Université Stendhal - Grenoble III
Equipe Cristal-Gresec

Gestion de conflits dans une architecture multi-agents d'analyse automatique de textes

Thèse de Doctorat en Informatique et Communication
Spécialité Informatique

présentée par

Karine WARREN

Soutenue publiquement le 14 Janvier 1998 devant le jury composé de :

Mr Jacques ROUAULT	Directeur de thèse
Mr Gérard LIGOZAT	Rapporteur
Mr Henri MADEC	Rapporteur
Mr Jacques COURTIN	Président
Mr Yves DEMAZEAU	Examinateur
Mme Marie-Hélène STEFANINI	Examinatrice

REMERCIEMENTS

Je remercie Mr Jacques ROUAULT, Professeur en Sciences de l'Information et de la Communication à l'Université Stendhal de Grenoble, pour m'avoir accueillie au sein de l'équipe de recherche CRISTAL-GRESEC, pour m'avoir fait profiter de ses nombreuses compétences et pour la confiance qu'il m'a toujours témoignée.

Je tiens à exprimer ma gratitude à Mr Gérard LIGOZAT, Professeur en Informatique à l'Université d'Orsay de Paris, d'avoir consacré spontanément une partie de son temps pour rapporter sur ce travail et de s'être intéressé à la problématique d'Intelligence Artificielle Distribuée.

Je remercie également Mr Henri MADEC, Docteur d'Etat, Maître de Conférences en Linguistique à l'Université de Franche-Comté, d'avoir accepté avec enthousiasme d'être un juge attentif de mes travaux de recherche et d'avoir fait preuve d'une grande disponibilité.

Je suis honorée de la participation au jury de deux spécialistes en informatique.
Je remercie Mr Jacques COURTIN, Professeur en Informatique à l'Université Pierre Mendès-France de Grenoble, d'avoir accepté d'examiner mon travail, m'apportant ainsi sa grande expérience en Traitement Automatique de la Langue.
Je remercie également Mr Yves DEMAZEAU, Chargé de recherche au laboratoire LEIBNIZ, Responsable du groupe MAGMA, de me faire bénéficier de ses compétences en multi-agents.

Ce travail doit beaucoup à Mme Marie-Hélène STEFANINI, Maître de Conférences en Informatique à l'Université Stendhal de Grenoble, qui est à l'origine de cette recherche. Je la remercie d'avoir suivi ce travail en me prodiguant des conseils toujours avisés lors de ses multiples relectures. Son optimisme et sa confiance sans cesse renouvelée ont fortement contribué à l'aboutissement de ce travail.

Je remercie les membres du GRESEC, et plus particulièrement l'équipe CRISTAL, dont l'accueil et la gentillesse m'ont permis d'effectuer ce travail dans d'agréables conditions.
Je tiens à exprimer ma gratitude aux personnes avec qui j'ai collaboré : Georges Antoniadis, Laurence Balicco, Viviane Clavier, Christel Froissart, Geneviève Lallich-Boidin et Claude Ponton.
Je souhaite manifester toute ma reconnaissance à Céline, Laurence, Lahcène, Laurent, toujours solidaires dans les moments difficiles et toujours enclins à "positiver".

J'associe mes amis de longue date qui ont suivi avec curiosité ces années particulières, notamment Delphine, Valérie, Virginie, Christophe, Olivier, et, surtout Nathalie qui a "subi", de plus près, mes états d'âme.

Enfin, un grand merci à mes parents pour leur confiance et leur soutien financier indispensables à la réalisation de ce travail, à ma sœur qui était déjà "passée par là", à Jérome pour sa présence dans tous les sens du terme, et à mon chat qui n'a toujours rien compris…

PLAN

INTRODUCTION GENERALE 13

PARTIE I :

PROBLEMATIQUE LINGUISTIQUE ET INFORMATIQUE.................................. 15

CHAPITRE I : PRESENTATION DE L'ARCHITECTURE TALISMAN 17

I) Problématique de l'analyse séquentielle du français écrit.. 17
  I-1) Présentation de l'analyse séquentielle du français écrit : le système CRISTAL 17
    I-1-1) Le niveau morphologique 18
      I-1-1-1) Différentes approches possibles 18
      I-1-1-2) La morphologie dans CRISTAL 19
    I-1-2) Le niveau syntaxique.. 21
      I-1-2-1) Différentes approches possibles 21
      I-1-2-2) La syntaxe dans CRISTAL.. 22
  I-2) Limites d'une architecture séquentielle : le système CRISTAL.. 23
    I-2-1) Manque d'interaction entre les niveaux de représentation 23
    I-2-2) Manque de distribution des connaissances et du contrôle 24
    I-2-3) Système difficilement évolutif.. 24
    I-2-4) Bilan.. 25

II) Etude de systèmes distribués de TAL 25
  II-1) Intelligence Artificielle Distribuée et Systèmes Multi-Agents.. 26
    II-1-1) L'Intelligence Artificielle Distribuée... 26
    II-1-2) Les Systèmes Multi-Agents... 27
      II-1-2-1) Le type d'agents 28
      II-1-2-2) Le type de comportement des agents... 28
      II-1-2-3) Les modes de communication.. 29
      II-1-2-4) Les types de contrôle.. 30
      II-1-2-5) Les structures d'implantation 30
      II-1-2-6) La résolution dans un Système Multi-Agents 31
  II-2) Systèmes distribués de Traitements Automatique de la Langue 31
    II-2-1) Présentation générale de systèmes distribués de TAL.. 32
      II-2-1-1) Le système HEARSEY II... 32
      II-2-1-2) Le système HELENE 32
      II-2-1-3) Le système CARAMEL 33
      II-2-1-4) Le système DIALECT 2... 33
      II-2-1-5) Le système TALISMAN 34
      II-2-1-6) Le système MICRO.. 34
      II-2-1-7) Le système ETCvérif.. 34
      II-2-1-8) Le système de Pompidor et Vergnaud 35
      II-2-1-9) Le système CELINE... 35
    II-2-2) Comparaison de systèmes distribués de TAL.. 36
      II-2-2-1) Comparaison en Traitement Automatique de la Langue 36
      II-2-2-2) Comparaison en SMA 37
      II-2-2-3) Bilan des comparaisons.. 39

III) Problématique de l'architecture TALISMAN.. 40
  III-1) Présentation détaillée de TALISMAN 40
    III-1-1) Architecture générale... 40
    III-1-2) Description des agents.. 41

III-2) Problématique linguistique 44
    III-2-1) Validation de l'architecture en grandeur réelle 45
    III-2-2) Développement d'agents 45
III-3) Problématique informatique 46

CHAPITRE II : GESTION DES CONFLITS LINGUISTIQUES 49

I) Les conflits linguistiques........ 48
I-1) Définition des conflits linguistiques........ 48
I-2) Etude des conflits linguistiques à résoudre 50
    I-2-1) Conflits résolus dans TALISMAN 50
    I-2-2) Améliorations envisagées........ 51
        I-2-2-1) Amélioration de la désambiguïsation........ 51
        I-2-2-2) Traitement de phénomènes linguistiques complexes 52

II) Méthodes classiques de résolution des conflits linguistiques 52
II-1) Stratégies classiques d'analyse 52
    II-1-1) Désambiguïsation morphologique........ 52
    II-1-2) Stratégies d'analyse syntaxique 53
    II-1-3) Stratégies de traitement d'ambiguïtés........ 54
    II-1-4) Stratégies de résolution de problèmes........ 55
    II-1-5) Exemples d'algorithmes d'analyse........ 56
II-2) Algorithmes vs heuristiques........ 56
II-3) Architecture séquentielle vs distribuée........ 57
II-4) Choix de résolution informatique de conflits linguistiques 57
    II-4-1) L'existant 58
    II-4-2) Améliorations envisagées 58
        II-4-2-1) Utilisation d'une forme de contrôle........ 58
        II-4-2-2) Protocoles de communication 60

III) Synthèse de protocoles de communication........ 61
III-1) Définition et objectifs d'un protocole de communication........ 61
    III-1-1) Définition 61
    III-1-2) Objectifs........ 62
        III-1-2-1) Echange de connaissances entre agents 62
        III-1-2-2) Traitement des conflits........ 62
III-2) Synthèse de protocoles de communication 65
    III-2-1) Présentation générale de protocoles de communication 66
        III-2-1-1) Protocoles d'échanges de Campbell et D'Inverno........ 66
        III-2-1-2) Acquisition de connaissances de Sian........ 66
        III-2-1-3) Protocole d'apprentissage MOSCA........ 66
        III-2-1-4) Smith et Davis : Contract net protocol........ 67
        III-2-1-5) Protocole hiérarchique de Durfee et Montgomery 67
        III-2-1-6) Protocole de Kreifelts et Von Martial 68
        III-2-1-7) Un protocole adapté à la négociation : Chang et Woo 68
        III-2-1-8) Lander et Lesser : recherche négociée 68
        III-2-1-9) Protocoles génériques de Burmeister, Haddadi, Sundermeyer 69
    III-2-2) Comparaison des protocoles étudiés........ 69
        III-2-2-1) Critères de comparaison........ 69
        III-2-2-2) Comparaison des protocoles........ 70
        III-2-2-3) Conclusion........ 72
III-3) Elaboration de protocoles linguistiques 73
    III-3-1) Choix de critères pour l'élaboration de protocoles linguistiques........ 73
    III-3-2) Messages et langage de communication dans TALISMAN II 74
        III-3-2-1) Communication par envois de messages........ 74
        III-3-2-2) Langage de communication........ 75
    III-3-3) Présentation des protocoles linguistiques 77
        III-3-3-1) Protocole d'affirmation 77
        III-3-3-2) Protocole de demande d'information........ 78

III-3-3-3) Protocole de demande de coopération 78

PARTIE II :
RESOLUTION DE SOLUTIONS MULTIPLES EN MORPHO-SYNTAXE 83

CHAPITRE III : RESOLUTION INTERNE AUX AGENTS MORPH ET SYNT 85

I) Etude du corpus 84

II) Agent MORPH 88
- II-1) Analyse morphologique dans TALISMAN II 88
- II-2) Ambiguïtés relevant du niveau morphologique 89
  - II-2-1) Ambiguïtés d'origine typographique 89
  - II-2-2) Ambiguïtés d'origine lexicale 90
  - II-2-3) Ambiguïtés d'origine morphologique flexionnelle 90
  - II-2-4) Ambiguïtés d'origine morphologique dérivationnelle 91
- II-3) Présentation de l'agent Morph 92
- II-4) La désambiguïsation linguistique 93
  - II-4-1) Règles de choix de catégories différentes concurrentes 93
  - II-4-2) Règles sur certaines formes 95
  - II-4-3) Règles sur le choix de variables avec catégories identiques 95
  - II-4-4) Règles sur certains critères linguistiques particuliers 96
- II-5) Bilan 96
  - II-5-1) Apports 96
  - II-5-2) Perspectives 97
    - II-5-2-1) Ambiguïtés d'origine typographique 97
    - II-5-2-2) Ambiguïtés d'origine morphologique dérivationnelle 98

III) Agent SYNT 98
- III-1) Analyse syntaxique dans TALISMAN II 98
- III-2) Ambiguïtés relevant du niveau syntaxique 99
  - III-2-1) Ambiguïtés liées à la grammaire du syntagme 100
  - III-2-2) Ambiguïtés liées à la grammaire de la proposition 100
  - III-2-3) Ambiguïtés liées à la grammaire de la phrase 100
- III-3) Présentation de l'agent Synt 101
- III-4) Les différentes compétences de Synt 102
  - III-4-1) Analyse en constituants 102
    - III-4-1-1) Grammaire syntaxique en constituants de la proposition 102
    - III-4-1-2) Grammaires satellites et méta-grammaires 104
  - III-4-2) Analyse fonctionnelle 106
    - III-4-2-1) Présentation générale 106
    - III-4-2-2) Exemples d'analyse fonctionnelle 106
  - III-4-3) Stratégies syntaxiques de désambiguïsation 108
    - III-4-3-1) Stratégies de choix du verbe de la proposition 108
    - III-4-3-2) Elimination de suites de catégories impossibles 109
    - III-4-3-3) Vérification d'accords 109
    - III-4-3-4) Stratégies de longue distance 110
    - III-4-3-5) Elimination des solutions arborescentes identiques 110
- III-5) Bilan 111
  - III-5-1) Apports 111
  - III-5-2) Perspectives 112

CHAPITRE IV : RESOLUTION COOPERATIVE 113

I) Exemples d'analyse coopérative dans TALISMAN II 113
- I-1) Un cas simple 113
- I-2) Analyse d'une phrase comportant des verbes ambigus 116

II) Stratégie de traitement de la coordination 120
  II-1) Cas particulier des phénomènes linguistiques complexes 120
  II-2) Problématique de la coordination 121
  II-3) Les agents Segm et Coord 123
    II-3-1) Agent Segm 123
    II-3-2) Agent Coord 124
  II-4) Exemples de traitement d'un coordonnant 125
    II-4-1) Traitement d'une coordination inter-propositions 125
    II-4-2) Traitement d'une coordination intra-proposition 130

III) Traitement d'un mot inconnu 134

IV) Vers de nouvelles résolutions coopératives 138
  IV-1) Coopération entre Regul et Morph 138
    IV-1-1) Ambiguïtés d'origine typographique 138
    IV-1-2) Ambiguïtés d'origine morphologique dérivationnelle 139
  IV-2) Coopérations avec Sema et/ou Pragm 139
  IV-3) D'autres coopérations de Coord 140

PARTIE III :
MISE EN ŒUVRE 143

CHAPITRE V : IMPLANTATION 145

I) Description externe du système 145
  I-1) Mise en œuvre de TALISMAN II 145
  I-2) Système de fenêtrage 146

II) Description interne du système 147
  II-1) Les agents 147
    II-1-1) Agent Morph 148
      II-1-1-1) Présentation générale 148
      II-1-1-2) Prédicats de désambiguïsation morphologique 149
    II-1-2) Agent Synt 149
      II-1-2-1) Présentation générale 149
      II-2-2-2) Algorithme en profondeur 151
      II-2-2-3) Prédicats de la grammaire en constituants 153
      II-2-2-4) Prédicats de la grammaire fonctionnelle 154
      II-2-2-5) Prédicats de désambiguïsation syntaxique 156
    II-1-3) Agent Segm 156
    II-1-4) Agent Coord 157
  II-2) La communication avec un contrôle décentralisé 159
    II-2-1) Le format des messages 159
    II-2-2) Les protocoles de communication 159
    II-2-3) Envoi de message 160
    II-2-4) Réception de message 161

III) Bilan 162
  III-1) Exécution du système 163
  III-2) Etude des résultats 171

CHAPITRE VI : PERSPECTIVES 173

I) Application à l'indexation automatique 174
  I-1) L'indexation automatique 174
    I-1-1) Présentation générale 175
    I-1-2) Présentation des travaux de l'équipe 175

I-2) Vers une intégration d'un agent INDEX dans TALISMAN II 176
  I-2-1) Description interne de l'agent INDEX 176
  I-2-2) Interactions avec les autres agents 177

II) Vers une validation de TALISMAN II 178
II-1) En linguistique : intégration de la morphologie dérivationnelle 178
II-2) En informatique : utilisation des protocoles linguistiques pour la génération 180
II-3) En TAL : Intégration d'autres phénomènes linguistiques 184
  II-3-1) Transformations 184
    II-3-1-1) Présentation 184
    II-3-1-2) Résolution 186
  II-3-2) Reconstruction d'ellipses 189
    II-3-2-1) Présentation 189
    II-3-2-2) Résolution 190

III) Conclusion 193
III-1) Bilan 194
  III-1-1) Bilan pratique 194
  III-1-2) Bilan théorique 195
III-2) Perspectives 195
  III-2-1) Modélisation d'autres phénomènes linguistiques complexes 196
    III-2-1-1) Traitement d'anaphores 196
    III-2-1-2) Portée de la négation 199
  III-2-2) Ambiguïtés des niveaux sémantique et pragmatique 203
    III-2-2-1) Ambiguïtés du niveau sémantique 203
    III-2-2-2) Ambiguïtés du niveau pragmatique 206

CONCLUSION GENERALE 208

BIBLIOGRAPHIE 210

ANNEXES 225

LES FIGURES

Figure 1 : L'architecture séquentielle du système d'analyse CRISTAL 18
Figure 2 : Agent cognitif communiquant avec son environnement 28
Figure 3 : Architecture du type "tableau noir" 30
Figure 4 : Plusieurs solutions produites par un agent 49
Figure 5 : Solutions concurrentes de différents agents 49
Figure 6 : Interpréteur de messages 59
Figure 7 : Utilisation des lois 60
Figure 8 : Protocole d'affirmation 78
Figure 9 : Protocole de demande d'information 78
Figure 10 : Protocole de demande de coopération 79
Figure 11 : Analyse d'une phrase "simple" 114
Figure 12 : Analyse d'une phrase comportant des verbes ambigus 116
Figure 13 : Analyse d'une coordination inter-propositions 126
Figure 14 : Analyse d'une coordination intra-proposition 131
Figure 15 : Analyse d'une phrase comportant une forme inconnue 135
Figure 16 : Système de fenêtrage 147
Figure 17 : Analyse d'une phrase complexe - Architecture 164
Figure 18 : Analyse d'une phrase complexe - Agent Regul 165
Figure 19 : Analyse d'une phrase complexe - Agent Transf 165
Figure 20 : Analyse d'une phrase complexe - Agent Morph 166
Figure 21 : Analyse d'une phrase complexe - Agent Segm 166
Figure 22 : Analyse d'une phrase complexe - Agent Coord 167
Figure 23 : Analyse d'une phrase complexe - Agent Synt 167
Figure 24 : Analyse d'une phrase complexe - Agent Synt 168
Figure 25 : Analyse d'une phrase complexe - Agent Synt 168
Figure 26 : Analyse d'une phrase complexe - Agent Synt 169
Figure 27 : Analyse d'une phrase complexe - Agent Coord 169
Figure 28 : Analyse d'une phrase complexe - Agent Synt 170
Figure 29 : Analyse d'une phrase complexe - Agent Morph 170
Figure 30 : Architecture du système distribué de génération 182
Figure 31 : Protocole d'interaction de Sian 237
Figure 32 : Protocole d'apprentissage MOSCA 239
Figure 33 : Protocole de réseau de contrats 240
Figure 34 : Protocole de Kreifelts et Von Martial 243
Figure 35 : Protocole de Chang et Woo (Phase 2) 245
Figure 36 : Protocole de Chang et Woo (Phase 4) 245
Figure 37 : Protocole de Lander et Lesser 247
Figure 38 : Protocoles primitifs de Burmeister, Haddadi et Sundermeyer 248
Figure 39 : Exemples de protocoles de Burmeister, Haddadi et Sundermeyer 249

LES TABLEAUX

Tableau 1 : Comparaison des systèmes d'un point de vue TAL 37
Tableau 2 : Comparaison des systèmes d'un point de vue multi-agents 38
Tableau 3 : Comparaison de protocoles de communication 71
Tableau 4 : Combinaisons possibles entre Type et Force 77
Tableau 5 : Formes fréquentes du corpus 85
Tableau 6 : Catégories morphologiques des formes les plus fréquentes du corpus 86
Tableau 7 : Catégories ambiguës parmi les formes les plus fréquentes 87
Tableau 8 : Synthèse des formes du corpus avant désambiguïsation 88
Tableau 9 : Exemples de variables de catégories morphologiques 91
Tableau 10 : Synthèse des formes du corpus après une première levée d'ambiguïtés 97
Tableau 11 : Résultats de l'analyse syntaxique de propositions "simples" 172

INTRODUCTION GENERALE

CADRE DE TRAVAIL

Nous avons réalisé nos travaux de recherche au sein de l'équipe CRISTAL-GRESEC (Connaissances, Recherche d'information, Interfaces, Systèmes de Traitement Automatique des Langues - Groupe de REcherche Sur les Enjeux de la Communication).
Le point de départ de nos travaux est le prototype TALISMAN : une architecture multi-agents d'analyse morpho-syntaxique du français écrit, proposée par Marie-Hélène Stéfanini. Ses travaux ont permis la définition de l'architecture générale du système et du modèle d'agent. Les fondements linguistiques de ce système sont les suivants :

- il travaille sur la langue en général,
- il permet la prise en compte de phénomènes linguistiques complexes (tels la coordination, la négation, l'ellipse,…),
- il est indépendant de toute application,
- il a pour objectif de fournir pour chaque phrase, la ou les analyses morpho-syntaxiques attestées par la langue dans un contexte donné.

Nous nous sommes inspiré de cette architecture pour développer un système en grandeur réelle permettant la gestion, voire la résolution, de solutions multiples (ou conflits linguistiques) : TALISMAN II. Ces travaux se situent à l'intersection de deux disciplines de recherche :

- *La linguistique* au travers de l'étude d'ambiguïtés, notamment des ambiguïtés relevant des niveaux d'analyse morphologique et syntaxique. Les ambiguïtés inhérentes à tout système général de traitement automatique de la langue, posent de nombreux problèmes nécessitant la mise en œuvre de traitements appropriés.
- *L'informatique*, et plus particulièrement, les systèmes multi-agents. Nous nous sommes intéressé à la communication (primordiale dans de tels systèmes) dans la mesure où elle doit permettre aux agents linguistiques de disposer de la bonne information au bon moment et plus généralement, elle doit permettre la gestion des conflits.

