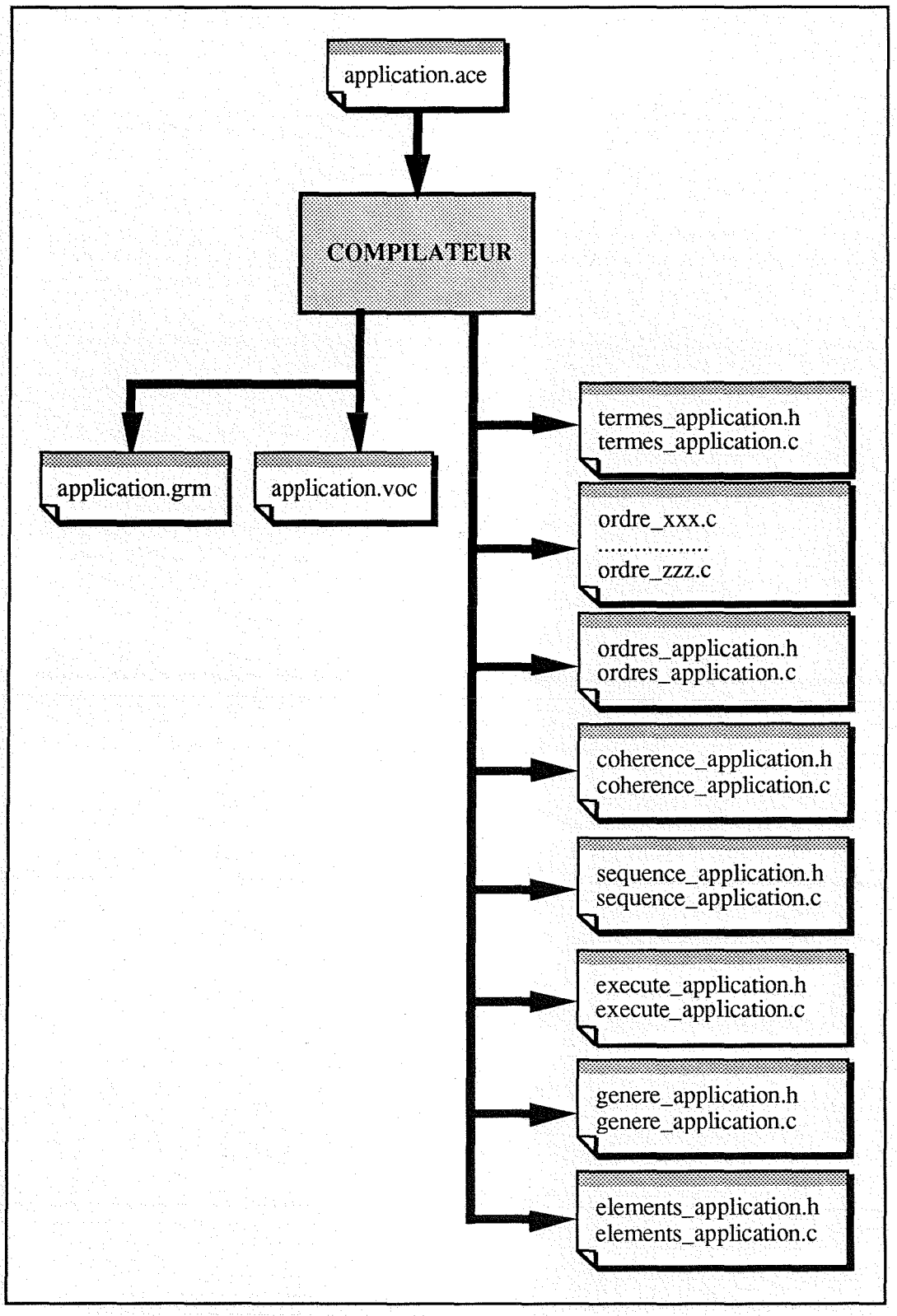


FIGURE 6.11 : LES FICHIERS ISSUS DU COMPILATEUR DE STRUCTURES SYNTAXICO-SEMANTIQUES

- les enchaînements sur le même ordre
- les enchaînements sur un ordre différent,
- les réponses aux question.

6.4.1.2 le fichier vocabulaire

Le fichier vocabulaire contient la liste du vocabulaire terminal défini pour l'application. Il est destiné à l'outil de gestion du lexique qui construira une première approche de la description phonétique des mots.

6.4.2 Les fichiers pour la compréhension

Les fichiers destinés au système de compréhension contiennent des déclarations de variables et de fonctions. Leur valeur est définie soit automatiquement par le compilateur, soit reste à être définie par le concepteur du système de dialogue. Le contenu détaillé de ces fichiers est présenté en ANNEXE E.

6.5 LES FORMATEURS

Les formateurs ont pour fonction de transformer les fichiers du vocabulaire-syntaxe général issus du compilateur en fichiers décrivant le même vocabulaire-syntaxe dans un format spécifique à l'analyseur auquel il est destiné.

6.5.1 Le formateur pour l'analyseur oral

Le formateur pour l'analyseur oral est en cours de spécification à THOMSON SINTRA DASM. Il devrait générer deux types de fichier :

- un fichier de description syntaxique des phrases,
- un fichier de description phonétique des mots du vocabulaire.

Cet outil se veut être une aide, il devrait réaliser une première ébauche du travail fait auparavant manuellement. Pour une description plus fine au niveau de la description phonétique en particulier l'intervention humaine est toujours envisagée.

6.5.2 Le formateur pour l'analyseur écrit

Le formateur pour l'analyseur écrit prend en entrée le fichier "application.grm" issu du compilateur et donne en résultat le fichier "application.ase" [FIGURE 6.12].

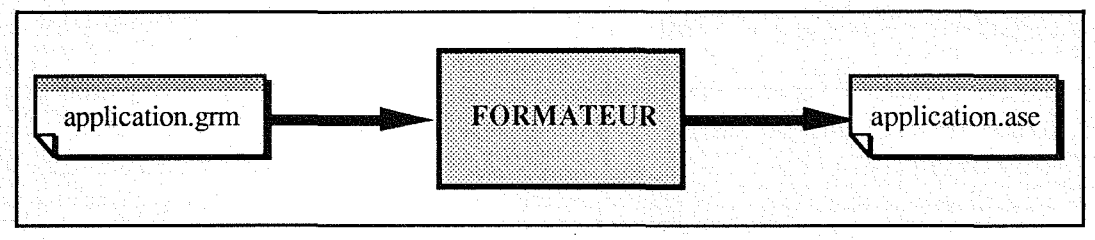


FIGURE 6.12 : LE FORMATEUR POUR L'ANALYSEUR ECRIT

Le fichier "application.ase" contient une description arborescente du vocabulaire-syntaxe de l'application.

EXEMPLE

Soit l'ordre IMAGE dont la représentation syntaxico-sémantique est la suivante :

image	**un** **deux**

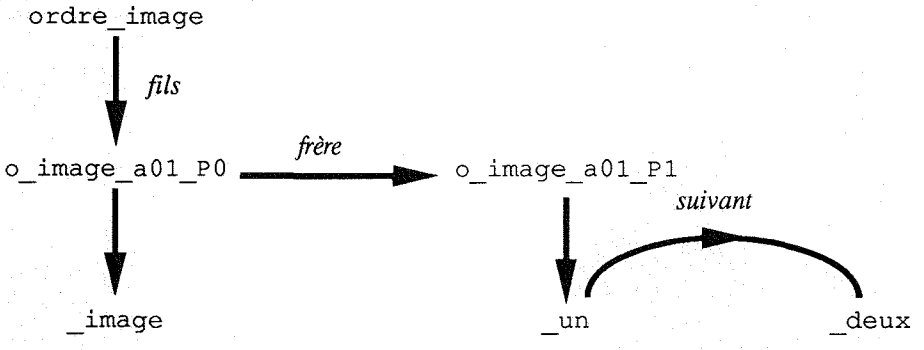
Le fichier "application.grm" issu du compilateur contiendra la déclaration suivante :

```
ordre_image       : o_image_a01_P0   o_image_a01_P1
                  ;
o_image_a01_P0    : _image
                  ;
o_image_a01_P1    : _un
                  : _deux
```

Le résultat brut issu du formateur sera le suivant :

```
     fils          frère  suivant
0  ordre_image       1      -1      -1
1  o_image_a01_P0    2       3      -1
2  _image           -1      -1      -1
3  o_image_a01_P1    4      -1      -1
4  _un              -1      -1       5
5  _deux            -1      -1      -1
```

Ce résultat correspond à l'arbre :