PLAN DU MEMOIRE

La première partie nous permet d'exposer la problématique linguistique et informatique de nos travaux.
Dans le premier chapitre, une étude de la problématique du Traitement Automatique de la Langue, nous amène à comparer différents systèmes de ce domaine utilisant une architecture distribuée. Nous présentons alors les principes de base sur lesquels repose TALISMAN ainsi que la problématique de cette architecture. Ceci, nous conduit à introduire les nouvelles orientations envisagées aux niveaux linguistique et informatique.
Le deuxième chapitre nous permet de définir les conflits linguistiques et de proposer des méthodes informatiques possibles permettant leur gestion, voire leur résolution. Une étude comparative de protocoles en Intelligence Artificielle Distribuée, nous permet d'introduire

des critères de choix pour l'élaboration de protocoles ; nous développons ensuite, des protocoles plus adaptés à la gestion de conflits linguistiques.

La seconde partie nous permet d'illustrer la gestion de conflits linguistiques par la désambiguïsation morpho-syntaxique réalisée dans TALISMAN II sur un corpus journalistique.
Le troisième chapitre montre comment les agents Morph(ologie) et Synt(axe) résolvent (en interne) certaines solutions multiples provoquées par les ambiguïtés relevant des niveaux d'analyse morphologique et syntaxique.
Dans le quatrième chapitre, nous présentons des exemples de résolution coopérative. Ce chapitre permet de comprendre la complexité d'un système général d'analyse automatique d'une langue, notamment lors de la prise en compte de phénomènes linguistiques courants (la présence d'une forme inconnue ou d'un coordonnant).

Nous développons les réalisations effectuées dans *la troisième partie* de ce mémoire.
Le cinquième chapitre nous permet de décrire l'implantation de TALISMAN II sous Prolog et d'étudier les résultats fournis par le système.
Dans le sixième chapitre, nous effectuons une évaluation de TALISMAN II au regard d'autres disciplines de recherche : nous proposons une application du système à l'indexation automatique de documents techniques volumineux, et nous développons plusieurs validations possibles du système (en informatique, en linguistique, et en traitement automatique de la langue). Nous présentons ensuite, certaines perspectives de recherche nous semblant importantes pour TALISMAN II.

PARTIE I :

PROBLEMATIQUE LINGUISTIQUE ET INFORMATIQUE

CHAPITRE I : PRESENTATION DE L'ARCHITECTURE TALISMAN

Dans ce chapitre, nous présentons la problématique linguistique et informatique d'une architecture distribuée d'un système de Traitement Automatique de la Langue (TAL) tel que TALISMAN.
Dans un premier temps, nous exposons les limites d'une architecture séquentielle pour l'analyse automatique de français écrit. Ceci nous conduit à étudier les systèmes existants de TAL utilisant une architecture distribuée. Nous effectuons ensuite, une comparaison de ces systèmes en fonction de leurs spécificités linguistiques et informatiques. Enfin, nous présentons l'architecture TALISMAN et sa problématique, puis nous exposons les améliorations linguistiques et informatiques nécessaires.

I) Problématique de l'analyse séquentielle du français écrit

Certains auteurs construisent des systèmes d'analyse utilisant des théories linguistiques mixant, en particulier, les niveaux morphologique et syntaxique (grammaires GPSG [Gazdar & al., 85], LFG [Kaplan & Bresnan, 82],…). Une étude comparative de ces analyseurs utilisant un formalisme unique (où les différents niveaux de représentation sont confondus) est exposée dans [Stéfanini, 93]. En ce qui nous concerne, nous nous limitons à l'étude de la problématique de systèmes séquentiels utilisant plusieurs formalismes, chacun spécifiant de façon séparée un niveau de représentation donné (morphologie, syntaxe,…).
Dans cette partie, nous présentons donc les aspects morphologiques et syntaxiques d'un système séquentiel d'analyse du français écrit (tel que le système CRISTAL). Puis, nous exposons les limites d'une architecture séquentielle pour un tel système.

I-1) Présentation de l'analyse séquentielle du français écrit : le système CRISTAL

Les systèmes d'analyse automatique du français écrit fondés sur une architecture séquentielle (dits "classiques") comportent en général quatre niveaux conceptuels de représentation : la morphologie, la syntaxe, la sémantique et la pragmatique.
Nous nous sommes intéressé au système séquentiel CRISTAL (cf. Figure 1), dans lequel, actuellement, seuls les niveaux morphologique et syntaxique sont implémentés. Ce système a été élaboré par l'équipe de traitement automatique des langues, sous la direction de J. Rouault et avec la collaboration de A. Berrendonner (cf. [Berrendonner, 83] et [Lallich & al., 90]). Les niveaux sémantique et pragmatique sont en cours d'élaboration [Manès-Gallo & Rouault, 92, 95, 96, 97], [Berrendonner & al., 92]. Une première spécification du niveau pragmatique a été prototypée dans le cadre d'une thèse [Fredj, 93].

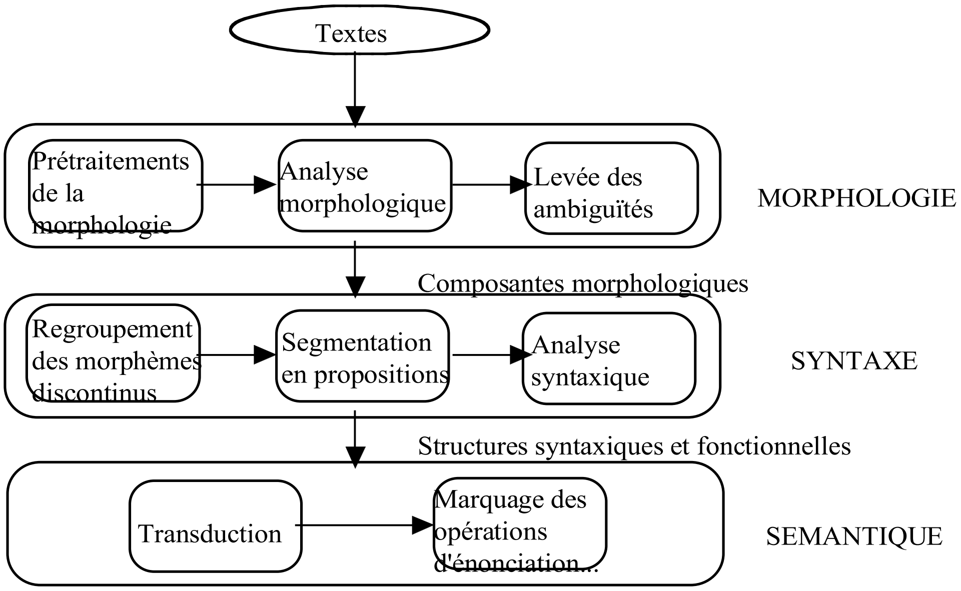

Textes
Prétraitements de la morphologie
Analyse morphologique
Levée des ambiguïtés
MORPHOLOGIE
Composantes morphologiques
Regroupement des morphèmes discontinus
Segmentation en propositions
Analyse syntaxique
SYNTAXE
Structures syntaxiques et fonctionnelles
Transduction
Marquage des opérations d'énonciation...
SEMANTIQUE

Figure 1 : L'architecture séquentielle du système d'analyse CRISTAL

Dans notre étude, nous nous attachons principalement aux aspects morphologiques et syntaxiques de l'analyse du français écrit.

I-1-1) Le niveau morphologique

L'analyse morphologique a pour objectif d'associer à chaque forme une ou plusieurs entrée(s) lexicale(s), et une ou plusieurs catégorie(s) (avec éventuellement, les valeurs de leurs variables) indépendamment, ou non, du contexte.

I-1-1-1) Différentes approches possibles

Différentes approches sont possibles pour l'analyse morphologique.

- **Sous-langage vs langue en général**

Nous distinguons, tout d'abord, les analyseurs qui travaillent dans le cadre d'un sous-langage (i.e. un langage propre à un domaine particulier) [Zweigenbaum & al., 89] en opposition à ceux dont l'ambition est de traiter la langue en général [Mekaouche & Bassano, 91]. Un analyseur rendant compte d'un sous-langage particulier est évidemment plus facile à réaliser dans la mesure où les ambiguïtés de la langue sont alors restreintes. Cependant, ce type de système ne peut être utilisé que pour des applications spécifiques. Nous avons choisi de développer un système d'analyse du français écrit générique i.e. utilisable pour différents domaines d'application. Nous travaillons donc sur la langue en général, ce qui entraîne une complexité du système et une multiplication des solutions concurrentes.

- **Type d'analyseur morphologique**

L'analyse morphologique peut être abordée de différentes manières : il est possible d'utiliser un analyseur morphologique flexionnel [Lallich & al., 90], couplé ou non avec une analyse dérivationnelle [Clavier, 96a] ; notons l'engouement actuel pour l'utilisation d'une "morphologie à deux niveaux"[1] [Koskenniemi, 83]. Une étude comparative des différents types d'analyseur morphologique est présentée dans [Clavier, 96a]. Nous n'utilisons, pour l'instant, qu'une analyse morphologique flexionnelle comprenant une grammaire régulière ; mais l'intégration de la morphologie dérivationnelle est en cours d'élaboration [Clavier & al., 96].

• Type de désambiguïsation

Enfin, nous distinguerons deux types de méthodes possibles de désambiguïsation : une première méthode qui met en œuvre des outils statistiques [Bouchaffra & al., 93] ; et une seconde stratégie qui s'appuie sur des critères purement linguistiques [Lebourg, 88]. Le système séquentiel CRISTAL utilise une méthode statistique.

I-1-1-2) La morphologie dans CRISTAL

Le but de la morphologie est d'associer à une forme, la ou les bonnes entrées lexicales, et la ou les bonnes catégories morphologiques (c'est-à-dire attestées par la langue) ainsi que les valeurs de leurs variables. La classification en catégories et variables utilisée est issue des travaux de [Berrendonner, 83].

Par exemple, la forme "la" peut être analysée morphologiquement comme :

Déterminant (féminin, singulier,…)	D	FEM SNG…
Préverbal (féminin, singulier, 3ème personne,…)	Y	FEM SNG PE3…
Nominal (masculin, singulier, nom,…)	F	MAS SNG NOM…

• Prétraitements

Certains traitements préalables au texte ASCII standard sont nécessaires afin de préparer l'analyse morpho-syntaxique. Ils utilisent l'algorithme d'Aho et Corasick [Aho & Corasick, 75] et permettent une certaine normalisation des caractères et un découpage du texte en formes : cette normalisation correspond aux *prétraitements*. Les prétraitements sont de deux types [Lallich et al., 90] :

1. Prétraitements morpho-graphiques :

- Cas des ponctuations :

Les ponctuations sont considérées comme des formes à part entière puisqu'elles sont interprétables dans l'analyse. Un espace est donc ajouté entre une ponctuation et la forme qui la précède (sauf, bien entendu, s'il existe déjà comme, par exemple, avant un point virgule). Certaines ponctuations non ambiguës (comme le point d'interrogation, les parenthèses,…) ne posent pas de problèmes. Contrairement à d'autres ponctuations (comme le point, l'apostrophe,…) qui peuvent jouer différents rôles.

Exemple : "Aujourd'hui, j'ai rencontré M. Dupond du C.N.R.S. au stade."

Dans "M.", le point sert à marquer une *abréviation*, pour "C.N.R.S.", les points sont utilisés dans un *sigle*,

---

[1] La morphologie à deux niveaux repose sur l'idée que l'on peut mettre en correspondance de manière statique une forme de surface avec une forme lexicale au moyen de règles bi-directionnelles qui énoncent les conditions nécessaires et/ou suffisantes de mise en correspondance entre les deux formes.

et dans "stade.", le point marque la *fin de phrase*.

- Cas des majuscules :

Une majuscule représente, soit une initiale de phrase, soit une initiale de nom propre (ou une abréviation). La substitution automatique d'une majuscule en début de phrase en une minuscule n'est pas toujours possible (surtout avec le problème du rétablissement de l'accentuation des voyelles en majuscules) comme dans les exemples suivants :

"Paris est la capitale de la France"

"A Paris, la circulation est beaucoup plus difficile"

- Traitement des accents :

Pour des raisons de portabilité, le texte est en ASCII Standard ; les caractères accentués sont remplacés par un jeu de deux caractères selon un ensemble de règles.

Exemples : è → e` é → e' ö → :o â → a^ ç → c_

- Traitement des consonnes géminées et extensions :

Les règles permettant ce traitement facilitent l'analyse morphologique flexionnelle.

Exemples :

ss → s* (gros, gros*e) ch → c* (blanc, blanc*e) v → f* (f*if, f*if*e)

u:e → u*e (aigu, aigu*e) gue → g*e (long, long*e)

2. <u>Prétraitements morpho-syntaxiques</u> :

- Eclatement d'amalgames orthographiques :

La langue française comporte des amalgames c'est-à-dire des formes résultant du regroupement de plusieurs formes ayant chacune un rôle syntaxique propre. L'éclatement de ces amalgames permet donc de faciliter l'analyse morpho-syntaxique du texte.

Exemples :

"*au*" devient "*à*" (P préposition) + "*le*" (D déterminant)

"*dont*" devient "*que*" (Q conjonction de subordination) + "*de*" (P) + "%" (F nom)

- Regroupement des formes invariables :

Le regroupement des formes invariables facilite l'analyse ; cependant il ne peut être systématique pour de nombreuses suites de formes comme "*grâce à*" *("il a trouvé la grâce à Lourdes*" vs "*il l'a trouvé grâce à lui*").

Exemples : "au fur et à mesure", "par conséquent",…

- Traitement de l'apostrophe :

Un espace est ajouté après chaque apostrophe, sauf dans certains cas (comme "*aujourd'hui*", "*presqu'île*",…) ; puis, les formes élidées non ambiguës sont remplacées à l'aide de règles (la forme "*l'*" ne fait l'objet d'aucune règle car elle peut aussi bien représenter "*le*" que "*la*").

Exemples : d' → de l'on → on j' → je

- Cas du trait d'union :

Le trait d'union peut être supprimé dans certains cas, mais bien évidemment, il doit être conservé pour les mots composés avec trait d'union. Ainsi, il ne doit pas être éliminé de certains nominaux comme "*morpho-syntaxique*", "*rendez-vous*",… Il est possible d'utiliser certaines règles de suppression du trait d'union.

Exemples : "-on" → " on" "-ci" → " ci"
"-la" → " la" "-tu" → " tu"

• Analyseur morphologique flexionnel

Une fois le texte prétraité, l'analyseur morphologique flexionnel, qui est un automate fini, associe une (ou plusieurs) catégorie(s) ainsi que les valeurs de leurs variables à chaque forme. Par exemple, la forme "*le*" sera reconnue comme pouvant être un préverbal Y ou un déterminant D (masculin, singulier).

• Désambiguïsation

Enfin, il existe un module de "levée d'ambiguïtés" permettant une réduction des catégories morphologiques possibles trouvées par l'analyseur morphologique. Ce module élimine certaines solutions parasites —c'est-à-dire une solution proposée par l'analyseur morphologique mais non attestée par la langue dans un contexte donné— pour faciliter la suite de l'analyse et éviter l'explosion combinatoire. Cette levée d'ambiguïtés est réalisée à l'aide d'une méthode statistique faisant appel aux filtres de Markov [Bouchaffra, 92], [Bouchaffra & al., 93]. Dans un tel modèle, la probabilité d'apparition d'une catégorie morphologique est calculée en fonction de l'apparition d'un nombre déterminé de catégories précédentes. Une fois la probabilité de chaque catégorie connue, la probabilité de la séquence entière s'obtient par produit de toutes les probabilités.

I-1-2) Le niveau syntaxique

L'objectif d'une analyse syntaxique est de reconnaître et de construire les différentes composantes syntaxiques d'une phrase.

I-1-2-1) Différentes approches possibles

Une comparaison des modules syntaxiques de différents systèmes d'analyse du français est exposée dans [Fay-Varnier & al., 91]. Nous présentons donc succinctement les hypothèses de l'analyseur syntaxique CRISTAL.

• Analyse syntaxique déterministe vs non déterministe

Certains auteurs mettent en œuvre des analyseurs déterministes [Rady, 83]. Ils supposent que leur analyseur peut trouver la "bonne solution" directement ; ce dont nous doutons. Nous voulons donc, pour effectuer l'analyse syntaxique d'une langue, utiliser une grammaire telle qu'elle puisse reconnaître toute phrase de la langue. Le langage est ambigu, la grammaire ne peut, par conséquent, que l'être aussi. Nous optons donc pour la mise en œuvre d'une grammaire non déterministe dont l'objectif est de fournir toutes les analyses syntaxiques attestées par la langue.

• Grammaire contextuelle vs hors-contexte

Il existe différents types de grammaires développées pour l'analyse syntaxique (cf. [Stéfanini, 93]). Dans la mesure où nous travaillons sur la langue en général, indépendamment d'une application, nous préférons utiliser une grammaire générale hors-contexte. Mais, dans ce type de grammaire, les phénomènes linguistiques mettant en jeu des dépendances de longue distance (comme la coordination ou l'ellipse) sont complexes à formuler. Il est possible d'utiliser des algorithmes opérant par îlots de confiance permettant de pallier cet inconvénient [Stock & al., 88].

I-1-2-2) La syntaxe dans CRISTAL

Le but de la syntaxe est de fournir la (ou les) bonne(s) analyse(s) syntaxique(s) d'une phrase.

• Regroupements des morphèmes discontinus

Des traitements préalables permettant le regroupement des morphèmes discontinus sont nécessaires.

Exemples : cas de la négation : "*ne … pas*"
cas des verbes conjugués avec un auxiliaire : "il *a … fait*"

• Segmentation

La segmentation des phrases en propositions est issue des travaux de Maegaard et Spang Hanssen [Maegaard & Spang Hanssen, 78]. Cet automate (cf. [Stéfanini, 88]) segmente une phrase en propositions, en détectant les introducteurs de subordonnées et les verbes conjugués. Mais cet algorithme peut segmenter de manière fortuite en l'absence d'indicateurs signifiants. Ce qui conduit parfois à une segmentation et, par suite, à une analyse erronées. Le risque d'erreurs est plus grand lorsque le nombre d'interprétations morphologiques concurrentes des formes est élevé.

Exemple : la phrase "*Il détient le pouvoir, il veut le garder*" est décomposée en deux propositions coordonnées : "*il détient le pouvoir*" et "*il veut le garder*".

• Analyse syntaxique

L'analyse syntaxique proprement dite des propositions est réalisée par un système expert (cf. [Dami et al., 92]) utilisant un algorithme dérivé de celui d'Earley [Earley, 70]. Cet analyseur combinatoire prédictif construit des structures syntaxiques engendrées sur une chaîne par une grammaire hors-contexte.

La grammaire hors-contexte utilisée est une grammaire de la proposition assertive [Lallich & al., 90]. Cette grammaire ne décrit que les propositions assertives, elle ne peut donc reconnaître : ni les interrogatives, les impératives,… (que nous regroupons sous le terme générique de "transformations" développées au chapitre VI), ni les phénomènes linguistiques complexes comme la coordination, les ellipses… Par conséquent, la construction de l'arbre syntaxique de la proposition est réalisée à l'aide de trois modules principaux : le module de transformation, le module coordination intra-proposition et l'expert en prédiction [Dami et Lallich, 91].

Par exemple, la proposition "*Il garde le fichier*" est analysée de la manière suivante :

Structure syntaxique :

```
            PPP
               PP
                  SV
                     Y il {MAS SNG…}
                  VVV
                  |  VV
                  |     V garder {P13 PST…}
                  NNN
                     DD
                     |  D  le {MAS SNG…}
                     NN
                        N
                           F fichier {NOM MAS SNG…}
```

I-2) Limites d'une architecture séquentielle : le système CRISTAL

L'intérêt d'utiliser une approche distribuée pour un système d'analyse a déjà été étudié au sein de notre équipe [Stéfanini & al., 92] et [Stéfanini, 93]. Des travaux en linguistique ont alors validé l'utilisation d'une telle architecture distribuée : traitement des erreurs [Froissart, 92], étude des syntagmes nominaux complexes [Eymard, 92], modèle de morphologie dérivationnelle [Clavier, 96a].

Nous voulons, ici, rappeler les principaux inconvénients des systèmes séquentiels à l'origine de l'adoption d'une architecture distribuée, à savoir : le manque d'interaction entre les niveaux de représentation, l'absence de distribution du contrôle et des connaissances, la difficulté d'évolution du système.

I-2-1) Manque d'interaction entre les niveaux de représentation

Un système séquentiel n'autorise pas de coopération entre les différents niveaux de représentation. Chaque module de l'analyse se limite à ses propres connaissances : chaque module comprend en entrée les résultats du module précédent et fournit en sortie l'entrée du module suivant. Par exemple, lors de l'analyse morphologique aucune information de nature syntaxique n'est disponible. Chaque module doit traiter toutes les solutions possibles (dont certaines solutions "parasites"). Soit le module transmet toutes les solutions aux niveaux ultérieurs ce qui provoque une complexification de l'analyse, voire une explosion combinatoire. Soit, il utilise une méthode permettant d'éliminer certaines de ces solutions ; le risque étant d'éliminer de "bonnes" solutions.

L'interaction entre modules peut éviter de transmettre certaines solutions parasites (c'est-à-dire une solution possible, de manière hors-contexte, mais non attestée par la langue dans un contexte précis) d'un niveau de représentation à l'autre ; en effet, chaque niveau engendre des ambiguïtés (c'est-à-dire plusieurs solutions pour un même problème) dont certaines peuvent être résolues à un niveau ultérieur.