6.6 LA GESTION DE L'HISTORIQUE

6.6.1 Les composantes de l'historique du dialogue

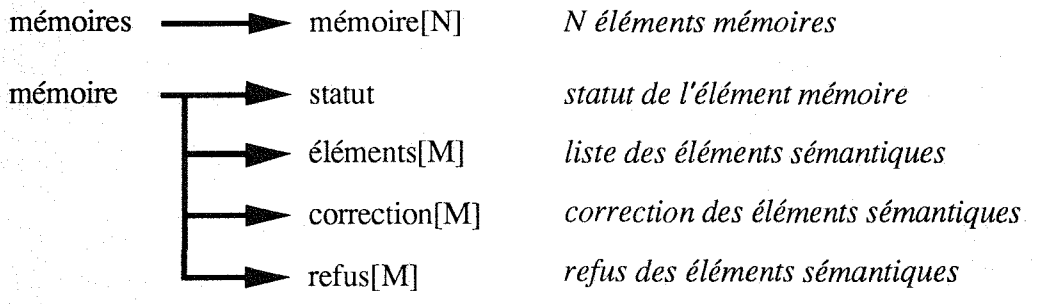
L'historique du dialogue est composé de deux mémoires : une mémoire à court terme et une mémoire à long terme.

Le but de DIAPASON est de réunir tous les éléments nécessaires à l'éxécution d'un ordre. Pour y arriver l'opérateur prononce une ou plusieurs phrases tels que la phrase initiale, les réponses aux demandes de complément d'information, les corrections du locuteur... Toutes ces interventions, ainsi que les différentes hypothèses proposées par le niveau reconnaissance pour une phrase, sont conservées dans la mémoire à court terme. Une fois que l'ordre est exécuté tous ses éléments sont conservés dans la mémoire à long terme. Si un ordre est incohérent, il est lui aussi conservé dans la mémoire à long terme. La mémoire à court terme est alors vidée.

6.6.2 La structure des mémoires

La mémoire à court terme et la mémoire à long terme ont la même structure. Ce sont des files où l'on fait des adjonctions en tête. Elles ont une taille maximum. Lorsqu'une file est pleine l'élément de queue est perdu. En effet rien ne sert de conserver l'historique complet des ordres donnés par l'opérateur pour résoudre les ellipses, le système n'utilise que les derniers propos de l'opérateur. "Les derniers propos" sont paramètrables de 1 à la taille maximum de la file. La recherche dans la mémoire se fait en partant de la tête de la file.

Structure de la mémoire :



Selon la mémoire, les champs sont valides ou non. Les abréviations que nous utiliserons pour indiquer la validité seront MCT pour la mémoire à court terme et MLT pour la mémoire à long terme.

Le champ **statut** d'un élément mémoire prend les valeurs suivantes :

AUCUN	valeur par défaut	MCT
EXECUTE	ordre exécuté	MLT
INCOHERENT	ordre incohérent	MCT et MLT

Le champ **éléments** conserve les éléments sémantiques de l'ordre (MCT et MLT).

Le champ **correction** indique sur quel élément porte la correction (MCT).
Le champ **refus** indique sur quel élément porte l'incohérence (MCT et MLT), il empêche DIAPASON de proposer à nouveau cet élément.

6.6.3 Les opérations sur les mémoires

Les opérations que DIAPASON peut effectuer sur la mémoire sont :
- initialiser la mémoire,
- tester si la mémoire est vide,
- tester si la mémoire est pleine,
- ajouter un élément sémantique en tête,
- vider la mémoire,
- afficher le contenu de la mémoire,
- rechercher un élément sémantique dans la mémoire,
- modifier le champ correction d'un élément mémoire,
- modifier le champ refus d'un élément mémoire,
- enlever l'élément en tête de file,
- prendre l'élément en tête de file.

6.7 LA REPRESENTATION DE L'ETAT DE LA TACHE

6.7.1 Utilité du contexte de la tâche

Le contexte de la tâche s'avère nécessaire pour faire l'interprétation correcte de certaines phrases. Il permet d'éliminer des hypothèses qui conduiraient à une incohérence.

Exemple : L'opérateur veut changer le paramètre temps début pour une image (application DIVA). L'état courant du temps est 6 minutes. Supposons alors que le système de reconnaissance fournisse dans l'ordre de score décroissant les deux hypothèses "temps début 6 mn" et "temps début 10 mn". Le fait de connaître l'état courant de temps début va amener le système à rejeter la première hypothèse et à proposer la seconde qui ne provoque pas d'incohérence. Cette incohérence ne peut pas toujours être résolue. En effet pour la même phrase prononcée, supposons que le système n'ait fourni que la première hypothèse, il n'y a pas d'alternative possible. Néanmoins deux possibilités de réponses du système sont possibles. Premier cas le système accepte l'incohérence et envoie alors un message d'erreur : "Le temps début est déjà à 6 minutes". Second cas le système refuse l'incohérence et pose la question "Quel temps début ?". Le choix d'un cas plutôt que l'autre peut être fonction du taux de reconnaissance global de la phrase ou local des mots, ou de l'origine des éléments, mémoire à court terme ou à long terme.

Le contexte de la tâche va permettre d'un autre coté de proposer des valeurs par défaut.

Exemple : L'ordre temps début peut paramétrer soit une image en temps différé, l'audio ou encore l'extraction demon. Selon l'état de la tâche (sur quel type d'objet travaille l'opérateur), il faudra proposer l'un de ces trois objets.

6.7.2 Etat actuel des travaux

A l'heure actuelle l'état de la tâche est représenté par une variable décomposée en champs et ses modifications sont faites de manière procédurale. Il nécessite donc d'être ré-écrit pour chaque nouvelle application.

Cette thèse pourrait se poursuivre par le développement d'un outil spécifique de représentation et de gestion de la tâche qui serait indépendant de l'application.

CONCLUSION ET PERSPECTIVES

Après un chapitre de généralités où nous avons fait une présentation rapide du dialogue oral homme-machine et où nous avons décrit quelques systèmes, nous avons présenté dans le chapitre deux les objectifs généraux de nos travaux et introduit l'environnement DIAPASON qui est le résultat de nos études. Le chapitre trois a décrit le modèle de dialogue mis en œuvre en introduisant la notion d'ordre, en illustrant les possibilités qu'offre le système à l'aide de nombreux exemples, et finalement en présentant les grammaires syntaxico-sémantiques qui guident la reconnaissance et la compréhension. Le chapitre quatre s'est attaché à décrire l'architecture générale du système de dialogue en insistant sur ses trois principales composantes (le module de gestion de la tâche, le module de gestion du dialogue et le module de gestion de la reconnaissance) et sur les informations qu'ils échangent entre-eux et avec l'extérieur (système(s) de reconnaissance, application). Le chapitre cinq a présenté les deux applications mises en œuvre en collaboration avec THOMSON SINTRA DASM, la commande orale d'une console sonar et celle d'un système d'aide à la classification de signaux acoustiques sous-marins, et les résultats obtenus au cours de ces deux études. Enfin le chapitre 6 a présenté les outils qui composent l'environnement DIAPASON (éditeur de structures syntaxico-sémantiques, compilateur, formateur, gestionnaire de l'historique et de la tâche) qui font de DIAPASON une plate-forme de développement pour la conception de systèmes de dialogue pour des applications cibles de type commande de machine.

Les deux expériences réalisées ont permis de montrer que :

- les taux de reconnaissance du système de reconnaissance de la parole doivent être élevés si l'on veut réaliser une démonstration performante,
- le temps de réponse du système de dialogue ne doit pas dépasser la seconde si l'on ne veut pas que l'utilisateur ait l'impression d'attendre,
- les confirmations visuelles en place s'avèrent efficaces des points de vue de la tâche et de l'ergonomie du système, le regard reste fixé sur la scène où se déroule l'action,
- la parole permet de concentrer en une seule phrase de trois à quatre mots des actions qui demandent un plus grand nombre d'opérations élémentaire par le mode classique,
- la liberté d'élocution introduite par le niveau dialogue est beaucoup plus importante que celle d'un simple système de reconnaissance de la parole, le dialogue permet un éventail de formulations différentes pour une même action,
- les procédures de correction sont très importantes dans les systèmes où il peut y avoir exécutions de commandes erronées qui n'ont pas l'utilisateur

pour origine ; pour qu'elles puissent être mises en œuvre il faut que les taux de reconnaissance soient suffisants,
- la prise en compte de l'entrée vocale doit être faite dès la phase de conception de l'application si l'on veut qu'elle soit efficace du point de vue de la tâche,
- l'introduction d'outils de développement devraient permettre un gain de temps appréciable dans la réalisation de systèmes de dialogue oral.