> La proposition "*il garde le fichier*" présente des ambiguïtés au niveau morphologique.
>
> Par exemple, la forme "*garde*" peut être un Nom ou un Verbe ; ce qui peut être résolu aisément au niveau syntaxique dans la mesure où la forme la précédant (i.e. "*il*") est un pronom.

I-2-2) Manque de distribution des connaissances et du contrôle

Le système séquentiel CRISTAL, analysant la langue en général, utilise un volume important de connaissances (dictionnaires, règles,…), ce qui entraîne une certaine complexité du système quant à son développement et sa maintenance. Même si l'utilisation de règles hors-contexte en réduit le volume, une décomposition des connaissances est tout de même nécessaire. En effet, une décomposition des tâches complexes en sous-tâches permet une réduction de cette complexité, un système distribué est donc souhaitable ; et plus particulièrement pour traiter certains phénomènes linguistiques complexes (tels que la coordination, les ellipses,…) qui nécessitent différentes grammaires.

Ainsi, la décomposition des connaissances en plusieurs modules permet, d'une part, de diminuer la complexité des différents niveaux "classiques" de représentation (morphologie, syntaxe,…), et d'autre part, de faciliter le développement, le test et la mise à jour des phénomènes linguistiques complexes utilisant des grammaires locales. Les grammaires locales —proposées par A. Berrendonner— étant un ensemble de règles de grammaire activables à partir de certains indices présents dans le texte (cf. [Stéfanini, 93]).

De plus, une architecture séquentielle, contrairement à distribuée, ne permet pas la distribution du contrôle. Cependant, le contrôle distribué permet des prises de décision interactives, à l'aide de mécanismes de coopération et de négociation, où les différents points de vue d'experts pour un même problème sont échangés. La distribution du contrôle permet donc de tenir compte des points de vue divergeant de différents experts.

I-2-3) Système difficilement évolutif

Une architecture séquentielle est difficilement évolutive, or l'analyse automatique de la langue est un domaine en constante évolution ; il en est de même de la configuration d'un analyseur.

- **Intégration de nouveaux modules**

Il est souhaitable de pouvoir intégrer aisément des stratégies nouvellement mises à jour de traitement de certains phénomènes linguistiques. De même, il doit être possible de modifier facilement et localement les données et les stratégies d'un module en fonction des résultats des recherches sur un phénomène linguistique donné.

- **Utilisation de méthodes complémentaires**

L'utilisation de méthodes complémentaires (méthodes dont les résultats sont complémentaires) est intéressante et aisée à mettre en œuvre dans un système distribué. Selon certains critères, comme par exemple le type d'application, il peut être intéressant de privilégier l'une, ou l'autre, ou les deux méthodes complémentaires. Par exemple, les deux analyseurs morphologiques suivants pourraient être utilisés : l'expert en morphologie flexionnelle utilisé dans le système séquentiel CRISTAL [Lallich & al., 90] et l'expert en morphologie dérivationnelle présenté dans [Clavier, 96a].

- **Manque d'expertises en concurrence**

Un système séquentiel se prête difficilement à l'utilisation d'expertises mises en concurrence. Par contre, dans une architecture distribuée, il est aisé d'intégrer plusieurs experts dans le même domaine utilisant différentes connaissances et/ou différentes méthodes. Les experts peuvent alors produire des expertises différentes, contradictoires ou complémentaires sur un problème donné. La comparaison des résultats fournis par les experts permet de valider ou d'infirmer la pertinence d'un expert ; ou encore de montrer l'intérêt d'utiliser les deux experts, chacun étant compétent dans des cas spécifiques.
Par exemple, deux méthodes de désambiguïsation morphologique pourraient être mises en concurrence : une méthode statistique utilisant l'algorithme de Markov [Bouchaffra, 92], [Bouchaffra & al., 93] et une méthode utilisant des règles linguistiques contextuelles [Lebourg, 88], [Stéfanini, 93].

I-2-4) Bilan

L'architecture séquentielle permet difficilement de résoudre des problèmes liés aux solutions multiples et à la complexité des structures [Dami & al., 92], [Eymard & al., 92]. D'une part, la propagation des solutions concurrentes d'un module vers un autre module provoque une explosion combinatoire ; et, d'autre part, la complexification des structures découle de l'utilisation d'une grammaire générale qui essaie de prévoir toutes les possibilités de la langue. Par conséquent, les objectifs sont : la restriction des solutions multiples par la coopération des modules et la décomplexification des structures par l'activation de grammaires locales.

Ces types de problèmes nous ont conduit à faire une étude de différents systèmes de traitement automatique de la langue utilisant une architecture distribuée.

II) Etude de systèmes distribués de TAL

De nombreux systèmes de Traitement Automatique des Langues (TAL) utilisent une architecture séquentielle comme les systèmes ASK [Thomson & Thompson, 85], LOQUI [Binot & al., 88], SAPHIR [Erli, 87], etc. Ces types de systèmes séquentiels ont été étudiés dans [Stéfanini, 93] et présentent des inconvénients similaires à ceux du système séquentiel CRISTAL.
Nous nous sommes donc intéressé au domaine de l'Intelligence Artificielle Distribuée (IAD) et, plus particulièrement, aux Systèmes Multi-Agents (SMA), pour pallier les

inconvénients exposés précédemment. Après une présentation générale des systèmes distribués, nous étudions différents systèmes de TAL utilisant de telles architectures.

II-1) Intelligence Artificielle Distribuée et Systèmes Multi-Agents

Nous exposons dans cette section, les principes de base de l'Intelligence Artificielle Distribuée puis les spécificités des Systèmes Multi-Agents.
Certains termes du vocabulaire usuel de l'IAD (termes issus de différents domaines) peuvent sembler naïvement anthropomorphiques et sont parfois controversés. Nous reprenons cependant cette terminologie, dans la mesure où nous considérons que les métaphores citées sont particulièrement utiles à un certain niveau de compréhension de certaines notions relativement complexes. Notons, qu'une première tentative d'homogénéisation du vocabulaire multi-agents a été réalisée [Pleiad, 92].

II-1-1) L'Intelligence Artificielle Distribuée

• Généralités

Le but de l'Intelligence Artificielle Distribuée (IAD) [Bond & Gasser, 88] est de distribuer logiquement voire géographiquement le contrôle et les données dans des entités qui collaborent en partageant les informations nécessaires pour la résolution d'une tâche globale.
L'intérêt pour l'IAD provient en partie de l'échec des méthodes de l'intelligence artificielle classique ; ainsi les méthodes distribuées, ou locales, utilisant la distribution des connaissances et du contrôle deviennent plus intéressantes. De plus, la distribution des problèmes rejoint un phénomène naturel : les problèmes présentent une distribution géographique ou spatiale, une distribution fonctionnelle des tâches à effectuer ou encore une distribution méthodologique (des solutions partielles concourant à la résolution d'un problème).

L'IAD se décompose en trois thèmes [Gleizes & Glize, 90] :
- la résolution distribuée de problèmes (Distributed Problem Solving),
- la résolution multi-agents,
- les architectures parallèles dédiées.

Les systèmes d'IAD sont adaptables à de nombreux problèmes naturellement distribués. Ils permettent une réduction de la complexité par décomposition d'un problème, une certaine modularité et une sûreté de fonctionnement. De plus, le développement, la gestion et l'évolution de tels systèmes sont facilités par la décomposition du problème initial.

• Problématique

Cependant, l'IAD pose différents problèmes [Bond & Gasser, 88] :
- décomposition, allocation et distribution de tâches : comment formuler, décrire, décomposer, allouer et distribuer les problèmes et comment synthétiser les résultats au sein d'un groupe d'agents autonomes,
- interaction, langage et communication : quoi, quand et comment les agents communiquent,

- cohérence et coordination : comment garantir la cohérence du système et une bonne coordination,
- modélisation des autres agents (ex. : de leurs compétences) et activités organisées : comment représenter les autres agents, leurs connaissances, leurs actions et leurs plans,
- disparité entre agents : comment reconnaître et gérer les incertitudes et les conflits.

II-1-2) Les Systèmes Multi-Agents

• Définitions

Un Système Multi-Agents ([Erceau & Ferber, 91], [Ferber & Gasser, 91], [Genesereth & Ketchpel, 94], [Ferber, 95]) est une population d'agents autonomes en interaction ; un agent est une entité réelle (robot) ou abstraite (module d'un logiciel) située dans un environnement dans lequel elle est capable d'agir. Cette entité dispose d'une capacité de perception et de représentation partielle de cet environnement ; elle peut communiquer avec d'autres agents. Un agent possède un comportement autonome, conséquence de ses observations, de ses connaissances et des interactions qu'il entretient avec les autres.

Un agent est donc une entité capable d'agir rationnellement et intentionnellement en respectant ses propres buts et l'état courant de ses connaissances.

Nous adoptons certains principes formulés par [Gaspar, 91] : la sincérité de l'agent émetteur, et la crédulité de l'agent receveur (un agent croit tout ce qui n'est pas contradictoire avec ses croyances).

Ces principes permettent de comprendre le comportement général d'un agent ; seul le cas de conflit (où les informations de l'agent émetteur sont contradictoires avec celles de l'agent receveur) reste problématique.

• Distinction entre Système Multi-Agents et Système Multi-Experts

La définition d'un système multi-agents par rapport à un système multi-experts varie selon les auteurs.

Certains auteurs, comme Haton [Haton, 89], distinguent deux grandes catégories de SMA suivant la nature des agents :

- Les Systèmes Multi-Robots dans lesquels les agents sont des entités physiques distinctes, ayant une autonomie de décision et d'action. Ce thème relève pour l'essentiel de la robotique [Ferber, 90].
- Les Systèmes Multi-Experts (SME) où les agents sont des entités informatiques.

Selon cette classification, notre application d'analyse du français écrit relève des systèmes multi-experts.

Par contre, pour [Gleizes & Glize, 90] un système multi-experts est un logiciel composé de modules "*agents experts*" alors qu'un "*agent est seulement capable d'effectuer quelques tâches ne demandant ni esprit ni réflexion*".

A partir de la définition d'un agent que nous avons retenue, nous dirons que dans un SMA l'agent dispose, en plus du module expert (i.e. ses connaissances, ses compétences,…), de capacités de raisonnement et d'interaction avec les autres agents du système ; alors que dans un SME, les experts ne comportent qu'un module d'expertise. En nous basant sur cette distinction entre SMA et SME, l'architecture TALISMAN a les caractéristiques d'un SMA.

Les SMA se différencient selon de nombreux critères dont : le type d'agents, le type de comportement des agents, le mode de communication, le type de contrôle et la structure d'implantation.

II-1-2-1) Le type d'agents

En général, on distingue deux principaux types d'agents : *réactifs* ou *cognitifs* [Ferber & Gasser, 91]. Une décomposition de ces deux types d'agents suivant leurs compétences peut être trouvée dans [Ferber & Gasser, 91] ; et une comparaison entre les agents réactifs et cognitifs est effectuée par [Mandiau, 93].

Le mode de fonctionnement des agents cognitifs est comparé à une organisation de type social et celui des agents réactifs s'inspire, par métaphore, de l'organisation biologique. Le manque de mémoire locale et l'absence de mécanismes de raisonnement des agents réactifs sont des inconvénients pour le traitement des phénomènes linguistiques qui nécessitent la mise en place d'heuristiques complexes ; nous nous intéressons donc aux agents cognitifs.

Nous dirons qu'un agent cognitif [Esfandiari et al., 95] est capable de raisonner à partir de représentations de son environnement, des autres agents, de ses propres connaissances et de celles des autres.

Schématiquement, nous dirons qu'un agent cognitif comporte au moins deux modules (cf. Figure 2) :

- un module de communication qui permet à l'agent de communiquer avec les autres,
- un module d'expertise du domaine correspondant au savoir-faire de l'agent.

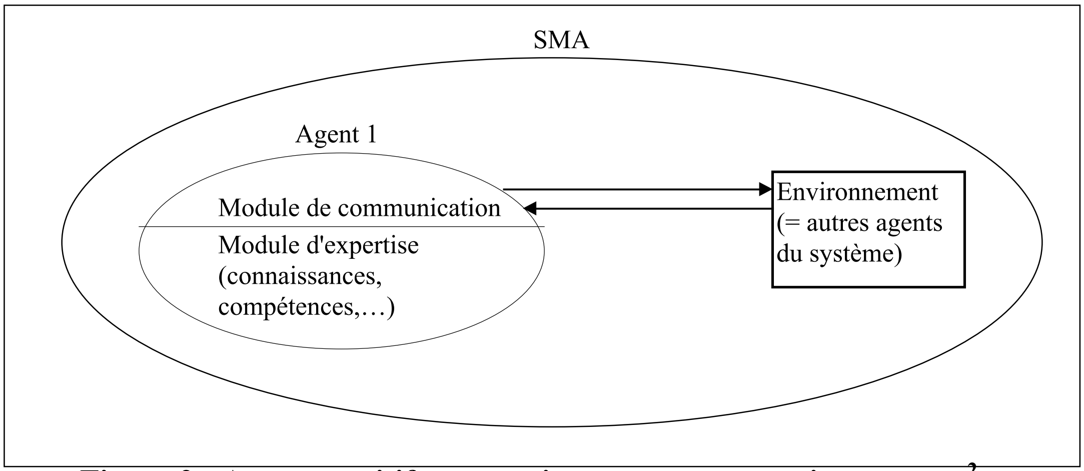

SMA
Agent 1
Module de communication
Module d'expertise (connaissances, compétences,…)
Environnement (= autres agents du système)

Figure 2 : Agent cognitif communiquant avec son environnement[2]

II-1-2-2) Le type de comportement des agents

Le type de comportement interne d'un agent peut varier (égoïste vs altruiste) ; de même, le type de comportement externe peut être différent (coopératif vs conflictuel).

- [Béron & al., 95] distinguent les agents *égoïstes* des agents *altruistes*. Un agent égoïste n'effectue que les tâches nécessaires pour atteindre ses objectifs individuels alors qu'un agent altruiste peut effectuer des tâches annexes à son objectif immédiat qui seront utiles à d'autres agents du système.

Nous nous placerons dans le cas d'agents altruistes, puisque les agents du système sont censés répondre aux besoins des autres dans la limite de leurs compétences.

---

[2] Nous considérons que l'environnement d'un agent est constitué par les autres agents du système avec lesquels il communique.

- [Haton, 89] distingue les agents *coopératifs et complémentaires* des agents en *compétition*.

Dans le cas d'agents coopératifs et complémentaires, la notion de base est celle de coopération volontaire, non conflictuelle, des agents. Chaque agent possède une expertise dans un certain domaine et participe à la résolution d'un problème. Cette situation recouvre des cas simples de découpage modulaire, et des situations complexes où il existe une forte interaction entre plusieurs experts (ayant des points de vue différents) qui doivent mettre toutes leurs compétences en synergie pour résoudre un problème difficile.
Dans l'autre cas, il s'agit de concurrence. Plusieurs agents sont susceptibles de rentrer en conflit pour la résolution d'un sous-problème, du fait d'une expertise différente. La conception de tels systèmes engendre des problèmes de gestion de conflits entre agents, de maintien de la cohérence et plus généralement de mise en place des structures de contrôle.

En pratique, il est parfois nécessaire de définir un système où la coopération volontaire et la concurrence entre agents coexistent (ce qui paraît le plus représentatif de la réalité humaine), afin d'améliorer la résolution d'un problème. Ce type de système correspond davantage à un système d'analyse automatique de la langue. En effet, les agents d'un système d'analyse ont un but commun et pour l'atteindre il doivent coopérer puisque ils possèdent des compétences complémentaires. Il ne faut cependant pas exclure la mise en concurrence de différentes méthodes (réparties dans différents agents) pour atteindre un même sous-but, ce qui peut être profitable aux performances du système.

II-1-2-3) Les modes de communication

Les SMA disposent de trois grands modes de communication (dans les deux premiers modes la communication est intentionnelle, dans le dernier elle est dite "non-intentionnelle") [Ferber & Gasser, 91] :

- *La communication par mémoire partagée* nécessite une structure de données commune à l'ensemble des agents, comprenant tous les éléments nécessaires à la résolution d'un problème. Les agents déposent et lisent les informations sur cette zone de données communes appelée "tableau noir" (ou "blackboard").
- *La communication par messages* est représentée par l'envoi de messages asynchrones suivant les protocoles de communication définis. Les connaissances et les mécanismes de traitement pour la résolution d'un problème sont distribués dans les agents. La communication s'établit alors point par point ou par diffusion ("broadcast").
- *La communication par environnement* provient du simple fait que les agents laissent des traces (signaux) de leur présence qui peuvent être perçues par d'autres agents. Ce mode de communication s'applique à des domaines comme l'éthologie ou la robotique.

Nous ne nous attardons pas sur la communication par environnement qui est peu applicable à notre domaine d'application. Nous devons donc choisir entre une communication directe ou indirecte entre agents. La communication intentionnelle par envois de messages est plus adéquate avec un système d'analyse du français écrit [Stéfanini, 93]. En effet, les interactions par partages d'informations —tableau noir— nécessitent l'existence d'une structure de données commune ce qui paraît inapproprié à l'analyse du français écrit car les différents niveaux de représentation disposent de structures différentes ; de plus, les résultats des différents modules ne sont pas nécessaires à tous. En effet, le module de prétraitement, par exemple, n'a pas besoin des résultats des autres modules comme la coordination, la syntaxe,… La communication directe est donc plus appropriée à un système d'analyse ; elle lui permet, de plus, de s'adapter dynamiquement à la modification de l'environnement (notamment de profiter des résultats des autres agents du système).

II-1-2-4) Les types de contrôle

Nous distinguons deux formes opposées de contrôle dans un système multi-agents : le contrôle centralisé et décentralisé [Gleizes & Glize, 90].

- Le *contrôle centralisé* est établi par un agent de la société qui supervise l'ensemble des activités des autres agents. Ce type de contrôle facilite la cohérence du système mais il restreint l'autonomie des agents. Ainsi, la modification, l'ajout ou la suppression d'un agent ne sont pris en compte que par le superviseur, et n'ont pas ou très peu d'influence sur les autres agents. L'inconvénient majeur d'un contrôle centralisé réside dans le fait que si l'agent superviseur est bloqué, l'ensemble du système l'est également.
- Avec un *contrôle décentralisé*, les décisions sont prises localement par chaque agent. Ce type de contrôle permet une plus grande fiabilité et extensibilité du système mais il engendre des problèmes de cohérence puisque les agents sont autonomes et ont tous le même poids dans la prise de décision.

En pratique, il peut être intéressant de développer des structures de *contrôle mixte* permettant des prises de décisions locales dont la cohérence est assurée par une forme quelconque de contrôle centralisé.

Pour notre système d'analyse, nous préférons utiliser un contrôle décentralisé dans la mesure où il permet de respecter l'autonomie des agents ; ainsi chaque agent utilise ses compétences linguistiques spécifiques, sur lesquelles les autres agents, n'ont bien souvent, aucune qualification de jugement ou compétences.

II-1-2-5) Les structures d'implantation

Les deux principaux types d'architecture sont les architectures de type "tableau noir" (ou "blackboard") et les architectures fondées sur les langages acteurs.

- *Le tableau noir* représente un modèle général de résolution de problèmes permettant la coopération de différentes sources de connaissances (experts). Ainsi, en général, les systèmes comprenant un contrôle centralisé utilisent une implantation de type tableau noir ; le mode de communication étant alors par mémoire partagée.

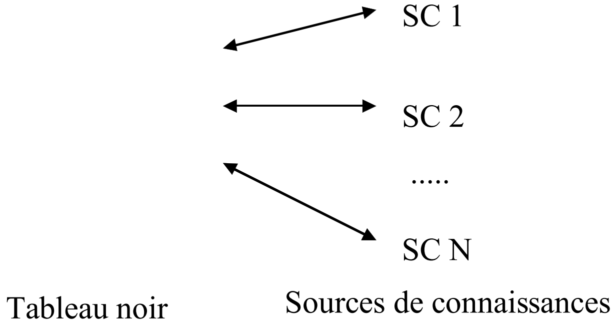


Figure 3 : Architecture du type "tableau noir"

La construction de la solution s'établit selon un processus incrémental : l'avancement de la solution est visible sur le tableau noir.

Nous ne nous attardons pas sur la notion de "*tableau blanc*" introduite par [Boitet & Seligman, 94] dans la mesure où il constitue un cas particulier des tableaux noirs. En effet, les auteurs supposent que la différence réside dans le fait que le tableau blanc, géré par un coordinateur, est caché des différents composants ; chaque composant possède un "manager" faisant le lien entre le composant et le tableau blanc.

- *Le modèle acteur* (ou systèmes à objets) est composé d'acteurs (entités informatiques non décomposables) qui communiquent les uns avec les autres par envoi de messages asynchrones. Un acteur possède un environnement, un ensemble d'accointances, un ensemble de données et d'actions qui définissent son comportement en fonction des messages reçus. Un SMA fondé sur le modèle acteur dispose donc d'une communication directe entre agents par l'intermédiaire d'envois de messages ; le type de contrôle est alors généralement décentralisé. Cette structure d'implantation est adapté à nos besoins : contrôle décentralisé et communication directe entre agents.

Il existe des solutions mixtes dans lesquelles le partage des connaissances et la communication entre agents relèvent des deux modèles de base développés ci-dessus ; ces systèmes sont qualifiés de "société d'experts" par [Haton, 89].