Les travaux réalisés au cours de cette thèse devraient se poursuivre par :
- la réalisation des derniers outils, en particulier ceux qui s'adressent au niveau reconnaissance de la parole,
- la validation de l'environnement pour une nouvelle application,
- une orientation multimode du système de dialogue afin de tenir compte des études actuelles en matière de désignation,
- l'étude de la représentation de la tâche et de l'outil de saisie correspondant.

BIBLIOGRAPHIE

[ALINAT 1975] Pierre ALINAT 1975, "Etude des phénomènes de la langue française au moyen d'une cochlée artificielle. Application à la reconnaissance de la parole", Revue Technique Thomson CSF vol 7, n°1, mars 1975.

[ALINAT 1987] Pierre ALINAT, Evelyne GALLAIS, Jean-Paul HATON, Jean-Marie PIERREL, Pascal RICHARD "A continuous Speech Dialogue System for the Oral Control of a Sonar Console", Proceedings of IEEE ICASSP-87, 1987.

[ALINAT 1992] Pierre ALINAT, Evelyne GALLAIS Gilles SOUVAY, Jean-Marie PIERREL, "Interaction vocale: démonstration en ASM", Rapport de synthèse final, à paraître.

[ANDRY 1990] F. ANDRY, E. BILANGE, F CHARPENTIER, K. CHOUKRI, M. PONAMLE, S. SOUDOPLATOFF, "Computerised simulation tools for the design of an oral dialogue system", Proc. of ETW Conference, Bruxelles 1990.

[ANGLADE 1990] Yolande ANGLADE, Jean-Claude JUNQUA, "Acoustic Study of Lombard Speech at the Phoneme Level in the case of Isolated Words", Proceeding EUSIPCO, Barcelona (España), September 1990, pages .

[ANGLADE 1991] Yolande ANGLADE, Jean-Marie PIERREL, A. SOUMAN, C. ANDLAUER, "Vers un système d'aide à une standardiste intégrant un dialogue oral homme-machine", Congrès FIRTECH "Univerdustrie 91", Nancy 23-24 Mai 1991.

[ARRONATEGUI 1991] U. ARRONATEGUI, T. HUGUET, J.-P. MACCHION, F. MIEULET, "Reconnaissance de plans pour des systèmes experts qui aident à faire", Actes des journées SEIGE'91, pages 93-116, Bayonne, Septembre 1991.

[ARRONATEGUI 1992] U. ARRONATEGUI, F. AZEMARD, T. HUGUET, J.-P. MACCHION, F. MIEULET, "Interprétation et gestion du dialogue sous LOIR", actes du séminaire dialogue, Dourdan 1992.

[BELLALEM 1991] Nadia BELLALEM, "Interprétation d'informations multimodales. application à la désignation souris", rapport de DEA, CRIN 1991.

[BILANGE 1991] Eric BILANGE, "Modélisation du dialogue oral finalisé personne-machine par une approche structurelle, théorie et réalisation", Thèse de l'Université de Rennes I, Décembre 1991.

[BONIN 1990] Jean-Jacques BONIN, Jean-Marie PIERREL, "Fréquence fondamentale et durée pour la détection de frontières syntagmatiques en parole continue", actes des XVIIIèmes Journées d'Etude sur la Parole, Montréal Mai 1990.

[BONNET 1980] A. BONNET, "Intelligence artificielle: promesses et réalités", Interéditions, Paris 1980.

[BOURGET 1992] M-L. BOURGUET, "ICPplan : dialogue multimodal pour la conception de plans architecturaux, 19ièmes JEP, Bruxelles, 19-22 Mai 1992, pages 369-374.

[BURTON 1976] R. BURTON, "Semantic Grammar. An engineering Technique for Constructing Naturel Language Understanding System", BNN, Report n°3353, Cambridge USA, 1976.

[BUISSON 1976] BUISSON & al, "Deux langages parlés pour un dialogue intelligent entre l'homme et la machine", compte-rendu final convention SESORI n°74-80, 1976, 112 pages.

[CAELEN 1979] Jean CAELEN, "Un modèle d'oreille. Analyse de la parole continue. Reconnaissance phonémique", Thèse d'état, Université Paul Sabatier, Toulouse, 1979.

[CAELEN 1985] Jean CAELEN, "Space/time data-information in the ARIAL project Ear Model", Speech communication 4, 1985, pages 163-180.

[CALLIOPE 1989] CALLIOPE, "La parole et son traitement automatique", Masson, Paris, 1989, 718 pages.

[CARBONNEL 1986] Noëlle CARBONNEL, Jean-Marie PIERREL, "Architecture and knowledge sources of an human-computer oral dialogue system", in proceedings of the NATO workshop, "Structure of Multimodal dialogues including voice", Corsica (France), 1986.

[CARRE 1991] René CARRE, Jean-François DEGREMONT, Maurice GROSS, Jean-Marie PIERREL, Gérard SABAH, "Langage humain et machine", Presses du CNRS, Paris, 1991.

[CHARPENTIER 1992] F. CHARPENTIER, F. GAVIGNET, K. CHOUKRI, F. ANDRY, E.BILANGE, J.-Y. MAGADUR, "Un système de dialogue oral pour une application de réservation téléphonique de billets d'avion", 19ièmes JEP, Bruxelles, 19-22 Mai 1992.

[CHOMSKY 1965] N. CHOMSKY, "Aspect of the Theory of Syntax", MIT Press Cambridge, Mass. USA, 1965 ; traduit en français sous le titre "Aspects de la théorie syntaxique", Editions du Seuil, Paris, 1971.

[CHOMSKY 1968] N. CHOMSKY, G.A. MILLER, "Analyse formelle des langues naturelles", Gauthier-Villars, Paris, 1968.

[CONDOM 1988] Jean-Marie CONDOM, André LOZES, "Commande vocale d'un robot manipulateur", actes des 17ièmes Journées d'Etude sur la Parole, Nancy, 20-22 Septembre 1988, pages 145-148.

[CONDOM 1992] Jean-Marie CONDOM, André LOZES, Michel COURDESSES, "Dialogue multimodal avec un robot manipulateur", actes du séminaire DIALOGUE, Dourdan (91), 15-16 avril 1992.

[COUTAZ 1990] COUTAZ Joëlle, "Interfaces logicielles: conception et réalisation", Paris, DUNOD 1990, 455 pages.

[COUTAZ & CAELEN 1990] Joëlle COUTAZ, Jean CAELEN, "PRC communication homme-machine : opération de recherche concertée interfaces homme-machine multimodales", publication du PRC "communication homme-machine", Juin 1990.

[DEVILLE 1989] Guy DEVILLE, "Modelization of Task-Oriented Utterances in Man-Machine Dialogue Systems", Thèse de doctorat en Lettres, Université d'Anvers, 1989.

[DRUART 1992] Pascal DRUART, "ACE l'éditeur de l'environnement DIAPASON", rapport technique, septembre 1992.

[FILLMORE 1968] C. FILLMORE, "The case for case", Universals in Linguistic Theory", E. Bach, R.T. Harm editors, Rinehart and Winston, New-York.

[FOHR 1986] Dominique FOHR, "APHODEX: un système expert de décodage acoustico-phonétique de la parole continue", Thèse de doctorat de l'Université de Nancy I, 1986.

[FROT 1992] Jean-Philippe FROT, "Le compilateur de l'environnement DIAPASON", rapport de stage de deuxième année, CRIN, septembre 1992.

[GAIFFE 1991] Bertrand GAIFFE, Laurent ROMARY, "Managing Multimodal informations and references in the Multiworks project", Rapport de contrat Multiworks, Décembre 1990.

[GALLAIS 1989] Evelyne GALLAIS Gilles SOUVAY, "Commande vocale de console", Rapport de synthèse final, Août 1989, diffusion restreinte.

[GALLAIS 1990] Evelyne GALLAIS, Pierre ALINAT, Gilles SOUVAY, Jean-Marie PIERREL, "Intégration d'un système de reconnaissance analytique de la parole dans une console sonar: vers un dialogue naturel", Traitement du signal, volume 7 n°4, 1990, pages 367-379.

[GALLAIS 1991] Evelyne GALLAIS Pierre ALINAT, Gilles SOUVAY, Jean-Marie PIERREL, "Intégration d'un système de reconnaissance de la parole dans une

console de sonar: bilan et perspectives", actes du congrès FIRTECH "Univerdustries", Nancy Mars 1992, pages 89-105.