II-1-2-6) La résolution dans un Système Multi-Agents

Nous dirons que la résolution distribuée d'un problème dépend de plusieurs paramètres dont :

- Le type d'agents (cognitifs, réactifs) utilisé.
- Le type de contrôle (centralisé, décentralisé ou mixte) implanté.
- Le type de comportement (coopératif vs conflictuel) des agents.
- La technique de mise en œuvre des interactions qui peuvent être dirigées par les données —partage de résultats— , dirigées par les buts —partage de tâches— ou mixte. La résolution par partage de tâches se déroule de la manière suivante : décomposition du problème en sous-tâches, distribution des sous-tâches, résolution et intégration des résultats. La résolution par partage de résultats nécessite une distribution des connaissances, une résolution et une synthèse des résultats. La résolution dans TALISMAN s'effectue par l'utilisation de ces deux techniques de base permettant la mise en œuvre des interactions entre agents.

II-2) Systèmes distribués de Traitements Automatique de la Langue

Il existe des systèmes de Traitement Automatique de la Langue qui utilisent une architecture distribuée pour pallier les inconvénients du séquentiel. Nous allons étudier certains de ces systèmes, dans la perspective de les comparer aux niveaux de leurs spécificités linguistiques et informatiques.

II-2-1) Présentation générale de systèmes distribués de TAL

Les systèmes de Traitement Automatique de la Langue sont présentés ici succinctement puisque notre objectif est d'établir une grille comparative[3] ; une présentation plus détaillée de chaque système se trouve dans les annexes (cf. Annexe-Présentation de systèmes distribués de TAL). Nous ne présentons que les systèmes sur lesquels nous avons pu obtenir suffisamment d'informations[4] ; ainsi les systèmes pas assez détaillés ne sont pas exposés comme [Fum & al., 88], [Finck, 94], [Schmutz, 95].[5]

II-2-1-1) Le système HEARSEY II

- **Présentation générale**

Le système *Hearsay II* [Erman et al., 80] de compréhension de la parole est fondé sur le principe d'un système multi-agents avec une communication indirecte entre agents par l'intermédiaire d'un tableau noir (ou "blackboard"). Chaque source de connaissance est un module qui peut générer des hypothèses qu'il inscrit sur le tableau noir et modifier des hypothèses existantes. Ce tableau noir est divisé en plusieurs parties correspondant aux différents niveaux de représentation intermédiaire du décodage de la parole. Il a deux rôles : représenter les états intermédiaires de l'activité de résolution du problème et communiquer les hypothèses d'une source de connaissance à d'autres, ce qui permet de les activer.

- **Remarques**

Ce système est le premier système de TAL à avoir utilisé une architecture distribuée. Il fournit une structure permettant à diverses sources de connaissances de coopérer pour la compréhension de la parole en réduisant l'incertitude. Ce système utilise un vocabulaire de seulement 1 000 mots et fournit une interprétation correcte à 90%. Son extension à un lexique plus conséquent permettrait de valider l'architecture proposée.

II-2-1-2) Le système HELENE

- **Présentation générale**

Le système *HELENE* développé par [Zweigenbaum & al., 89] est un système utilisé pour la compréhension de comptes-rendus d'hospitalisation.

Les niveaux d'interprétation du texte sont : analyse morpho-lexicale (lexique, modèles de la flexion), analyse syntaxique (grammaire LFG), analyse sémantique (modèles du domaine, lexique sémantique) et analyse événementielle (modèles du texte).

Le contrôle dans le système est assuré par le tableau noir ABACAB [Bachimont, 90] qui possède un contrôleur. Ce dernier est considéré comme un agent rationnel raisonnant sur ses connaissances, sa représentation du problème et de sa résolution, pour prendre des décisions.

---

[3] Les systèmes sont exposés par ordre chronologique pour mettre en évidence l'évolution des procédés informatiques utilisés.

[4] Les lecteurs noteront certainement que la plupart des systèmes exposés traitent du français. Il était bien entendu intéressant pour nous de comparer les différentes approches des systèmes distribués de TAL traitant la langue française. Notons que, d'autre part, les caractéristiques informatiques des systèmes distribués de TAL traitant d'autres langues (que nous avons pu trouver à ce jour), ne sont souvent que très peu développées ; par conséquent, ces systèmes n'étaient pas très représentatifs pour notre grille comparative.

[5] Le système DIAL (de [Roussanaly & Pierrel, 92]) n'est pas non plus présenté ; il propose une architecture statique avec des liens de communication prédéfinis. L'architecture du système est figée et dépendante de l'application (le dialogue oral homme-machine).

• Remarques

Ce système est destiné à construire des comptes-rendus d'hospitalisation ; il est donc très spécifique à une application et ne possède qu'environ 320 entrées lexicales. Par conséquent, nous pouvons douter de son extension à d'autres domaines d'application ou tout du moins de sa généralisation à un lexique plus complet.

II-2-1-3) Le système CARAMEL

• Présentation générale

Le système *CARAMEL* (Compréhension Automatique de Récits, Apprentissage et Modélisation des Echanges Langagiers) [Sabah, 90a, 90b et 90c], est un système multi-experts utilisant le parallélisme avec une communication indirecte par tableau noir et un mécanisme de gestion des tâches par agenda.

Le système comprend des "processus" (ou experts) liés à l'analyse de phrases, à la compréhension de textes et à la génération de textes. Il utilise une mémoire structurée contenant les connaissances et les représentations nécessaires aux processus ; chaque processus développe son propre tableau noir et le superviseur SIROP effectue une synthèse des tableaux des différents processus.

• Remarques

Ce système paraît très complet du point de vue du traitement de la langue : il comprend l'analyse, la compréhension, et la génération de textes. Il travaille sur un lexique comportant 15 000 entrées (correspondant à environ 350 000 formes fléchies). Quant à l'architecture du système, elle est clairement définie et très intéressante ; nous nous demandons cependant si la gestion des différents tableaux noirs n'est pas trop lourde à assurer et si le superviseur SIROP n'est pas très complexe à mettre en œuvre.

II-2-1-4) Le système DIALECT 2

• Présentation générale

Le système *DIALECT 2* [Mekaouche & Bassano, 91], [Braunwarth & al., 93] est un analyseur linguistique multi-experts pour la recherche documentaire. Dans le système, plusieurs experts coopèrent pour extraire l'essentiel de l'information intéressante dans des documents pertinents. Le système comprend deux modules : un analyseur linguistique et un module de reformulation. Chacun de ces deux modules est fondé sur une architecture de type tableau noir. Certains experts de la reformulation et de l'analyseur linguistique fonctionnent en parallèle.

Le tableau noir contient un agenda gardant la trace de la progression de l'analyse de la phrase.

• Remarques

Un des intérêts de ce système est qu'il permet de "traiter les erreurs lexicales qui représentent environ 80% des erreurs d'un texte" [Braunwarth & al., 93]. Certains aspects de la communication dans cette architecture nous semblent obscurs. En effet, le déclenchement d'un expert par le tableau noir se fait par envoi de messages et les échanges d'informations entre experts sont aussi assurés par des envois de messages : pourquoi les experts ne lisent-ils pas et n'écrivent-ils pas directement sur le tableau noir ? De plus, nous ne voyons pas l'intérêt d'utiliser deux tableaux noirs distincts pour les modules de l'analyseur linguistique et de reformulation puisqu'ils peuvent travailler en parallèle sur certaines données qui leur sont communes.

II-2-1-5) Le système TALISMAN

• Présentation générale

TALISMAN [Stéfanini, 93] est une architecture multi-agents gouvernée par des lois, pour l'analyse morpho-syntaxique du français écrit. L'architecture proposée utilise une communication directe (par l'envoi de messages) entre des agents cognitifs, leur permettant de coopérer. Ces échanges directs de messages sont régulés par un protocole de communication. Le langage de communication et le protocole de communication utilisé dans le système est celui de Sati Sian [Sian, 91].

Le contrôle du système est mixte. Il est assuré par l'utilisation de lois inspirées de [Minsky, 91a et b] : les lois globales (pour la gestion du système) et les lois locales (pour le contrôle réparti dans chaque agent).

Ce système sera exposé plus en détail dans la section suivante car il a permis de définir le cadre général de nos travaux de recherche.

• Remarques

L'objectif de ce système est l'analyse automatique de textes en langue naturelle. L'architecture proposée est intéressante ; une première maquette a été implantée. Son implémentation grandeur réelle est indispensable à sa validation. La problématique de ce système est présentée en détail au paragraphe III de ce chapitre.

II-2-1-6) Le système MICRO

• Présentation générale

Le système *MICRO* (Modélisation Informatique de la Cognition en Reconnaissance de l'Oral) est un système multi-agents pour la compréhension de la parole [Caillaud & al., 94]. L'architecture de ce système est fondée sur l'environnement MAPS [Baujard & Garbay, 90] qui distingue : les connaissances descriptives (données, hypothèses, résultats) contenues dans des agents "KS" (Knowledge Server) et les connaissances opératoires et décisionnelles (règles et méta-règles) contenues dans des agents "KP" (Knowledge Processor). Les agents du système communiquent par envois de messages et réagissent à des événements extérieurs. Les agents KS et KP sont supervisés par d'autres agents appelés "méta-KP" chargés du contrôle des tâches.

Les agents sont rassemblés en 5 groupes : perception, acoustico-phonétique (analytique), linguistique (analytique), prosodie (holistique), et dialogue.

• Remarques

Les auteurs ont simulé une partie du système afin tester quel type de contrôle était nécessaire. Ils ont testé différentes stratégies, avant de conclure à la nécessité d'adopter une stratégie adaptative ; ce qui nous paraît très intéressant. Quant à l'application réalisée par [Antoine, 94], elle comprend un vocabulaire d'environ 200 racines morphologiques.

II-2-1-7) Le système ETCvérif

• Présentation générale

ETCvérif [Cochard & Froidevaux, 95] est un environnement multi-agents de reconnaissance automatique de la parole en continu. Il comprend des agents macroscopiques et microscopiques qui cohabitent et communiquent pour créer une dynamique de résolution de problèmes partiellement répartis ; le système utilise un agent de négociation qui supervise le processus de résolution. Les résultats élémentaires produits par chaque agent macroscopique donnent naissance à d'autres entités réactives nommées agents microscopiques. Des îlots de confiance sont obtenus en interrogeant les agents microscopiques sur leur facteur de confiance en soi. L'agent de négociation gère le processus de calcul de la confiance du système, ainsi lorsqu'un résultat est jugé suffisamment fiable, il ne sera jamais remis en cause jusqu'à la fin de l'exécution.

- **Remarques**

Il nous semble intéressant qu'un agent négociation utilise une heuristique fondée sur des facteurs de confiance pour trouver une solution acceptable. Par contre, nous ne comprenons pas bien l'intérêt de créer de nouveaux agents microscopiques pour chaque résultat élémentaire produit par un agent macroscopique.

II-2-1-8) Le système de Pompidor et Vergnaud

- **Présentation générale**

Le système développé par *Pompidor et Vergnaud* [Pompidor & Vergnaud, 95] est un système multi-agents pour la traduction automatique du chinois dans un but pédagogique. La traduction du chinois en français est vue comme une coopération de connaissances hétérogènes. Le système comporte un agent cognitif de contrôle général et la communication entre agents est directe. Dans le système, les outils informatiques intégrés sont des bases de données gérées soit par des agents cognitifs, soit par un ensemble d'agents réactifs. Ces derniers communiquent par envois de messages à l'aide d'un protocole de communication. L'état de stabilité des agents réactifs ("atteint lors de la raréfaction des messages") correspond à une émergence possible de solution.

- **Remarques**

L'utilisation d'une architecture distribuée pour un système de traduction est novatrice. Cependant comme le soulignent les auteurs, la traduction d'une phrase nécessitant déjà l'action de 16 agents réactifs, quel sera le nombre d'agents créés et les temps de communication pour une traduction plus longue ?

II-2-1-9) Le système CELINE

- **Présentation générale**

CELINE (CorrEcteur LexIco syNtaxiquE) [Genthial & al., 96], [Ménézo & al., 96a et b] est un système multi-agents de détection et de correction des erreurs lexicales et syntaxiques. Il présente une architecture pyramidale et hiérarchique. Le système comprend un superviseur, des "pilotes" par secteurs d'activités et des agents de travail ; dans ce SMA mixte, cohabitent des agents réactifs et des agents cognitifs et la communication se fait par envois de messages.

Le superviseur coordonne l'action de plusieurs pilotes. Chaque pilote est responsable d'un secteur d'activité et coordonne le travail de plusieurs agents de travail. Chaque agent est spécialiste d'un domaine restreint mais peut dépendre de plusieurs pilotes.

- **Remarques**

Il nous semble très intéressant de définir pour la détection et correction des erreurs, un modèle utilisateur comme "*modèle linguistique partiel suffisant*". De plus, CELINE est un système ouvert acceptant des lexiques ou analyseurs divers, et les couvertures lexicales et grammaticales sont donc celles de ces lexiques ou analyseurs. Nous regrettons de ne pas disposer à ce jour d'informations concernant l'échange direct de messages entre agents (format des messages, mode de transmission, rôle du superviseur, protocole de communication,…).

II-2-2) Comparaison de systèmes distribués de TAL

Nous avons effectué une comparaison de ces systèmes distribués de Traitement Automatique de la Langue au niveau de leurs caractéristiques linguistiques et de leur architecture générale.

II-2-2-1) Comparaison en Traitement Automatique de la Langue

Notre objectif est de comparer les aspects linguistiques classiques des systèmes de traitement automatique de la langue étudiés. Ces caractéristiques sont présentées dans le tableau suivant (cf. Tableau 1).

	Spécificité	**Couverture de la langue**	**Morphologie**	**Syntaxe**	**Sémantique**
HEARSAY II	Compréhension de la parole	Vocabulaire de 1 000 mots	?[6]	Analyseur simultané syntaxique et sémantique ("grammaire sémantique" hors-contexte)	
HELENE	Compréhension de comptes-rendus d'hospitalisation	320 entrées lexicales	Analyse morpho-lexicale (lexique, modèles de la flexion)	Analyseur syntaxique (grammaire LFG)	Analyse sémantique et événemen-tielle
CARAMEL	Compréhension automatique de récits	15 000 entrées lexicales	Analyse morphologique flexionnelle	Analyseur simultané syntaxique et sémantique (grammaire de cas et grammaire systémique)	
DIALECT 2	Analyseur linguistique pour la recherche documentaire	Langue en général	Analyse morphologique flexionnelle	?	?
TALISMAN	Analyse morpho-syntaxique du français écrit	Langue en général (50 000 entrées lexicales)	Analyse morphologique flexionnelle	Analyse syntaxique (grammaire hors-contexte de constituants)	?
MICRO	Compréhension de la parole	200 racines morphologiques	Analyse syntaxique (grammaire LFG)		Oui
ETCvérif	Reconnaissance automatique de la parole	Lexique de + de 275 000 entrées	?	?	?
Pompidor et Vergnaud	Traduction automatique du chinois en français	Dictionnaire de mots chinois-français	(morphologie inexistante dans la langue chinoise)	Recueils de structures grammaticales et de phrases types	Eléments sémantiques de base

---

[6] Nous indiquons par un point d'interrogation les aspects qui ne sont pas développés dans les articles étudiés.

	Spécificité	Couverture de la langue	Morphologie	Syntaxe	Sémantique
CELINE	Détection et correction des erreurs lexicales et syntaxiques	Accepte divers lexiques	Analyse lexico-morphologique	Analyse syntaxique (grammaire de dépendances ou de constituants)	?

Tableau 1 : Comparaison des systèmes d'un point de vue TAL

• Spécificité

Nous avons étudié en détail différents systèmes de traitement automatique des langues utilisant une architecture distribuée : trois systèmes de reconnaissance (dont deux intégrant la compréhension) de la parole (HEARSAY II, MICRO et ETCvérif), deux systèmes de compréhension automatique de textes (HELENE et CARAMEL), un système de traduction automatique (de Pompidor et Vergnaud), un système de détection et de correction des erreurs lexicales et syntaxiques (CELINE) et deux systèmes d'analyse du français écrit (DIALECT 2 et TALISMAN).

• Couverture de la langue

Les deux systèmes travaillant sur la langue en général (DIALECT 2 et TALISMAN) utilisent un dictionnaire issu du LADL. Les autres proposent une couverture de la langue plus ou moins large ; les couvertures les plus petites étant HELENE avec 320 entrées lexicales et HEARSAY II avec 1 000 mots. Le système ETCvérif utilise un lexique de plus de 275 000 entrées. Quant au système CELINE, il est intéressant dans la mesure où il est censé accepter divers lexiques pouvant ainsi s'adapter à l'application visée.

• Analyseurs morphologiques, syntaxiques, sémantiques

• Pour les systèmes pratiquant une analyse de l'écrit, *l'analyseur morphologique* utilise en général des modèles flexionnels (HELENE, DIALECT 2, TALISMAN).

• Les grammaires mises en œuvre pour *l'analyse syntaxique* sont diverses :

- HELENE et MICRO utilisent une grammaire LFG.
- TALISMAN utilise une grammaire en constituants de la proposition.
- CELINE est un système ouvert acceptant différents types de grammaire. Les grammaires utilisées par ce système sont : une grammaire de dépendances ou une grammaire de constituants.

• Les systèmes permettant une *analyse sémantique* sont rares :

- HELENE permet une analyse sémantique et événementielle, mais ce système travaille sur un domaine d'application très restreint : les comptes-rendus d'hospitalisation.
- Le système HEARSAY II permet une analyse syntaxique et sémantique simultanée (à l'aide d'une "grammaire sémantique"), mais pour un vocabulaire de seulement 1 000 mots.
- Le système CARAMEL, très complet, effectue simultanément l'analyse syntaxique et sémantique (à l'aide d'une grammaire de cas et d'une grammaire systémique) pour un domaine d'application large (compréhension de récits) avec une couverture lexicale importante (15 000 entrées).

Nous noterons que l'analyse sémantique [Manès-Gallo & Rouault, 96, 97] est en cours de conception pour le système TALISMAN ; son intégration permettrait d'obtenir un système général complet d'analyse automatique du français écrit.

II-2-2-2) Comparaison en SMA

Les critères informatiques retenus pour comparer l'architecture des systèmes étudiés sont présentés dans le Tableau 2.

	Type de contrôle	**Mode de communication**	**Type d'agents**	**Utilisation du parallélisme**
HEARSAY II	Centralisé (gestionnaire de tâches et moniteur du tableau)	Indirecte (tableau divisé par niveaux)	Sources de connaissances	?
HELENE	Centralisé (contrôleur du tableau)	Indirecte (tableau)	"Modules"	?
CARAMEL	Mixte (superviseur et sous-superviseurs)	Indirecte (plusieurs tableaux)	Experts ("processus" ou sources de connaissances)	Oui (plusieurs machines)
DIALECT 2	Centralisé (tableau avec données de contrôle)	Mixte (2 tableaux et envois de messages)	Experts	Oui (pour certains experts)
TALISMAN	Mixte (lois globales et locales)	Directe (envois de messages et protocole d'apprentissage)	Agents cognitifs	Pseudo-parallélisme (simulé)
MICRO	Mixte (Plusieurs méta-KP)	Directe (envois de messages)	Agents "KP", "KS" et "méta-KP"	Oui (grâce à MAPS)
ETCvérif	Mixte (agent de négociation et "relations entre agents")	Directe (envois de messages et protocole de négociation)	Agents réactifs (macro- et micro-scopiques)	?
Pompidor et Vergnaud	Centralisé (par un agent cognitif de contrôle)	Directe (protocole de communication)	Agents réactifs et cognitifs	?
CELINE	Mixte (superviseur et pilotes d'activités)	Directe (envois de messages)	Agents réactifs et cognitifs (ou "pilotes")	?

Tableau 2 : Comparaison des systèmes d'un point de vue multi-agents

• Type de contrôle

Aucun des systèmes étudiés n'utilise un contrôle totalement *décentralisé*. Nous expliquons cela par le fait qu'un tel contrôle est difficile à maintenir sans problèmes d'incohérences. De nombreux systèmes utilisent d'un contrôle *mixte*[7] :

• avec un superviseur et des sous-superviseurs pour CARAMEL, ou avec un superviseur et des pilotes d'activités pour CELINE, ou avec des "méta-KP" pour MICRO, ou encore avec un agent de négociation et des "relations entre agents" pour ETCvérif,

• par le biais de lois (locales ou globales) comme le système TALISMAN.

Les systèmes utilisant un contrôle totalement *centralisé* sont :

• par un module de contrôle situé sur le tableau noir : HEARSAY II, HELENE et DIALECT 2,

• par un agent cognitif de contrôle comme dans le système de Pompidor et Vergnaud.

• Mode de communication

Les systèmes utilisant un mode de communication *indirecte* sont :

---

[7] Nous dirons qu'un système utilise un contrôle mixte lorsqu'il comprend à la fois des aspects de contrôle centralisé et décentralisé.

- HELENE avec un tableau noir classique,
- HEARSAY II avec un tableau noir divisé par niveaux,
- CARAMEL avec plusieurs tableaux noirs.

D'autres systèmes se servent d'une communication *directe* entre agents :

- les agents de CELINE communiquent entre eux par envois de messages,
- dans le système MICRO les agents KS[8] communiquent avec les agents KP par envois de messages et vice versa.
- Pompidor et Vergnaud ont défini un protocole de communication,
- ETCvérif permet des envois de messages et utilise un protocole de négociation,
- TALISMAN a un format de messages précis et un protocole d'apprentissage régulant les envois de messages.

Le seul système utilisant un mode de communication ***mixte*** est DIALECT 2 : il contient deux tableaux noirs qui permettent aux agents de communiquer d'une manière indirecte, et une communication directe entre agents est aussi possible par l'intermédiaire d'envois de messages.