[GRECO 1985] groupe "Dialogue à composante orale du GRECO Communication parlé", "Dialogue oral homme-machine en situation assistée par l'action", Actes du 5ième Congès AFCET "Reconnaissance des formes et intelligence artificielle", Grenoble, 1985, pages 281-296.

[HATON 1990] Jean-Paul HATON, Anne BONNEAU, Dominque FOHR, Yifan GONG, Yves LAPRIE, Jean-Marie PIERREL, "Décodage acoustico-phonétique: problèmes et éléments de solution", Traitement du signal Traitement du signal, volume 7 n°4, 1990.

[HATON 1991] Jean-Paul HATON, Jean-Marie PIERREL, Guy PERENNOU, Jean CAELEN, Jean-Luc GAUVAIN,."Reconnaissance automatique de la parole", Dunod Informatique 1991.

[JOHNSON 1978] Stephan C. JOHNSON "YACC: Yet Another Compiler Compiler", manuel d'utilisation, 1978.

[KLEIN 1990] Joëlle KLEIN, "Contribution à la commande orale d'un robot doté d'un système de vision", Thèse de l'Université de Nancy I, Octobre 1990.

[LAFONT 1983] Sylvie LAFONT, "Etude et mise en œuvre d'outils informatiques pour non-voyants: console braille et commandes vocales", Université de Nancy I, Janvier 1983.

[LONG 1991] Patrick LONG, "Outils pour la compréhension et la gestion de dialogue.", rapport de stage, CRIN, 1991.

[MAGADUR 1992] Jean-Yves MAGADUR, "Représentation structurelle du dialogue oral homme-machine et prédictions", 19ièmes JEP, Bruxelles 19-22 Mai 1992.

[MAIS 1992] Chantal MAIS, "Evaluation de DIVA vocal : ses performances", rapport CERDSM, Août 1992.

[MANGEOL 1988] Bernard MANGEOL "La composante lexicale dans les systèmes de dialogue oral homme-machine du CRIN", Thèse de l'Université de Nancy I, Septembre 1988.

[MARIANI 1978] J. MARIANI, J-S. LIENARD, "Esope 0: un programme de compréhension automatique de la parole", actes du congrès AFCET/IRIA, Chateney-Malabry 1978, pages 169-175.

[MARIANI 1982] J. MARIANI, "ESOPE: un système de compréhension de la parole continue", Thèse de doctorat de Paris VI, 1982.

[PIERREL 1978] Jean-Marie PIERREL "Un système de compréhension automatique du discours continu utilisant des contraintes morphologiques syntaxiqes et sémantiques", RAIRO informatique, Vol 12-2, 1978, pages 83-105.

[PIERREL 1981] Jean-Marie PIERREL, "Etude et mise en œuvre de contraintes linguistiques en compréhension automatique du discours continu", Thèse d'Etat, Université de Nancy I, Mars 1981.

[PIERREL 1987] Jean-Marie PIERREL, "Dialogue oral Homme-Machine", Hermès, Paris, 1987, 239 pages.

[RICHARD 1987] Pascal RICHARD, "Commande vocale d'une console sonar analyse syntaxique, interprétation et gestion du dialogue", rapport de DEA, Université de Nancy I, 1987.

[ROULET 1985] E. ROULET, A. AUCHLIN, J. MOESCHLER, C. RUBATTEL et M. SCHELLING. "L'articulation du discours en français contemporain", Editions Lang, Berne, 1985.

[ROUSSANALY 1988] Azim ROUSSANALY "DIAL: la composante dialogue d'un système de communication orale homme-machine finalisé en langage naturel", Thèse de l'Université de Nancy I, 1988.

[SABAH 1990] Gérard SABAH "CARAMEL : a flexible model for interaction between cognitive processes underlying naturel language understanding", COLING-90 Helsinki.

[SIROUX 1984] J. SIROUX, G. Mercier, R. VIVES, "Dix ans de communication orale et de dialogue entre l'homme et la machine avec KEAL", actes du séminaire "Dialogue homme-machine à composante orale", Nancy 11-12 Octobre 1984, pages 123-135.

[SMAILI 1991] Kamel SMAILI, François CHARPILLET, Jean-Marie PIERREL, Jean-Paul HATON, "Vers une première réalisation d'une machine à dicter destinée au grands vocabulaires", Actes des 2ièmes journées du GRECO PRC "Communication homme-machine", EC2 éditeur, Toulouse, janvier 1991.

[SMAILI 1992] Kamel SMAILI, François CHARPILLET, Jean-Marie PIERREL, Jean-Paul HATON, "La composante lexicale de la machine à dicter: MAUD", Actes du séminaire communication homme-machine", Toulouse, janvier 1992.

[SOUVAY 1990] Gilles SOUVAY, "Spécifications du compilateur", rapport interne CRIN, Octobre 1990.

[SOUVAY 1992] Gilles SOUVAY, Jean-Marie PIERREL, Pierre ALINAT, "Interaction vocale: outils pour la compréhension et la gestion de dialogue", rapport intermédiaire contrat DRET, diffusion restreinte, Février 1992.

[SOWA 1984] J-F SOWA, "Conceptual Structures, information processing in mind and machine", Addison Wesley, Massachusetts 1984.

[TARNAUD 1984] Pierre TARNAUD, "Intégration du dialogue vocal dans un environnement réel complexe", actes du séminaire "Dialogue homme-machine à composante orale", Nancy 11-12 Octobre 1992, pages 404-416.

[TASSET 1991] Daniel TASSET, "DIVA : spécifications de l'interface homme-machine (IHM)", CERDSM, Janvier 1991.

[VILNAT 1992] Anne VILNAT, Lydia NICAUD, "Un système de dialogue homme-machine: STANDIA", Actes du séminaire DIALOGUE, Dourdan (91), 15-16 avril 1992.

[VIVES 1978] R. VIVES, "Réalisation d'un automate de dialogue", Recherches en acoustique,vol III, 1976, pages 245-263.

[WOODS 1970] A. WOODS, "Transition Network grammars for natural language analysis", Communications of ACM, vol 13-10, 1970, pages 561-602.

[ZEEVAT 1987] H. ZEEVAT, E. KLEIN, J. CALDER, "An introduction to Unification Categorial Grammar", Working papers in cognitive Science, V.1, 1987.

UNIVERSITE DE NANCY I

NOM DE L'ETUDIANT : **Monsieur SOUVAY Gilles**

NATURE DE LA THESE : **DOCTORAT DE L'UNIVERSITE DE NANCY I**
en INFORMATIQUE

VU, APPROUVE ET PERMIS D'IMPRIMER

NANCY, le **28 OCT. 1992** n° 494

LE PRESIDENT DE L'UNIVERSITE DE NANCY I

UNIVERSITE NANCY I
LE PRESIDENT

M. BOULANGÉ

RESUME

Les machines et systèmes que nous utilisons dans la vie courante ou dans la vie professionnelle deviennent de plus en plus sophistiqués et leur commande s'avère d'autant plus complexe. Il serait donc utile de trouver un moyen aisé pour leur faire exécuter les tâches que nous leur demandons.

Une idée simple en apparence, consiste à remplacer la méthode classique, appui sur une série de boutons, choix dans un menu, par une phrase prononcée dans un microphone relié à la machine qui doit être commandée. La parole est un mode de communication rapide et concis, l'homme l'emploie depuis des millénaires. L'utilisateur d'un tel système joue un rôle plus actif et jouit d'une plus grande liberté d'action.

Seulement dans la pratique cela s'avère plus complexe qu'on ne l'imagine. Introduire une composante orale ne consiste pas seulement à greffer une boite noire qui en entrée reçoit de la parole et en sortie renvoie la phrase prononcée, ce problème est déjà relativement complexe à résoudre. La parole porte en elle un éventail de formulations différentes pour un même sens, des possibilités de mauvaises compréhensions, de mauvaises interprétations, d'ambiguïtés, d'imprécisions qui s'ajoutent aux nouvelles erreurs commises par l'opérateur du fait qu'il est moins astreint à suivre un schéma imposé par la machine.

Il faut alors réaliser une interface entre le système de reconnaissance vocale et la machine ou l'application à diriger : le système d'interprétation et de gestion du dialogue. De plus si l'on veut pouvoir réutiliser ce système pour une nouvelle application, il faut définir des outils d'aide à la mise en place des connaissances spécifiques de l'application. L'ensemble formera un environnement de travail, qui dans notre cas tire son nom du système de dialogue : l'environnement DIAPASON.

MOTS-CLES

dialogue oral homme-machine finalisé, commande orale de console, reconnaissance analytique, outils de gestion de dialogue, réseau syntaxico-sémantique.