• Type d'agents

Les auteurs utilisent différents termes pour qualifier les entités intervenant dans les systèmes multi-agents : sources de connaissances (HEARSAY II), "serveurs" et "processeurs" de connaissances (MICRO), modules (HELENE), processus (CARAMEL) et experts (DIALECT 2). Il est, par ailleurs, difficile de déterminer si cette différence d'appellation relève d'entités différentes. Toujours est-il que les systèmes les plus récents emploient le terme d'agent cognitif ou réactif. Certains systèmes n'utilisent qu'un seul type d'agents : TALISMAN possède des agents cognitifs, ETCvérif des agents réactifs. D'autres systèmes font cohabiter à la fois des agents réactifs et cognitifs : le système CELINE et le système de Pompidor et Vergnaud.

• Parallélisme

L'utilisation du parallélisme est assez recherchée mais pas encore très développée. Le système CARAMEL le met en œuvre en distribuant ses experts sur différentes machines. Quant au système DIALECT 2, il permet le parallélisme pour certains experts seulement. Dans le système MICRO, le parallélisme est possible grâce à l'utilisation de l'environnement multi-agents (MAPS [Baujard & Garbay, 90]). Enfin, l'architecture TALISMAN utilise du "pseudo-parallélisme".

II-2-2-3) Bilan des comparaisons

Cette étude de systèmes distribués existants en TAL montre qu'il n'existe pas à l'heure actuelle de consensus sur la conception d'un système de traitement automatique de la langue. Chacun essaie de créer un système adapté à ses besoins et ses objectifs ; il n'est pas possible d'affirmer qu'un système est plus performant qu'un autre. L'architecture TALISMAN présente donc des avantages et des inconvénients comme tout système complexe.

Cependant, l'architecture TALISMAN est intéressante dans la mesure où :

- ***Du point de vue linguistique***, les deux seuls systèmes d'analyse travaillant sur la *langue en général* sont TALISMAN et DIALECT 2. Ce dernier est destiné à la recherche documentaire alors que TALISMAN est *indépendant de toute application*. Une des limites actuelles de TALISMAN est qu'il ne comprend pas d'analyse sémantique (contrairement aux systèmes HELENE et CARAMEL). Nous nous limitons aussi dans le cadre de ce travail, à l'analyse morpho-syntaxique du français écrit.

---

[8] Les Knowledge Server contiennent les connaissances descriptives (données, hypothèses, résultats) et les Knowledge Processor comprennent les connaissances opératoires et décisionnelles (règles et méta-règles).

• ***Du point de vue informatique :***

- Il dispose d'un *contrôle mixte* : il nous paraît difficile d'utiliser un contrôle centralisé c'est-à-dire une structure qui posséderait toutes les compétences nécessaires pour superviser l'ensemble des différentes tâches des agents d'un système d'analyse. Si nous voulions créer un tel superviseur, il serait extrêmement complexe.
- Il utilise un mode de *communication directe* entre agents : les différentes composantes ou agents d'un système d'analyse n'ont besoin que d'informations ponctuelles en provenance d'autres agents. Par conséquent, l'utilisation d'un tableau noir engendrerait une disponibilité des résultats de tous les agents, disponibilité disproportionnée par rapport à leurs besoins.
- Il utilise des *agents cognitifs* : l'utilisation d'un mode de communication directe entre agents implique qu'un agent doit avoir des connaissances sur les autres ; ainsi un agent sait, à qui demander telle ou telle information, et à qui fournir tel ou tel résultat.
- Il propose l'utilisation du *parallélisme* : ce dernier permettrait aux agents du système de travailler en parallèle lors de l'analyse d'un texte non seulement sur des phrases différentes, mais aussi sur la même phrase ou proposition.

L'architecture TALISMAN nous semble donc très intéressante et constituera le point de départ de nos travaux de recherche ; c'est pourquoi, nous présentons, dans la section suivante, la problématique linguistique et informatique afin d'exposer les modifications nécessaires.

III) Problématique de l'architecture TALISMAN

Nous commençons par exposer plus en détail le système TALISMAN, dont nous nous inspirons : son architecture générale et les agents qu'il comprend. Ceci nous permet de faire le bilan de ce système, et de proposer les nouvelles orientations linguistiques et informatiques que nous avons mises en place.

III-1) Présentation détaillée de TALISMAN

Les travaux de recherche de M.H. Stéfanini [Stéfanini, 93] ont permis la définition de l'architecture du système et en particulier, celle de la structure et du contenu des agents.

III-1-1) Architecture générale

L'architecture de Talisman [Stéfanini, 93] est celle d'un Système Multi-Agents Gouverné par des Lois (SMAGL) ; ce modèle est formé par le quadruplet (A, L, P, C) où :

- A est l'ensemble des agents de la société,
- L est l'ensemble des lois de la société,
- P est l'ensemble des protocoles de communication de la société,

- C est le langage de communication de la société.

Cette architecture repose sur le modèle acteur dans lequel :

- les agents cognitifs peuvent traiter des phénomènes linguistiques complexes,
- la communication directe entre agents est réalisée par envois de messages,
- le système est contrôlé par l'utilisation de lois [Minsky, 91a et b] : les lois globales (pour la gestion du système) et les lois locales (pour le contrôle réparti dans chaque agent).

Les messages échangés dans TALISMAN comportent 3 champs (émetteur, message, destinataire).
Le langage de communication des agents [Stéfanini & Demazeau, 95] est fondé sur le langage d'apprentissage développé par Sati Sian [Sian, 91] ; chaque agent utilise ses informations locales pour modifier et accroître le degré de confiance en ses hypothèses.
L'utilisation des lois [Minsky, 91a et b] permet de fournir un contrôle supérieur sur la communication ; ces lois permettent de superviser le comportement du système au niveau des échanges de messages entre agents.

III-1-2) Description des agents

• Structure d'un agent

Le modèle d'agent de TALISMAN est inspiré du modèle d'agent générique défini par Y. Demazeau [Demazeau & Müller, 90].
Un agent comprend deux parties, l'une interne représentant sa structure, l'autre externe qui traduit son comportement vis à vis de l'extérieur.

• La ***structure interne*** d'un agent linguistique, telle qu'elle est définie dans TALISMAN, comporte les deux parties suivantes (la première statique, l'autre dynamique) :

1. La représentation des connaissances composée :

- des connaissances de l'agent : mémoire à court terme (point de vue ou opinion sur la phrase) et mémoire à long terme (accointances, structure de données, dictionnaires, grammaires),
- des buts,
- des compétences[9],
- des choix,
- des actions.

2. Le traitement des connaissances de l'agent composé :

- des stratégies,
- des mécanismes de raisonnement et de décision.

• La ***structure externe*** d'un agent représente son comportement ; les agents sont qualifiés de flexibles, adaptatifs et autonomes dans leurs actions et donc dans leurs prises de décision.

• Découpage des agents

Dans TALISMAN, le découpage des agents est réalisé d'une part, en fonction des différents niveaux conceptuels de représentation (ex. : agents Morph, Synt), et d'autre part, suivant les phénomènes linguistiques complexes à traiter (ex. : agents Coord, Néga).

---

[9] Les compétences représentent les différentes procédures informatiques dont dispose l'agent.

Le découpage des agents est un problème classique de l'intelligence artificielle distribuée, dont la résolution diffère d'une application à l'autre. Il est possible d'effectuer ce découpage selon les critères suivants : la localisation, le type d'entrées ou de sorties, le type de tâche, le type de structure de données,…
Dans TALISMAN, le découpage des agents est effectué selon différents critères :

• D'une part, nous dirons que les agents "classiques" (correspondant aux différents niveaux de représentation) ont été définis en fonction de la structure de données sur laquelle ils travaillent. En effet, les agents classiques ont pour objectif la reconnaissance de différents types d'unités : l'agent Regul travaille au niveau du caractère, l'agent Morph réalise la reconnaissance des formes, l'agent Synt permet la détection des syntagmes composant une proposition et l'agent Segm réalise le découpage d'une phrase en propositions. La structure de données se complexifie avec l'avancée de l'analyse ; ainsi, par exemple, la structure de données nécessaire au travail de l'agent Synt correspond à celle fournie en sortie par l'agent Morph.

• D'autre part, le découpage des agents "transversaux" (traitant de différents phénomènes linguistiques complexes) relève du type de tâches à effectuer. Ils ont la particularité de travailler sur différentes structures de données (par exemple, la portée de la coordination peut être : des catégories morphologiques, des syntagmes ou des propositions) et ils n'ont pas forcément besoin de l'ensemble de cette structure de données. Par ailleurs, ces agents transversaux s'activent différemment des agents classiques dans la mesure où leur intervention n'est pas toujours nécessaire (par exemple, l'agent traitant de la coordination ne doit être activé qu'en présence d'un coordonnant).

Le découpage de TALISMAN pourrait éventuellement être remis en cause (ce qui n'engendrerait que peu de modifications), mais cela ne fait pas partie de nos préoccupations dans le cadre de cette étude.

• Agents linguistiques

TALISMAN comprend huit agents cognitifs coopératifs, traitant différents phénomènes linguistiques. Ces agents sont spécifiés dans [Stéfanini, 93].

• *les 4 agents dits "classiques" :*

Pret[10] : expert en prétraitements

But : prétraite, à partir de ses compétences, un fichier de caractères (ou une phrase saisie par l'utilisateur) en le transformant en une suite de formes prétraitées afin qu'elles puissent être analysées.

Connaissances : ensemble de règles permettant les prétraitements morpho-graphiques et morpho-syntaxiques.

Compétences : l'expertise de l'agent comprend les prétraitements morpho-graphiques et les prétraitements morpho-syntaxiques.

Stratégies[11] : exemple de stratégie utilisée par cet agent : tout d'abord, repérage du marqueur de fin de phrase ; puis, traitement morpho-graphique ; enfin, traitement morpho-syntaxique.

---

[10] L'agent Pret(raitements) est renommé dans la suite de ce mémoire en Regul(arisations). En effet, l'agent Pret agissait de manière déterministe et son interprétation n'était jamais remise en cause. Il permettait en quelque sorte l'initialisation du système. Mais, nous pensons que les compétences de cet agent doivent être améliorées, notamment il doit pouvoir faire appel à d'autres agents (en particulier Morph) pour d'éventuels problèmes de découpage de formes (ce qui concerne les formes composées avec ou sans trait d'union).

[11] La stratégie permet de spécifier l'enchaînement des compétences.

Morph : expert en morphologie
But : trouve pour chaque forme les entrées lexicales, et les bonnes catégories morphologiques ainsi que les valeurs de leurs variables (ex. : "*contrôle*" = "contrôle", nom, masculin, singulier,… ou "contrôler", verbe, présent,…).
Connaissances : dictionnaire du français courant construit à partir du DELAS, dictionnaires de spécialité suivant l'application, dictionnaire de certains mots composés, la liste des modèles flexionnels des noms, des adjectifs et des verbes, la liste des flexions ainsi que leur compatibilité mutuelle, les régularisations de formes et de bases.
Compétences : analyseur morphologique flexionnel, désambiguïsation à l'aide de règles linguistiques, regroupement des morphèmes discontinus.
Stratégies : choix du dictionnaire, analyse morphologique flexionnelle, désambiguïsation à l'aide de règles linguistiques.

Segm : expert en segmentation
But : segmente la phrase en propositions (ex. : la phrase "*Il détient le pouvoir, il veut le garder*" est constituée des deux propositions suivantes : "*Il détient le pouvoir*" et "*il veut le garder*").
Connaissances : liste des marqueurs de borne de propositions (introducteurs de propositions subordonnées Q, conjonctions de coordination C, virgules et verbes conjugués).
Compétences : compteur de verbes, détection des propositions principales et subordonnées dans la phrase, construction des arbres représentant la structure de la phrase en propositions et détection de certaines incises.
Stratégies : prédiction sur le nombre de propositions possibles dans une phrase, segmentation et découpage en niveaux, stratégie particulière lors d'une détection de coordination, stratégie de détection de certaines incises.

Synt : expert en syntaxe (en constituants et fonctionnelle)
But : trouve les bonnes analyses syntaxiques d'une proposition (ex. : "*il détient le pouvoir*" → "*il*" est le préverbal Sujet", "*détient*" est le Verbe et "*le pouvoir*" est le Syntagme Nominal Complément d'Objet Direct du verbe).
Connaissances : le dictionnaire des indicateurs de structures lexicaux[12] (ISL), des indicateurs de structures grammaticaux[13] (ISG), la liste de certaines prépositions introduisant des circonstanciels (manière, temps, lieu,…), la grammaire de base.
Compétences : l'analyse syntaxique en constituants de la proposition standard, la vérification des accords (le but étant de désambiguïser, et non pas de corriger des erreurs), la détection des ISG pour la reconnaissance des syntagmes, la détection des ISL pour la détermination du rôle fonctionnel des syntagmes régis et la reconnaissance de certains circonstanciels, et la réalisation de l'analyse fonctionnelle.
Stratégies : prédiction basée sur les ISL et les ISG, appel à l'analyseur d'Earley, détermination du caractère régi ou circonstanciel des compléments, et détermination de la structure fonctionnelle.

- ***les 4 agents transversaux traitant de phénomènes linguistiques :***

---

[12] Les indicateurs de structure lexicaux (ISL) concernent les verbes, les noms et adjectifs dérivés de verbe, quelques adverbes, noms et adjectifs non déverbaux ou régissant des compléments. Un lexique donne pour chacun d'eux un ou plusieurs schémas syntaxiques.

[13] Les indicateurs de structure grammaticaux (ISG) indiquent avec certitude le début d'un syntagme :
- les déterminants (D) qui permettent de repérer les débuts de syntagmes nominaux (SN),
- les prépositions (P) qui introduisent les syntagmes prépositionnels (SP),
- les pronoms préverbaux (Y) et les verbes (V) qui sont initiaux de syntagme verbal (SV).

Transf : expert en transformation
But : détecte, annote les phrases impératives et interrogatives dans le but d'obtenir un ordre canonique (ex. : "*Est-ce qu'il détient le pouvoir ?*" → "*il détient le pouvoir*", question totale notée QTO).
Connaissances : module d'inversion du sujet, liste des formes introduisant une interrogative, une phrase clivée ou une impersonnelle.
Compétences : détection et transformation de certaines questions totales, questions partielles, impératives, phrases clivées et phrases impersonnelles.
Stratégies : recherche et normalisation de certains types de questions totales, questions partielles, impératives, phrases clivées et phrases impersonnelles.

Et les trois agents suivants ayant fait l'objet d'une première modélisation : ***Ellip*** (expert en ellipse), ***Coord*** (expert en coordination) et ***Néga*** (expert en négation). Ces phénomènes linguistiques sont particulièrement complexes à analyser, seulement certaines de leurs compétences ont été modélisées [Stéfanini, 93].

• Bilan

L'introduction d'une architecture distribuée pour un système d'analyse permet une amélioration de ses performances (traitements appropriés aux moments adéquats) grâce à la coopération entre modules linguistiques [Stéfanini, 93]. En effet, les travaux de Marie-Hélène Stéfanini ont montré que l'on obtient une réduction de la complexité par décomposition du problème : la modification des modules existants est facilitée, et l'intégration de nouveaux experts linguistiques est aisée. La pratique d'expertises différentes, complémentaires ou concurrentes est donc possible.
Il a donc été démontré qu'une architecture de type multi-agents (cf. TALISMAN) est mieux adaptée pour l'analyse automatique du français écrit que le traitement séquentiel dans la mesure cette approche permet une augmentation de la modularité, de la flexibilité, de l'extensibilité, de la robustesse et de la concurrence du système ; ainsi il fournit une solution possible à la multiplicité des analyses. L'utilisation d'une telle architecture permet l'élimination rapide de solutions parasites dues aux ambiguïtés inhérentes à la langue : l'objectif étant la résolution des solutions multiples au moment le plus adéquat.

Cependant, l'adoption d'une architecture telle que TALISMAN soulève de nombreux problèmes qui nécessitent une étude approfondie de sa problématique tant au niveau linguistique qu'informatique.

III-2) Problématique linguistique

La présentation des fondements linguistiques de TALISMAN, nous permet de développer la problématique et les nouvelles orientations linguistiques envisagées dans le cadre de ce travail.
Les fonctionnalités linguistiques de base de TALISMAN sont conservées :

- ce système travaille sur la langue en général,
- il est indépendant de toute application,
- son objectif actuel est de fournir la ou les analyse(s) attestée(s) par la langue dans un contexte syntaxique donné.

Cependant, nous envisageons de nouvelles orientations linguistiques : d'une part, une utilisation optimisée des compétences des agents pour résoudre, au bon moment, les solutions multiples ; et d'autre part, des améliorations visant à sa validation en grandeur réelle, et à l'augmentation de ses capacités d'analyse.

III-2-1) Validation de l'architecture en grandeur réelle

Il nous semble indispensable pour valider l'architecture TALISMAN de tester l'analyse de phrases en grandeur réelle.

- L'utilisation d'un **corpus** est primordiale pour tester les performances du système. Nous validerons nos hypothèses sur un corpus journalistique.

- L'intégration dans TALISMAN du **dictionnaire** utilisé dans le système séquentiel CRISTAL, issu du DELAS est nécessaire [Agnel, 97] ; ce dictionnaire comprend plus de 50 000 entrées lexicales permettant de reconnaître environ 300 000 formes fléchies.

- Enfin, une amélioration importante est d'empêcher qu'une forme inconnue du dictionnaire bloque le système. Pour cela, il nous faut introduire une stratégie de **traitement de mots inconnus**.

III-2-2) Développement d'agents

Une étude des agents spécifiés dans TALISMAN, nous conduit à introduire des modifications importantes pour l'amélioration des performances du système.

- Nous souhaitons améliorer les compétences de désambiguïsation de l'agent **MORPH**. Pour cela, nous développerons des règles linguistiques contextuelles de désambiguïsation et nous étudierons sur un corpus les résultats fournis par ces règles.

- L'agent **SYNT** de TALISMAN utilisait l'analyseur d'Earley pour construire l'arbre représentant la structure syntaxique des propositions. Nous allons transformer cet agent en proposant un analyseur syntaxique mixte permettant l'utilisation d'heuristiques (ces transformations seront justifiées au chapitre II).

- Nous proposons différentes stratégies pour l'agent **SEGM** de prédiction du nombre de propositions et de découpage d'une phrase en propositions. Nous étudierons l'intervention de cet agent, notamment dans le cas d'une coordination de deux propositions.

- Nous voulons montrer comment TALISMAN II peut gérer l'analyse de phénomènes linguistiques complexes : nous développerons les compétences linguistiques de l'agent **COORD** (expert en coordination) ; et nous étudierons ses interactions avec les autres agents.
De plus, nous améliorons la modélisation des agents **TRANSF** (expert en transformations), **ELLIP** (expert en reconstruction d'ellipses) et **NEGA** (expert sur la détermination de la portée de la négation). Enfin, nous proposons une première spécification de l'agent **ANAPH** (expert en résolution d'anaphores).

III-3) Problématique informatique

Du point de vue logiciel, TALISMAN est une maquette qui simule le comportement d'agents au sein d'une architecture distribuée.
Certaines caractéristiques informatiques pertinentes de TALISMAN seront conservées, telles que :

- l'utilisation d'un système multi-agents avec une communication directe entre agents,
- le modèle d'agent cognitif, inspiré du modèle générique proposé par Y. Demazeau,
- le découpage des agents de TALISMAN contenant les différents modules linguistiques nécessaires à l'analyse morpho-syntaxique de textes.

Cependant, nous pensons que d'autres aspects de l'architecture générale de TALISMAN (notamment au niveau de la communication et du contrôle) et de son implémentation, méritent des transformations.

• Le concept de lois [Minsky, 91a] n'est pas retenu, dans le cadre de cette étude, (par rapport à [Stéfanini, 93]) ; il fournissait une certaine forme de contrôle aussi bien au niveau local que global. En effet, notre objectif est de vérifier qu'un **contrôle totalement décentralisé** suffit au bon fonctionnement du système ; toute forme de contrôle centralisé étant "impartiale" et ne correspondant pas à nos objectifs :

- chaque agent est autonome,
- les prises de décisions sont locales,
- les agents n'ont pas de relation de hiérarchie.

• Une structuration précise de la **communication** est primordiale pour le bon fonctionnement d'un système distribué utilisant un mode de communication direct entre agents. TALISMAN utilise le langage et le protocole de communication de Sian [Sian, 91] qui permet la coopération entre agents. Nous allons étudier en détail la communication, et en particulier, les protocoles de communication ; cette étude nous permettra de définir une communication adaptée aux besoins en coopération d'un système d'analyse automatique du français écrit. Les conflits linguistiques dérivant de cette coopération doivent être spécifiés ainsi que des procédés permettant leur gestion, voire leur résolution.

• La maquette TALISMAN a été implémenté sous Bim-Prolog ; une implémentation plus conséquente (au niveau des connaissances et des compétences des agents) pour un système d'analyse est indispensable pour justifier l'approche proposée.
L'utilisation d'un langage de programmation tel Prolog est en adéquation avec un système d'analyse. Nous disposons, à l'heure actuelle, de **Prolog II+**[14] qui est plus performant que Bim-Prolog dans la mesure où il permet notamment la simulation du parallélisme. TALISMAN doit donc être implémenté sous Prolog II+.

---

[14] Prolog II+ a été développé par PROLOGIA (Marseille).

CHAPITRE II : GESTION DES CONFLITS LINGUISTIQUES

D'une manière générale, la cohabitation de plusieurs agents dans un même système engendre forcément des conflits entre agents. Ces conflits peuvent être de différentes natures : "conflits d'opinion" (les agents n'ont pas le même point de vue sur un problème), "conflits de ressources" (les agents veulent accéder en même temps à des ressources identiques), "conflits de but" (les agents essaient d'atteindre des buts incompatibles),…

Les ambiguïtés inhérentes à la langue entraînent l'apparition de ces différents conflits. Nous ne nous intéresserons, dans le cadre de cette étude, plus particulièrement aux conflits d'opinion (que nous appellerons conflits linguistiques). Ces conflits linguistiques posent un problème informatique de gestion, voire de résolution.

Nous commençons par présenter les méthodes classiques de résolution des conflits linguistiques : les stratégies classiques de désambiguïsation en analyse, les types d'algorithmes utilisés et l'architecture générale des systèmes. Suite à cette étude, nous présentons les méthodes informatiques retenues pour la résolution de conflits linguistiques : les protocoles de communication. Ensuite, nous effectuons une synthèse des protocoles de communication de la littérature de l'Intelligence Artificielle Distribuée ; avant de proposer les protocoles adaptés à notre problématique linguistique.

I) Les conflits linguistiques

Dans une architecture distribuée telle que TALISMAN, des solutions différentes et concurrentes sont qualifiées de conflits. Les ambiguïtés engendrées par un système de TAL (qu'elles soient virtuelles ou linguistiques) provoquent donc dans un système multi-agents ce que nous nommerons des "conflits linguistiques".

I-1) Définition des conflits linguistiques

Nous dirons qu'il y a conflit chaque fois que plusieurs solutions sont possibles pour un même problème. Lors de l'analyse morphologique d'une forme, nous dirons qu'il y a conflit pour cette forme, notamment, lorsqu'elle relève de plusieurs entrées lexicales (qu'elles correspondent à des catégories morphologiques identiques ou concurrentes) ; lors de l'analyse syntaxique, un conflit peut, par exemple, apparaître lorsque plusieurs analyses syntaxiques concurrentes sont possibles pour une même phrase.

Ces solutions concurrentes sont la conséquence de différents cas possibles :

- **Un agent qui produit plusieurs solutions pour un même problème**

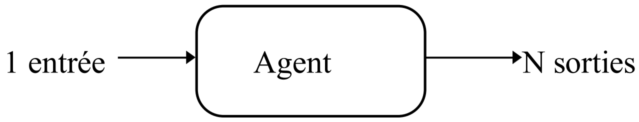


Figure 4 : Plusieurs solutions produites par un agent

Le conflit provient du fait que l'agent manque de connaissances (à un moment donné) pour évaluer les solutions qu'il a trouvées. L'agent doit alors fournir toutes les solutions concurrentes qu'il a engendrées (et qu'il n'a pas pu éliminer) et un autre agent sera peut-être en mesure d'éliminer les solutions parasites. Un agent fournit toutes les solutions concurrentes qu'il crée, car elles doivent toutes être testées puisque le système est non déterministe.

Par exemple, en morphologie, les formes suivantes présentent plusieurs ambiguïtés : "*le double*" où "*le*" est Déterminant ou Pronom et "*double*" est Nom ou Verbe. Lors de l'analyse morphologique de ces formes, les solutions multiples seront conservées par l'agent Morph. Mais un autre agent, relevant vraisemblablement du niveau syntaxique, disposera peut-être de connaissances suffisantes sur le contexte afin de résoudre cette ambiguïté.

- **Plusieurs agents qui produisent des solutions concurrentes pour un même problème**

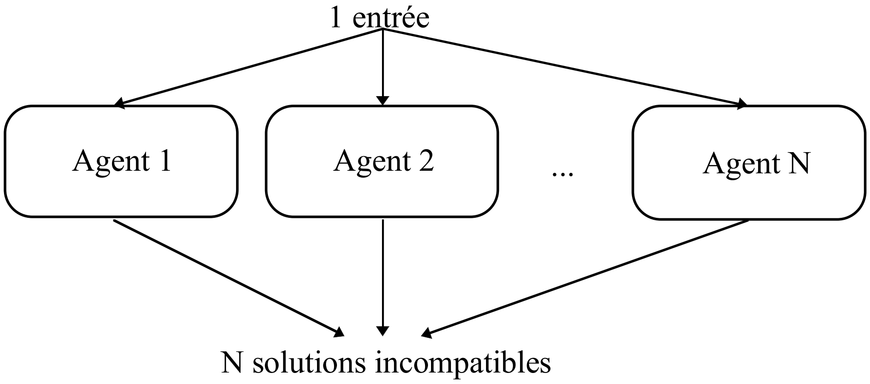


Figure 5 : Solutions concurrentes de différents agents

Dans ce cas, chacun des agents a une solution différente donc incompatible pour un même problème, à un moment donné.

Exemples :

1. "…le fait que les divisions blindées françaises *ne* comptent *que* 200 chars…" (négation restrictive)
2. "CBS, ABC et NBC *ne* peuvent donc toucher les bénéfices de programmes *qu*'ils ont pourtant largement financés et …" (négation totale)

A un moment donné, un conflit peut apparaître entre les agent Nega et Segm lors de l'analyse de ces phrases. En effet, pour l'agent Nega la forme "*que*" sera considérée comme la fin de la négation restrictive introduite par "*ne*". Quant à l'agent Segm, il considérera toujours la forme "*que*" comme un introducteur de propositions.
Ce type de conflit entre différents agents apparaît à un moment donné dans le système, mais il est, en général, résolu avec l'avancement de l'analyse morpho-syntaxique de la phrase.

- **Conclusion**

Nous ne résoudrons les conflits que dans la limite des connaissances du système. Ainsi, certaines solutions parasites seront éliminées par un agent à un moment donné. Les solutions multiples seront, par défaut, conservées, puisque le but du système est de fournir la (ou les) bonne(s) analyse(s) morpho-syntaxique(s). En effet, par exemple, pour le syntagme "*conférence de presse commune*", le système doit fournir les deux structures syntaxiques possibles (cf. chapitre III). Nous supposons que lorsque les niveaux sémantiques et/ou pragmatique seront intégrés, il sera possible d'éliminer certaines analyses syntaxiques erronées.
Quand un agent engendre plusieurs solutions, il envoie aux autres agents toutes les solutions qu'il a trouvées et il indique en premier choix la solution qui lui paraît la plus probable (s'il en a une). Certaines solutions seront éventuellement éliminées ultérieurement, par d'autres agents ou par l'agent lui-même.
De même, dans le cas où différents agents ont émis des solutions incompatibles, certaines seront peut-être éliminées ultérieurement.

I-2) Etude des conflits linguistiques à résoudre

Nous rappelons brièvement les ambiguïtés linguistiques que la maquette TALISMAN permet d'ores et déjà de résoudre. Puis, nous développons les nouveaux conflits linguistiques que nous nous proposons de résoudre.

I-2-1) Conflits résolus dans TALISMAN

Les ambiguïtés ou conflits linguistiques que le système TALISMAN permet d'ores et déjà de résoudre sont les suivantes :

- **Certaines ambiguïtés morphologiques grâce aux règles de désambiguïsation**

Grâce aux règles linguistiques contextuelles de désambiguïsation morphologiques, certains conflits linguistiques produits par l'agent Morph sont résolus.
Exemples :

Règle 1 : Déterminant + (Nom ou Verbe) → Déterminant + Nom
("cette voile")
Règle 2 : Préverbal + (Nom ou Verbe) → Préverbal + Verbe
("il voile")

- **Certaines ambiguïtés grâce aux interactions**

Les interactions possibles dans TALISMAN, permettent une désambiguïsation plus rapide de certains conflits linguistiques. En particulier, pour la désambiguïsation morphologique de certaines formes, les informations de l'agent Synt sont très utiles.

> Exemple :
>
> "Elle fabriquera, dans un premier temps, le produit liquide qui entre dans le processus des photocopies ainsi que des pièces détachées pour la filiale de Minolta en RFA. "

Le syntagme "*le produit liquide*" reste totalement ambigu même après le premier passage de la levée d'ambiguïtés par l'agent Morph : "*le*" est D ou Y, "*produit*" est F ou V et "*liquide*" est F ou V. Mais les informations de nature syntaxique sur le schéma du verbe "*fabriquer*" (nécessitant un C1, i.e. complément d'objet direct), doivent permettre à l'agent Synt de désambiguïser ce syntagme.

I-2-2) Améliorations envisagées

Les améliorations envisagées sont présentées ici succinctement ; elles sont détaillées dans la seconde partie de ce mémoire.

I-2-2-1) Amélioration de la désambiguïsation

- **Par l'agent MORPH**

La méthode de désambiguïsation proposée dans TALISMAN est une méthode fondée sur des règles linguistiques contextuelles de désambiguïsation. Nous voulons affiner les règles morpho-syntaxiques afin d'améliorer la performance du module de levée d'ambiguïtés morphologiques.

- **Par l'agent SYNT**

Dans TALISMAN, l'agent Synt utilisait, pour effectuer l'analyse syntaxique d'une proposition, l'algorithme prédictif d'Earley. Cet algorithme n'est pas suffisant dans la mesure où il ne permet pas de faire appel aux agents appropriés au moment le plus opportun, ce qui représente un des avantages d'une architecture distribuée.

Nous pensons donc qu'un analyseur syntaxique utilisant des heuristiques permet d'améliorer les performances de l'analyse syntaxique, ainsi l'agent Synt peut utiliser au maximum les compétences des autres agents. Cet analyseur syntaxique utilise une analyse descendante et ascendante (i.e. mixte) de la structure en constituants et une analyse fonctionnelle (fondée sur les schémas syntaxiques des verbes) [Stéfanini & Warren, 97].

- **Par les possibilités d'interaction**

Enfin, l'intérêt majeur d'utiliser une architecture distribuée réside dans le fait qu'un agent doit pouvoir disposer de la bonne information au bon moment, ce qui permet une désambiguïsation plus efficace. Pour cela, les agents communiquent entre eux. Cette communication entre agents est primordiale dans un système multi-agents : elle doit être définie très précisément et doit répondre aux besoins en communication que les agents peuvent avoir. Nous nous proposons donc d'étudier plus en détail la communication nécessaire aux agents de TALISMAN II afin d'améliorer les possibilités d'interaction du système et par conséquent d'améliorer les performances du système.

I-2-2-2) Traitement de phénomènes linguistiques complexes

Notre objectif est de fournir un cadre pour le développement d'agents transversaux traitant de phénomènes linguistiques complexes.

- **Résolution de conflits liés à la coordination intra-phrastique**

Nous pensons qu'il est intéressant de développer les stratégies d'un agent transversal afin d'observer son comportement dans un système multi-agents. Nous nous proposons de traiter le cas de la coordination intra-phrastique dans la mesure où ce type de coordination est très fréquente et que, de plus, elle nécessite de nombreuses interactions en particulier avec les agents Morph, Segm et Synt.

II) Méthodes classiques de résolution des conflits linguistiques

Notre objectif est d'obtenir les bonnes analyses au bon moment. Nous devons utiliser des traitements non déterministes afin de rendre compte des ambiguïtés inhérentes à la langue ; et nous ne résoudrons les conflits linguistiques que dans la limite des connaissances du système. En d'autres termes, si le système comprend les connaissances linguistiques suffisantes à la résolution d'un problème, il sera résolu ; sinon le conflit sera conservé.
Nous allons étudier les stratégies classiques de désambiguïsation en analyse ; puis nous opposons l'utilisation classique d'algorithmes à la mise en œuvre d'heuristiques. Cela nous conduit à exposer deux types d'architecture radicalement opposés. Enfin, nous présentons nos choix de résolution de conflits linguistiques.

II-1) Stratégies classiques d'analyse

La plupart des systèmes d'analyse du français écrit utilisent des méthodes de désambiguïsation automatique morphologique et syntaxique pour éviter l'explosion combinatoire conséquence des ambiguïtés de la langue. Le choix de ces stratégies est primordial [Wehrli, 89] dans la mesure où il conditionne un des problèmes majeurs de l'analyse : l'ambiguïté. Certains auteurs, comme [Boitet & Tomokiyo, 96], proposent une désambiguïsation interactive (par un expert humain, par l'utilisateur ou par un système expert) après avoir effectué la désambiguïsation automatique. Nous nous limiterons dans cette étude à la désambiguïsation automatique.

II-1-1) Désambiguïsation morphologique

Nous présentons trois méthodes classiques de traitement des ambiguïtés morphologiques :

- **Désambiguïsation statistique**

L'utilisation d'une méthode statistique avec échantillonnage probabiliste (comme celui utilisé par [DeRose, 88] ou dans le système CRISTAL [Berrendonner & al., 80], [Kallas, 87], [Lallich & al., 90]) fournit de bons résultats[15] ; les calculs sont fondés sur la probabilité de co-occurrence de catégories. Cependant, nous pensons que l'utilisation de critères statistiques pour résoudre les ambiguïtés de la langue est pertinente lorsqu'aucun autre modèle n'est disponible. Ainsi, l'idée de mettre en œuvre des critères linguistiques nous semble plus appropriée.

- **Désambiguïsation linguistique**

L'utilisation de règles linguistiques contextuelles de désambiguïsation [Lebourg, 88] nous paraît très intéressante. En effet, nous pensons que l'utilisation de règles fondées sur des critères linguistiques est appropriée à une désambiguïsation morphologique.

Une comparaison des deux méthodes (linguistique et statistique) de désambiguïsation morphologique a été réalisée dans [Chanod & Tapanainen, 95]. Les auteurs ont obtenus un pourcentage d'erreurs moins élevé lors de l'utilisation de la désambiguïsation avec des règles linguistiques.

Notre objectif est d'améliorer le module de désambiguïsation morphologique qui est fondé sur des critères linguistiques fiables ; ce qui n'exclut pas l'utilisation d'une méthode statistique pour traiter les ambiguïtés restantes.

- **Attente ou intégration d'informations de niveaux supérieurs**

Certains systèmes ne procèdent à aucune désambiguïsation au niveau morphologique mais attendent des connaissances des niveaux supérieurs, en particulier, du niveau syntaxique pour permettre des désambiguïsations morphologiques. Dans notre perspective, cette attente d'informations des niveaux supérieurs est inappropriée : pourquoi retarder la levée d'ambiguïtés lorsque l'on peut utiliser des informations pertinentes le plus tôt possible.

En fait, les désambiguïsations morphologiques (aussi bien statistiques que linguistiques) utilisent certaines informations de nature syntaxique ; mais l'idée est de pouvoir distribuer les connaissances dans les différents agents et aussi de réduire l'augmentation inutile des informations au niveau morphologique.

II-1-2) Stratégies d'analyse syntaxique

Les trois stratégies classiques d'analyse syntaxique, sont :

- **Analyse descendante**

L'analyse descendante (dite dirigée par les hypothèses) : l'interpréteur part des buts, c'est-à-dire de l'ensemble des structures grammaticales possibles (structures qui sont données par les règles syntaxiques), pour arriver aux faits c'est-à-dire aux mots réellement inclus dans une phrase donnée et à leurs catégorisations morphologiques.

L'analyse descendante permet d'éviter de construire des constituants qui, en fin d'analyse, se révéleraient impossible à rattacher. Par contre, elle ne permet pas des analyses partielles ; i.e. toute le phrase doit être reconnue.

En général, c'est la stratégie descendante qui est utilisée : en analysant un groupe nominal, on émet des hypothèses sur ses constituants.

- **Analyse ascendante**

---

[15] Le modèle de Markov utilisé permet une désambiguïsation correcte à plus de 95 %. Il faut cependant noter que cette méthode ne permet pas de désambiguïser au sein d'une même catégorie (par exemple un nominal de catégorie F qui peut être un nom ou un adjectif).

L'analyse ascendante (dite dirigée par les données) : à l'inverse l'interpréteur va des données (mots de la phrase) vers les buts (une structure grammaticale qui peut les représenter).
Au fur et à mesure de la lecture des mots de la phrase, elle tente de remplacer chaque mot par sa (ou l'une de ses) catégorie(s), puis elle parcourt les règles "à l'envers" pour réécrire par étape les éléments obtenus en constituants de plus en plus larges, jusqu'à remonter à l'axiome. Les règles s'appliquent alors en remplaçant leur partie droite par leur partie gauche.
L'analyse ascendante est plus flexible puisqu'elle permet de construire des solutions partielles (il est possible de reconnaître un syntagme nominal dans une proposition même si l'ensemble de la proposition n'est pas reconnue). Mais d'un autre côté, elle est vraisemblablement plus difficile à mettre en œuvre (elle nécessite notamment le recours à la méta-programmation) ; ce qui explique qu'elle soit moins utilisée.

• Analyse mixte : ascendante et descendante
Bien que les deux types d'analyse soient formellement équivalents, on voit que les analyses ascendante et descendante se heurtent à des difficultés de nature différente. Diverses stratégies mixtes ont donc été mises au point pour pallier les inconvénients de ces deux méthodes. Par exemple, certains systèmes utilisent une stratégie guidée partiellement par les données : certaines données servent à déclencher des règles, qui, une fois sélectionnées, s'appliquent de manière descendante. Cela a l'avantage d'utiliser au mieux le fait que, dans une langue comme le français, les débuts de syntagmes sont souvent introduits par des marqueurs spécifiques.
Notre choix s'est donc porté sur un analyseur syntaxique mixte (présenté dans la seconde partie de ce mémoire) qui a les caractéristiques suivantes :
- une analyse descendante pour la mise en œuvre de la grammaire standard hors-contexte de la proposition assertive,
- une analyse ascendante activée lors de la détection d'indices précis dans le texte (avec des appels à des grammaires satellites ou à des méta-grammaires).

II-1-3) Stratégies de traitement d'ambiguïtés

Dans le cas où un algorithme (ou analyseur) doit faire un choix entre plusieurs solutions possibles et n'est pas en mesure de décider (situation d'ambiguïté), plusieurs stratégies sont possibles. Les trois stratégies les plus classiques sont :

• La stratégie du retour en arrière
Elle consiste à choisir, chaque fois que plusieurs solutions possibles se présentent, une solution parmi celles qui sont pertinentes. Le choix d'une solution se fait de façon plus ou moins arbitraire selon les cas : on peut échapper à l'arbitraire total en se donnant un ordre de plausibilité des différentes solutions, ou une heuristique de choix.
On continue alors l'analyse en mémorisant les solutions non retenues (dans une pile) et l'état de l'analyse au moment du choix. Si la suite de l'analyse remet en cause le choix effectué, alors on retourne à l'endroit du choix et l'analyse repart en sélectionnant une autre solution (la décision au sommet de la pile). Si le premier choix est le bon, cette stratégie est efficace (car elle gagne beaucoup plus de temps de traitement), par contre s'il faut effectuer plusieurs retours en arrière, le temps de traitement est long (car cela conduit souvent à refaire plusieurs fois certains calculs). Par ailleurs, les cas de réelle ambiguïté linguistique ne sont pas dépistés puisque dès qu'un choix fonctionne, aucune autre solution n'est

essayée : cet algorithme est donc apparemment déterministe. Enfin, si la pile est vide, on a alors abouti à un échec.

- **La stratégie du traitement en parallèle**

Elle consiste à développer toutes les solutions possibles lorsqu'un choix se présente, et donc à effectuer toutes les analyses possibles de la phrase. Certaines décisions prises conduiront probablement à des impasses et les autres à des analyses pertinentes. Elle présente l'inconvénient d'impliquer des temps de calcul relativement longs et d'entraîner des difficultés techniques liées à l'écriture d'algorithmes simulant le parallélisme. En revanche, cette stratégie permet de ne pas parcourir plusieurs fois la même chaîne ni de refaire plusieurs fois les mêmes calculs, et de donner toutes les solutions dans les cas d'ambiguïté globale linguistique : cette stratégie est donc non déterministe.

- **La stratégie du regard en avant**

Cette stratégie a été utilisée par [Kuno & Oettinger, 62] qui ont proposé un algorithme prédictif descendant non déterministe pour l'analyse de la langue.
De même, le choix de [Rady, 83] s'est portée sur une stratégie de décision qui remplace en quelque sorte le retour arrière et le traitement en parallèle : la stratégie de regard en avant. Il propose un examen des mots suivants, des structures entières suivantes et de leur description sémantique pour résoudre les ambiguïtés syntaxiques locales et certains problèmes de traitement sémantique (surtout le choix du sens des mots).

II-1-4) Stratégies de résolution de problèmes

[Haton & al., 91] opposent deux méthodes d'exploration de l'espace de recherche en résolution de problèmes : un parcours systématique vs un parcours heuristique de l'espace de solutions.
Le parcours systématique de l'ensemble des solutions s'effectue à l'aide des méthodes classiques de résolution de problèmes : en largeur d'abord ou en profondeur d'abord.
Pour [Bratko, 88], le parcours **en profondeur** est le plus simple à programmer mais il engendre des problèmes de boucles informatiques. Des stratégies doivent être mises en place pour éviter les boucles : limiter la profondeur du parcours ou détecter le passage sur des sommets déjà visités. En revanche, le parcours **en largeur** est plus complexe à implémenter car il nécessite la mémorisation d'un ensemble de chemins candidats.
Le problème de ces deux types de parcours systématique est l'explosion combinatoire pour les grands espaces d'états ; il faut alors se tourner vers des "parcours heuristiques".
Le principe heuristique de la recherche du **meilleur d'abord** est de guider le processus de parcours, en choisissant à chaque fois le sommet le plus prometteur selon les estimations heuristiques. Evidemment, dans le cas de problèmes complexes, la difficulté est de trouver une "bonne" fonction heuristique. Une étude rigoureuse d'algorithmes de parcours sélectif a été réalisée par [Pearl, 90]. L'auteur définit l'heuristique de la manière suivante :

> «Les heuristiques sont des critères, des principes ou des méthodes permettant de déterminer parmi plusieurs lignes de conduite, celle qui promet d'être la plus efficace pour atteindre un but. Elles représentent des compromis entre deux exigences : le besoin de rendre de tels critères simples et, en même temps, le désir de les voir établir une distinction entre les bons et les mauvais choix.»

II-1-5) Exemples d'algorithmes d'analyse

Nous présentons deux algorithmes classiques d'analyse pour des grammaires hors-contexte [Aho & Ullman, 72] :

- **Algorithme de Cocke**

Cet algorithme est ascendant : partant de la chaîne, il cherche à remonter à l'axiome de la grammaire. Il construit simultanément toutes les analyses que la grammaire associe à la chaîne à analyser. Son principal inconvénient est qu'il est un algorithme combinatoire (c'est-à-dire qu'il effectue l'exploration de toutes les voies possibles) qui n'utilise aucune méthode de prédiction (méthode destinée à éviter des solutions partielles qui sont vouées ultérieurement à l'échec).

- **Algorithme d'Earley**

Cet algorithme est, à l'origine, descendant car les structures sont construites en partant de l'axiome (par la suite, des versions ascendantes de cet algorithme ont été développées). Il génère simultanément toutes les structures syntaxiques. Il a la particularité de pouvoir être prédictif car il est possible de tenir compte des n symboles suivants. Ainsi cet analyseur ne construit que les structures compatibles avec les n symboles suivants. Le paramètre peut être quelconque, mais bien évidemment, si l'on choisi n=0 la prédictivité n'intervient pas. En général, on choisit n=1, ce qui donne de bons résultats.

II-2) Algorithmes vs heuristiques

Notre objectif est de montrer les limites de l'utilisation d'algorithmes "classiques" pour un système d'analyse, et de proposer la mise en œuvre d'heuristiques linguistiques.

- **Limites des algorithmes classiques**

Les analyseurs travaillant sur la langue en général et à vocation non déterministe qui utilisent des algorithmes classiques ne permettent pas de traiter efficacement les subtilités de la langue. D'une part, ils sont confrontés au problème de l'explosion combinatoire. Et d'autre part, ils prennent difficilement en compte tous les phénomènes linguistiques de la langue (coordination, ellipse,…). Même les analyseurs prédictifs (comprenant à l'intérieur d'un même module des éléments pour aider l'analyse) ne fournissent pas de résultats probants car ils se heurtent aux mêmes types de problèmes.
Nous pensons qu'une première étape pour l'amélioration de la qualité et des performances de ces analyseurs réside dans le développement d'heuristiques linguistiques. Il faut de plus rechercher des architectures informatiques plus performantes (cf. [Stéfanini, 93]) ou mettre en place des techniques connexionnistes (cf. [Madec, 96]) qui permettent de réduire la production d'ambiguïtés inhérente à tout système général d'analyse.

- **Utilisation d'heuristiques linguistiques**

Face aux inconvénients de l'utilisation d'algorithmes (même prédictifs) pour traiter une application aussi complexe que l'analyse d'une langue, la mise en place d'heuristiques s'est développée. En effet, les heuristiques sont difficilement intégrables dans des algorithmes classiques. Par contre, elles sont souvent utilisées dans les systèmes experts en intelligence

artificielle. L'utilisation d'heuristiques pour l'analyse a déjà été proposée notamment par [Yang, 87], [Stock & al., 88].
Nous dirons qu'une heuristique est une méthode d'exploration qui procède par des évaluations successives et hypothèses provisoires. C'est une méthode de recherche fondée sur l'approche progressive d'un problème donné. Dans le cadre de l'analyse automatique d'une langue, nous considérons l'utilisation d'heuristiques linguistiques dans un analyseur comme des stratégies aidant au traitement d'ambiguïtés et de phénomènes linguistiques complexes ; ces stratégies correspondent à des critères de choix fondés sur des bases linguistiques.
De plus, l'amélioration de la qualité et des performances des analyseurs se situe, notamment, dans la recherche d'architecture informatiques plus efficaces. En effet, pour tirer profit au maximum de l'utilisation de plusieurs heuristiques, il est nécessaire de mettre en place une architecture adéquate : une architecture distribuée. Une telle architecture permettant la multi-expertise facilite la définition de différentes heuristiques linguistiques adaptées aux besoins de chaque expert.

II-3) Architecture séquentielle vs distribuée

Nous avons déjà évoqué certaines limites de l'utilisation d'une architecture séquentielle, pour un système d'analyse du français. Une architecture distribuée permet, de plus, de pallier de nombreux inconvénients des analyseurs que nous venons d'exposer ; elle permet ainsi de mieux traiter l'ambiguïté en vue de sa résolution.
L'utilisation d'une architecture distribuée rend possible le développement d'heuristiques plus adaptées en permettant aux agents d'obtenir l'information nécessaire au moment opportun.
Ainsi, l'interaction ou l'échange de connaissances entre les agents linguistiques, leur permet d'éliminer rapidement les solutions parasites dans la mesure où il peuvent disposer d'informations provenant d'autres agents linguistiques qui leur sont nécessaires à un moment donné. De plus, il est possible de développer des modules concurrents qui permettent d'utiliser plusieurs méthodes afin de confirmer ou d'infirmer des hypothèses. Par contre, il ne sera pas nécessaire de faire des choix (i.e. de privilégier certaines solutions par rapport à d'autres) si aucune information pertinente n'est présente pour décider. En d'autres termes, nous n'aurons pas recours à des choix arbitraires dénués de fondements linguistiques pertinents.

Nous allons maintenant exposer les choix informatiques qui représentent notre contribution par rapport à ceux de TALISMAN.

II-4) Choix de résolution informatique de conflits linguistiques

Nous établissons un rappel des méthodes informatiques sur lesquelles nous nous sommes fondé au vu des différentes stratégies possibles que nous venons d'exposer. Puis nous présentons les améliorations informatiques envisagées.

II-4-1) L'existant

L'architecture TALISMAN est non déterministe dans la mesure où elle doit rendre compte des ambiguïtés inhérentes à la langue. L'analyse morphologique utilise une méthode de désambiguïsation linguistique à l'aide de règles contextuelles. L'analyse syntaxique est effectuée à partir de l'algorithme d'Earley et utilise donc un analyseur descendant prédictif. L'architecture distribuée, du type "système multi-agents", permet la mise en œuvre d'heuristiques. Le système utilise une *communication directe* entre agents par envoi de messages, régulés par le protocole de communication de [Sian, 91]. Le *contrôle mixte* défini, utilise les lois locales et globales proposées par [Minsky, 91a].

Les principaux problèmes informatiques d'une architecture distribuée sont la définition et la gestion de la communication et du contrôle. Ces derniers sont au cœur des problèmes multi-agents : c'est eux qui permettent, s'ils sont bien définis, d'assurer la cohérence et le bon déroulement du système, ainsi que la résolution des conflits. Nous allons donc étudier comment améliorer la résolution de conflits par le biais de la communication et du contrôle.

II-4-2) Améliorations envisagées

Afin d'améliorer la résolution informatique de conflits linguistiques, nous allons développer les possibilités d'échanges d'information entre agents.

Pour résoudre les conflits dans un système multi-agents, il existe deux possibilités : utiliser des protocoles de communication ou mettre en place une forme de contrôle.

II-4-2-1) Utilisation d'une forme de contrôle

Nous distinguons différentes formes de contrôle :

- **Contrôle centralisé par un agent**

Cette méthode d'utilisation d'un agent superviseur est souvent utilisée dans la littérature des SMA. Par exemple, [Kreifelts & Von Martial, 91], introduisent un agent "Coordination" qui est le médiateur entre tous les autres agents. Les agents planifient leurs activités puis coordonnent leurs plans grâce à la communication ; pour cela ils envoient leurs plans à l'agent Coordination. Ce dernier cherche les actions nécessitant une négociation et supervise les négociations. Mais, comme le soulignent les auteurs, les compétences de l'agent Coordination peuvent être distribuées dans plusieurs agents (pour plus de détails cf. Annexe-Présentation de protocoles de communication).

Nous pensons que l'utilisation d'un agent Coordination est d'abord coûteuse en envois de messages, de plus, nous n'en voyons pas l'utilité dans TALISMAN II dans la mesure où il nous paraît difficile de définir un agent capable de superviser l'ensemble des agents linguistiques.

- **Contrôle réparti dans chaque agent**

La mise en place d'un interpréteur de messages est utilisée par [Maruichi & al., 90] ; ils introduisent la notion d'interpréteur de messages dans la réalisation d'un système orienté objet. Dans ce système un agent a son propre interpréteur qui gère les messages et la file de messages, les tâches et les protocoles.

Un interpréteur permettra donc de ne traiter et d'ordonner que les messages possédant certaines caractéristiques de sélection propre à l'agent (cf. Figure 6).

Quand un message arrive à un agent, le message est ajouté à la file d'attente. L'interpréteur de messages cherche un message (en fonction de certains critères) dans la file, le récupère et fait appel aux méthodes qui conviennent. Cette sélection des messages constitue l'autonomie d'un agent.
Le rôle principal de l'interpréteur de messages est donc de gérer la file d'attente de messages.

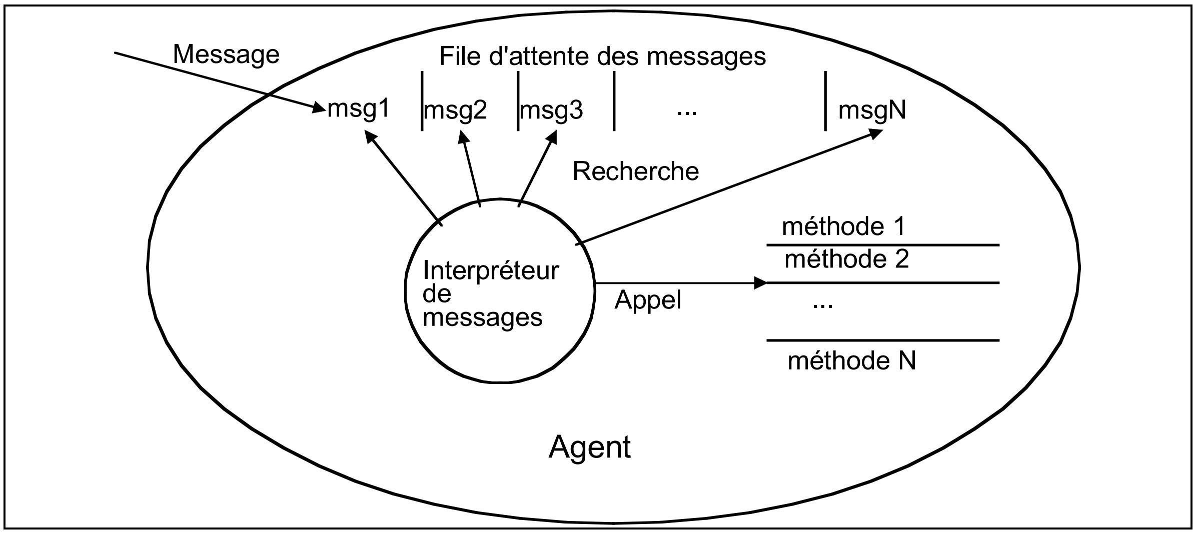


Figure 6 : Interpréteur de messages

Ce mécanisme d'interpréteur de messages relativement complexe est intéressant, mais pas nécessaire dans le cadre de notre application. En effet, nous pensons qu'il n'est pas utile de donner des priorités à des messages (pourquoi lire un message avant un autre ?) ; de plus, la définition de critères de priorité pour les messages est difficile à instaurer. Ainsi, dans TALISMAN II, les agents du système lisent les messages au fur et à mesure de leur réception, et savent effectuer les traitements nécessaires.

• Contrôle externe à tout agent

L'utilisation de lois a été proposée par [Minsky, 91a, 91b] pour faire communiquer des agents qui ont été développés séparément et qui ne connaissent pas forcément le protocole de communication utilisé dans la société d'agents ; les lois permettent aussi de vérifier l'application des protocoles de base et de donner les capacités de communication de la société d'agents.
Une loi est donc une fonction du type send (s,m,t) → deliver (m1, t1) ; fail.
où s est l'objet expéditeur, t est l'objet destinataire, m est le message envoyé, m1 est le message livré, t1 est le nouvel objet destinataire.
Minsky décompose donc l'envoi d'un message en deux étapes pour permettre à la loi du système d'intervenir pour changer le message (le destinataire ou le contenu) ou pour le bloquer (cf. Figure 7).

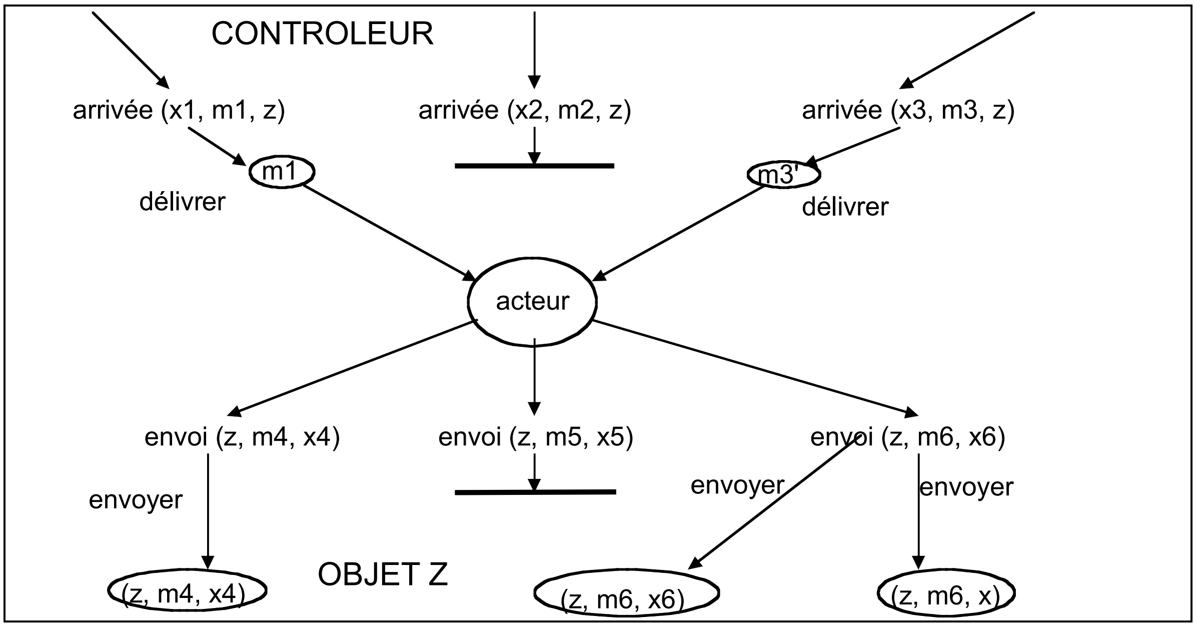


Figure 7 : Utilisation des lois

L'utilisation de deux types de lois (des lois globales pour gérer le contrôle du système et des lois locales pour gérer le contrôle local) avait été proposée dans TALISMAN. Cependant nous ne voulons pas les réutiliser pour différentes raisons. D'une part, dans TALISMAN II, nous considérons que les agents connaissent les protocoles de communication, donc les lois n'ont pas besoin d'être utilisées pour rendre possible la communication entre deux agents. D'autre part, les lois ont aussi pour vocation de réguler les communications entre agents. Mais, nous ne voyons pas comment définir une loi de priorité entre deux agents linguistiques dans la mesure où un agent n'a pas toujours raison par rapport à un autre. Les lois de modification du contenu d'un message (géré par la fonction deliver) ne sont pas nécessaires car nous ne voyons pas sous quels critères ni comment les réaliser dans une application telle que l'analyse du français.

II-4-2-2) Protocoles de communication

Un protocole de communication est un ensemble de règles contenant l'enchaînement des interactions possibles lors d'une conversation entre agents.
Les protocoles de communication permettent donc la coopération entre agents et ils fournissent des stratégies de résolution de problèmes ou de résolution de conflits engendrés par la cohabitation de plusieurs agents dans un même système.
Nous pensons que, si les protocoles de communication sont bien définis et adaptés aux besoins en communication du système, l'utilisation d'un contrôle décentralisé est nécessaire à la bonne marche du système.

L'utilisation des protocoles de communication nous semble un bon moyen pour permettre la coopération et la résolution de certains conflits. Nous étudions donc, dans la section suivante, les protocoles de communications existants. Une comparaison de ces protocoles nous permet de dégager des critères nous semblant important avant de présenter nos propres protocoles de communication.

III) Synthèse de protocoles de communication

Nous avons montré que l'architecture TALISMAN, utilisant une communication directe entre agents, nécessite une structuration précise de la communication afin d'éviter toute anarchie. De plus, la communication directe entre agents est un moyen qui peut devenir coûteux en temps ; elle est donc à utiliser avec parcimonie. L'utilisation de protocoles de communication est un bon moyen pour réguler les échanges. Ils permettent de fournir des stratégies de résolution de problèmes engendrés par la cohabitation de plusieurs agents dans un même système.

Notre objectif est la définition de protocoles de communication adaptés à un système d'analyse du français écrit. Tout d'abord, nous définissons et donnons les objectifs d'un protocole de communication. Puis, nous effectuons une comparaison de plusieurs protocoles de la littérature de l'intelligence artificielle distribuée. Enfin, nous présentons le langage de communication et les protocoles de communication définis pour TALISMAN II.

III-1) Définition et objectifs d'un protocole de communication

Après une définition détaillée des protocoles de communication, nous exposons leurs objectifs : l'échange de connaissances et le traitement des conflits.

III-1-1) Définition

- **Définition générale**

Le terme de protocole [Rosenschein, 93] utilisé pour décrire des réseaux, renvoie à la structure des messages qui permet aux ordinateurs de transmettre des informations ; mais dans le domaine des SMA [Esfandiari & al., 95], un protocole d'interaction comporte un ensemble de spécifications permettant aux agents de savoir comment réagir à un message, ce qu'ils doivent attendre après l'envoi d'un message,…
Nous définirons un protocole de communication comme *un ensemble de règles contenant l'enchaînement possible des interactions lors des échanges de messages entre agents*.

- **Intérêt des protocoles**

Les protocoles de communication permettent de formaliser et de contrôler les échanges d'information pour réduire à la fois, les temps de communication, et la quantité d'informations échangées. La communication peut devenir un moyen coûteux qui augmente les délais de réalisation des tâches à effectuer. Il faut donc formaliser et contrôler les échanges possibles entre les agents : c'est le rôle des protocoles. Les protocoles imposent des règles à la communication et constituent donc une loi de comportement pour les agents.
En régissant les interactions, les protocoles permettent de communiquer à l'aide d'un langage commun réduit, diminuant le volume des données qui transitent, et les contraintes liées à l'utilisation d'un langage plus évolué par des agents indépendants.

- **Représentation des protocoles**

Les protocoles de communication sont souvent représentés par des graphes dont les différents composants n'ont pas toujours la même signification. Par exemple, pour [Demazeau & al., 94b], un protocole d'interaction est un graphe d'états prédéterminé dans lequel l'agent évolue en fonction des types de messages reçus ; alors que pour [Sian, 91], un protocole est un graphe qui donne les enchaînements possibles des primitives sans que les états des agents ne soient pris en compte. Les protocoles sont donc souvent difficilement comparables [Burmeister & al., 93], car chacun corrèle de différentes manières, des aspects de bas niveau (ex. la syntaxe des messages) et des aspects de dialogue de coopération.

III-1-2) Objectifs

Nous distinguerons les protocoles qui ont pour but l'acquisition de connaissances de ceux qui visent à la résolution de conflits.

III-1-2-1) Echange de connaissances entre agents

- **Définitions**

Pour [Chevrier, 93], les ***interactions*** d'un système représentent les activités des agents qui ont une influence sur le système et sur le comportement des autres agents.
Quant à la coopération, elle se rapporte à l'activité globale d'un ensemble d'agents. Ainsi, [Bouron, 93] considère que la ***coopération*** sur une proposition existe entre deux agents, si au moins un des deux agents est engagé dans la réalisation de cette proposition, dans la mesure où ils croient avoir un but commun.
Nous considérerons donc que l'acquisition de connaissances d'autrui passe par l'échange de connaissances donc par les interactions entre agents ; ces dernières permettant l'émergence de la coopération. Nous dirons que *la coopération est une activité consistant à échanger des informations entre agents dans le but de résoudre un problème local ou commun*.

- **Caractéristiques de la coopération**

Nous rejoindrons [Mandiau, 93] considérant que la coopération suppose un but commun entre les agents coopérants ; chaque agent reconnaît que le but de l'autre agent peut conduire à la résolution du but commun. La réalisation de ce but commun nécessite parfois la réalisation de sous-buts qui sont adoptés de la même manière.
La coopération doit permettre d'augmenter la vitesse de résolution des tâches en parallèle, d'augmenter l'ensemble ou la portée des tâches réalisables par le partage des ressources et d'augmenter la probabilité d'achever des tâches (fiabilité) par leur duplication et si possible par l'utilisation de différentes méthodes pour les réaliser.

III-1-2-2) Traitement des conflits

Les conflits sont au cœur des préoccupations des systèmes multi-agents. Nous commençons par étudier les types de conflit auxquels nous nous sommes intéressé ; puis nous étudions comment ces conflits peuvent être traités dans un SMA, et en particulier, nous exposons différentes méthodes usuelles pour leur résolution.

A) Etude des conflits

• Différents types de conflits

D'une manière générale, on distingue différents types de conflits : les conflits de buts, de résultats et de ressources [Béron & al., 95]. Nous dirons que les conflits de buts sont dus à l'incompatibilité des buts poursuivis par différents agents. Les conflits de résultats sont produits par l'existence de contradictions entre les résultats des différents agents. Ils sont dus au fait que les agents possèdent des connaissances incomplètes voir erronées. Quant aux conflits de ressources, ils se produisent lorsque des agents ont besoin d'une même ressource au même moment.

Dans le cadre de notre étude, nous ne nous intéressons qu'aux conflits de résultats (ou d'opinion), que nous appelons conflits linguistiques. Nous rejoindrons donc la définition de [Vignal, 94] : "*Un conflit apparaît lorsque deux agents ont des hypothèses contradictoires sur la partie commune de leur intervalle de travail.*" En général, un conflit suppose au moins deux agents ou deux groupes d'agents. Or, comme nous l'avons vu précédemment, nous considérons aussi comme conflit, les conflits en interne : les solutions multiples produites par un agent (même si la résolution de ces types de conflit diffère). Ces conflits linguistiques proviennent donc, de résultats locaux internes à un agent ou de résultats différents de plusieurs agents, à un moment donné. Les conflits de résultats à la fin du déroulement du système (lorsque ce dernier devient stable) ne sont pas traités puisque l'objectif du système est de fournir des solutions multiples tant qu'elles ne sont pas infirmées par un agent quelconque du système.

• Différentes conceptions des conflits

Les conflits sont perçus très différemment selon les auteurs ; en effet, tous n'essaient pas de les éviter ou de les résoudre, puisque pour certains, ils ont un rôle positif dans la mesure où ils empêchent la stagnation du système et permettent d'accéder à la solution [Galliers, 90], [Vignal, 94].

Nous nous contenterons de constater que l'apparition de conflits est inévitable et nous ne chercherons pas ni à les éviter, ni à les provoquer, ni à les résoudre systématiquement. Notre objectif est de résoudre les conflits apparaissant à un moment donné dans le système, dans la limite des possibilités (en d'autres termes, des connaissances) du système.

• Traitement des conflits

En général, on distingue plusieurs phases dans le traitement des conflits :

- Détection : un agent détecte un conflit lorsqu'il engendre plusieurs solutions pour un même problème, soit lorsqu'il reçoit une information en contradiction avec les siennes.
- Evaluation : cette phase consiste à déterminer si le conflit doit être résolu ou pas. Dans notre application, seuls les conflits créés par un agent qui produit plusieurs solutions concurrentes, feront l'objet d'un traitement en vue de leur résolution.
- Résolution : la résolution d'un conflit peut se faire de différentes manières.

B) Différentes méthodes de résolution de conflits

En ce qui concerne la résolution des conflits, les avis sont très partagés : nous avons les adeptes de la négociation (ou relâchement de buts), ceux qui préfèrent le mécanisme de vote et enfin ceux qui utilisent un contrôle supérieur. Nous noterons que la méthode de résolution de conflits nous semble fortement liée au type de conflit que l'on désire résoudre ainsi qu'au type d'application du système.

[Ferber, 95] distingue deux méthodes de résolution de conflit : par négociation ou par arbitrage. Nous décomposons la résolution par arbitrage en deux sous-types : utilisation d'une forme de contrôle (arbitrage centralisé) ou d'un mécanisme de vote (arbitrage décentralisé).

• Utilisation d'un mécanisme de négociation

Pour de nombreux auteurs (notamment pour [Lander & Lesser, 93]), la résolution de conflit est possible en utilisant la négociation ; ainsi pour [Béron & al., 95], la négociation est la technique la plus employée pour résoudre des vues incohérentes ou pour atteindre un accord sur la façon de travailler ensemble.

D'une manière générale, la négociation désigne une stratégie de résolution qui utilise le dialogue pour parvenir à un accord visant à résoudre des conflits de croyances ou de buts [Bouron, 93].

Nous adopterons la définition de [Chevrier, 93], "*la négociation est une activité qui consiste à échanger des informations entre agents afin d'aboutir à un compromis mutuellement acceptable*". Pour ce dernier, la négociation est un processus adaptatif, sûr et efficace.

Nous retiendrons que la négociation requiert plusieurs hypothèses [Lander & Lesser, 93] :

- les propositions sont engendrées par un ou plusieurs agents,
- les agents évaluent les propositions en fonction de leurs critères personnels d'acceptation de la solution, et ils fournissent un retour sur ce qu'ils "aiment" ou "n'aiment pas" des propositions faites,
- les agents peuvent jouer différents rôles dans le processus de négociation (manager, employer,…),
- la décision d'accepter ou pas une proposition est un processus démocratique.

• Utilisation d'un mécanisme de vote

L'utilisation du vote permet aux agents d'exprimer leur avis sur une proposition, la proposition obtenant le plus de "voix" ou dépassant un certain "seuil" sera retenue. Ce mécanisme est utilisé, par exemple, par [Ephrati & Rosenschein, 93].

Nous distinguons la négociation du mécanisme de vote dans la mesure où nous considérons que la solution obtenue par un mécanisme de négociation n'est pas toujours possible au début du processus de négociation (puisque les agents peuvent, par exemple, relâcher leurs contraintes au cours du déroulement de la négociation) contrairement à une solution obtenue par un mécanisme de vote.

• Utilisation d'un mécanisme de contrôle

Un mécanisme quelconque de contrôle peut être mis en place pour résoudre des conflits. Par exemple, les conflits peuvent être résolus par un protocole privilégiant une des croyances conflictuelles. Ces conflits peuvent aussi être résolus par l'abandon (ou suppression) d'une croyance conflictuelle au profit des autres croyances ; cet abandon peut être déterminé à partir d'un algorithme ou d'une décision arbitraire [Bouron, 93].

Nous citerons, à titre d'exemple, Minsky [Minsky, 91a] qui s'est intéressé à l'"imposition" (i.e. le contrôle) des protocoles dans les systèmes hétérogènes distribués évolutifs. Les lois permettent d'exprimer des types de communication relatifs à des résolutions particulières : apprentissage et résolution de conflits. Minsky décompose donc l'envoi d'un message en deux étapes pour permettre à la loi du système d'intervenir pour changer le message (le destinataire ou le contenu) ou pour le bloquer.

Ces lois ont été utilisées par [Stéfanini, 93], afin d'exercer un contrôle général sur le système ; de plus, des lois de priorité pouvaient être définies afin de résoudre des conflits spécifiques.

Nous nous intéressons pour notre application uniquement à la coopération entre agents, puisque les conflits sont résolus seulement lorsque les connaissances d'un agent le permettent. Nous allons donc étudier des exemples de protocoles de communication qui permettent aux agents d'acquérir des connaissances sur l'environnement et de résoudre leurs éventuels conflits, cela par l'intermédiaire d'envois de messages.

III-2) Synthèse de protocoles de communication

Nous allons étudier des exemples de protocoles de communication permettant l'acquisition des connaissances sur l'environnement et la résolution de conflits ; les protocoles explicités nous semblent intéressants par leur diversité (expression d'intention, apprentissage, coopération, allocation de tâches, résolution de conflits, négociation). Une synthèse peut être trouvée dans [Warren, 96].

Nous ne nous sommes intéressé qu'aux protocoles suffisamment détaillés ce qui explique que nous n'ayons pas étudié des protocoles tels que celui de Gaspar [Gaspar, 91] qui a défini 4 types différents de messages (*présentation*, *information*, *demande*, *réponse*) correspondant aux modes de communication de base entre agents.

D'autre part, de nombreux auteurs ([Osawa, 93], [Zlotkin & Rosenschein, 91], [Matsubayashi & Tokoro, 93]) introduisent dans leur approche les notions de coût, de valeur ou d'utilité ; mais ces notions ne nous semblent pas adaptées à notre système d'analyse automatique du français dont le but principal est d'arriver à une ou plusieurs solutions correctes quelque en soit le coût. De même, nous n'avons pas étudié les protocoles utilisant le relâchement ou le changement de but ([Werner, 90], [Zlotkin & Rosenschein, 91], [Matsubayashi & Tokoro, 93]) pour résoudre les conflits dans la mesure où dans notre application, les agents ont des buts qui ne doivent pas être remis en cause.

Nous ne nous sommes pas non plus attardé sur les mécanismes de vote ([Ephrati & Rosenschein, 93]) pour résoudre les conflits puisque nous désirons résoudre les conflits dans la limite des connaissances du système et garder ainsi les conflits liés aux ambiguïtés inhérentes à la langue.

Enfin, nous n'avons pas retenu pour notre étude des protocoles trop "spécifiques" comme le protocole de Berthet [Berthet, 91] qui permet l'introduction d'un nouvel agent dans un SMA, car l'insertion d'un nouvel agent ne fait pas parti de nos objectifs ; cependant ce protocole pourrait nous être utile par exemple pour l'introduction de nouveaux agents (comme l'agent Sémantique pas encore implanté). De même, nous n'exposons pas le protocole de négociation de [Vernadat & al., 93], dont le but est de provoquer une réunion entre différents sites (constitués de trois agents : un maître, un synchronisateur et un négociateur) pour qu'ils échangent leurs propositions et décident d'une position commune.

III-2-1) Présentation générale de protocoles de communication

Nous allons donc présenter d'une manière succincte les protocoles de communication ayant retenu notre attention[16] ; une présentation détaillée de ces protocoles est fournie en annexe (cf. Annexe- Présentation détaillée des protocoles de communication).

III-2-1-1) Protocoles d'échanges de Campbell et D'Inverno

- **Présentation**

J. Campbell et M. D'Inverno [Campbell & D'Inverno, 90] sont partis de l'hypothèse que tout agent entamant un échange de messages a une intention : tout échange est la conséquence d'un plan ou d'un but d'un agent. Ils ont défini 21 "tons" possibles dans un dialogue. Pour chacun de ces tons, les auteurs supposent possible la construction de protocoles. L'idée est qu'un agent voulant entamer une communication va choisir un des protocoles. Les auteurs introduisent pour la formalisation des échanges, des prédicats logiques et des axiomes.

- **Remarques**

Campbell et D'Inverno ont montré la capacité des protocoles de communication à décrire toutes les intentions possibles. Il semble cependant que parmi les 21 tons, tous n'ont pas réellement d'utilité (*bargaining* (marchander), *amusing* (amuser),…). De plus, aucune implémentation n'a été faite et elle paraît complexe (à cause des prédicats, des axiomes et des 21 protocoles).

III-2-1-2) Acquisition de connaissances de Sian

- **Présentation**

Sati Sian [Sian, 91] est parti du principe que les systèmes intelligents se définissent par leur capacité à s'adapter en apprenant de leur environnement. Chaque agent utilise ses informations locales pour inférer des hypothèses partielles et ensuite interagit avec les autres pour modifier et/ou augmenter son degré de confiance en ses hypothèses.

Le langage développé par Sati Sian est un langage commun comportant 9 primitives : *assert* (affirme), *propose* (propose), *modify* (modifie), *agreed* (accord), *disagree* (désaccord), *noopinion* (sans opinion), *confirm* (confirme), *accept* (accepte), *withdraw* (rejet).

- **Remarques**

L'intérêt principal de ce protocole est qu'il permet d'accroître la performance des agents qui coopèrent. Les agents coopèrent car ils n'ont pas d'informations complètes ; ainsi, l'interaction permet d'accroître le degré de confiance dans une hypothèse acceptée.

Un inconvénient est que le système d'apprentissage comprend un temps d'exécution élevé car il est coûteux en envoi de messages. L'acquisition et l'amélioration de la qualité des informations se fait ici en commun. Cela implique une implémentation assez complexe des capacités de raisonnement.

III-2-1-3) Protocole d'apprentissage MOSCA

---

[16] Nous avons choisi d'ordonner la présentation des protocoles par leurs spécificités.

• Présentation

Le protocole MOSCA [Reitz, 92] modélise une situation typique d'apprentissage. L'apprentissage est réalisé par l'interaction de cinq différents types d'entités : le Maître, l'Oracle, la Sonde, le Client et l'Apprenti. Les différents types de message que les agents peuvent s'envoyer correspondent aux interactions entre les différents éléments intervenant dans un processus d'apprentissage automatique.

L'Apprenti est un résolveur de problèmes qui possède un échantillon de problèmes résolus à partir duquel il travaille. Les cinq rôles sont à partager entre deux acteurs : la machine et son usager.

• Remarques

Le protocole MOSCA a été utilisé [Guin & al., 93] pour l'apprentissage d'un élève par une interaction didactique entre l'enseignant artificiel et l'élève (i.e. l'Apprenti). En effet, l'environnement proposé se prête à la modélisation de ce type de situation d'apprentissage. La modélisation des cinq rôles différents ne nous paraît cependant pas aisée. La spécification, en particulier, des connaissances du Maître pouvant juger toute solution, est difficile.

III-2-1-4) Smith et Davis : Contract net protocol

• Présentation

Ce protocole de réseau de contrat [Smith & Davis, 83] permet une allocation dynamique et distribuée des tâches. Le manager va décomposer ses tâches en sous-tâches puis il va faire un appel d'offres sur une tâche. Les agents susceptibles d'effectuer cette tâche vont lui faire des propositions. Le manager attribuera la tâche au travailleur de son choix (en fonction de certains critères) et le contrat sera établi.

• Remarques

Ce protocole reproduit donc la négociation entre agents sous la forme de l'offre et de la demande ; il permet de formaliser et contrôler la répartition des sous-tâches entre les agents constituant le système. Ce protocole de résolution de problèmes permet d'obtenir un comportement coopératif entre les agents en s'appuyant sur la métaphore du système de l'offre et de la demande. Un réseau d'agents se construit ainsi dynamiquement pour résoudre une tâche globale. Le contrôle est distribué car un même agent peut être travailleur ou manager pour des tâches différentes.

III-2-1-5) Protocole hiérarchique de Durfee et Montgomery

• Présentation

Dans ce protocole [Durfee & Montgomery, 90], chaque agent possède des comportements associés à la réalisation de son but. Ces comportements sont composés de différentes informations (réparties dans les six dimensions suivantes : *qui*, *quoi*, *quand*, *où*, *comment*, *pourquoi*) et sont organisées de manière hiérarchique. Les auteurs ont défini un protocole qui a été utilisé pour le problème général de la détection et de la résolution de conflits de ressources par des robots. Les robots construisent une liste indiquant leur hiérarchie en fonction des priorités de leur but. Le robot de la plus haute autorité envoie l'information concernant son comportement le plus abstrait aux autres robots (inférieurs dans la hiérarchie sociale). Chaque robot (inférieur) compare alors ses informations locales avec ce qu'il a reçu et recherche l'existence de conflits ; un conflit se produit si deux robots sont dans une même région (*où*) dans le même intervalle de temps (*quand*). Quand le robot supérieur a reçu un message de tous les agents l'informant qu'il n'y a plus de conflits, il passe le contrôle au prochain robot dans la hiérarchie sociale. Ce mécanisme se poursuit jusqu'à ce qu'il n'y ait plus de robots inférieurs ; les conflits sont alors tous résolus.

• Remarques

Ce protocole permet une forme de coopération appropriée aux domaines où, d'une part, les agents ne savent pas, a priori, avec qui ils vont interagir, et dans lesquels, d'autre part, il est possible d'établir une hiérarchie sociale entre agents.

III-2-1-6) Protocole de Kreifelts et Von Martial

• Présentation

Pour ces auteurs [Kreifelts & Von Martial, 91], la négociation est pratiquée par des conversations avec un agent "Coordination" qui est le médiateur entre les autres agents.

Les agents planifient leurs activités puis coordonnent leurs plans grâce à la communication ; pour cela ils vont envoyer leurs plans à l'agent Coordination. Ce dernier va chercher les actions nécessitant une négociation. Les agents négocieront par l'intermédiaire de l'agent Coordination donc indirectement.

Le protocole de communication (ou règles de conversation) permet de relier les états et les types de messages. Ainsi, le modèle de conversation a trois composants : l'état de la conversation, le type de messages que l'on peut échanger, les règles de la conversation qui spécifient, suivant l'état de l'agent, quel type de message il peut envoyer ou recevoir.

• Remarques

Ce protocole est approprié à la résolution de conflits par l'intermédiaire d'un agent Coordination qui gère la négociation entre agents via la communication.

III-2-1-7) Un protocole adapté à la négociation : Chang et Woo

• Présentation

Ce protocole SANP[17] [Chang & Woo, 92] permet de résoudre les conflits à l'aide d'une négociation évoluée et flexible, et de prendre en compte le résultat des échanges. Ce protocole comprend de nombreux actes de dialogue dont : *make claim* (requête), *agree* (accord), *withdraw* (rejet), *argumentative attack* (attaque argumentée), *offer compromise* (offre d'un compromis),…

Il comprend six principales étapes : situation initiale, demande et réponse, attaque, phase tactique, acheminement vers un accord, résultat final.

• Remarques

Ce protocole montre que les protocoles de communication sont des outils performants pour contrôler les échanges et guider les agents vers une solution finale sans contrôle centralisé. Cependant, ce protocole est complexe (langage de communication important et réponses possibles nombreuses), sa mise en œuvre paraît donc difficile.

III-2-1-8) Lander et Lesser : recherche négociée

• Présentation

---

[17] Speech Act based Negotiation Protocol.

Les auteurs [Lander & Lesser, 93] ont étudié le rôle de la négociation pour la résolution de conflits parmi des agents hétérogènes. Ils présentent un algorithme qui reconnaît explicitement et exploite les conflits. Cet algorithme de recherche négociée utilise le conflit comme source d'informations de contrôle pour diriger les activités de recherche des agents, dans leur quête d'une solution mutuellement acceptable. Une proposition ne représente pas la solution complète mais une partie de la solution qui interagit avec d'autres. La proposition devient solution quand elle est acceptée par tous les agents. Une proposition de base est créée à partir d'exigences : exigences locales imposées par la spécification du problème, et des exigences externes apprises d'autres agents.

- **Remarques**

Ce protocole permet de résoudre les conflits via la négociation ; il est simple grâce aux 5 opérateurs de base et aux 6 états possibles ; seules les issues suivantes sont possibles : obtention d'une solution complète acceptable, d'une solution infaisable ou d'une solution inacceptable (complète ou non).

III-2-1-9) Protocoles génériques de Burmeister, Haddadi, Sundermeyer

- **Présentation**

Les auteurs [Burmeister & al., 93] proposent des protocoles de coopération génériques fondés essentiellement sur les actes de langage. Les messages sont préparés et utilisent des procédures d'envois. Les protocoles sont représentés d'une manière graphique ; le début représente le nom du protocole avec quelques paramètres et l'arbre étiqueté représente les étapes possibles du dialogue dans ce protocole. Les nœuds de l'arbre représentent les états du dialogue, les liens représentent les transitions d'un état du dialogue à un autre, et les branches correspondent aux transitions alternatives. La racine du nœud d'un arbre de protocole permet de le caractériser. Les autres nœuds sont des appels à d'autres protocoles (sous-protocoles). Les auteurs ont défini trois protocoles : *informing* (informer), *querying* (demander) et *commanding* (commander). Ces protocoles peuvent être utilisés tels quels pour de simples interactions ou peuvent faire parti de protocoles plus complexes.

- **Remarques**

Les auteurs ont présenté des protocoles génériques, construits à partir des types de message et des procédures de traitement des messages. A partir des trois protocoles primitifs, il est aisé de construire d'autres protocoles plus complexes.

L'étude détaillée de ces protocoles de communication (cf. Annexes), nous permet d'en effectuer une comparaison.

III-2-2) Comparaison des protocoles étudiés

Les protocoles de communication étudiés sont assez génériques et représentatifs des protocoles de coopération et de résolution de la littérature de l'intelligence artificielle distribuée. Notre objectif est de définir, à partir de l'étude comparative des protocoles, des protocoles de communication adaptés en linguistique.

III-2-2-1) Critères de comparaison

[Zlotkin & Rosenschein, 93], [Rosenschein, 93] ont élaborés des critères nécessaires aux mécanismes d'interaction dont : une distribution du contrôle (pas de décisions centralisées), une symétrie des agents (les agents ont des rôles similaires dans le processus d'interaction), efficacité des protocoles (pas de ressources gaspillées, ni d'utilité perdue), simplicité (afin d'éviter un coût en communication trop élevé).

Dans le cadre de TALISMAN II, nous avons retenu certains critères concernant les protocoles de communication :

- ils n'ont pas besoin d'être particulièrement complexes (contrairement à celui de [Chang & Woo, 92] comportant 6 phases),
- ils doivent permettre de respecter l'autonomie des agents (pas comme dans le protocole de [Kreifelts & Von Martial, 91]),
- le coût en communication ne doit pas être trop élevé (contrairement aux communications entre tous les robots supérieurs et inférieurs de [Durfee & Montgomery, 90]),
- le langage de communication ne doit pas être complexe (contrairement aux 22 actes de [Chang & Woo, 92]), il doit surtout répondre aux besoins de nos agents linguistiques.

III-2-2-2) Comparaison des protocoles

Nous avons comparé les protocoles de communication étudiés, permettant la coopération et la résolution de conflits, en fonction des critères définis ci-dessus (cf. Tableau 3).

	Particularités des protocoles	**Simplicité des actes de langage**	**Contrôle décentralisé**	**Coût possible en communication**	**Simplicité d'implémen--tation**
Campbell et D'Inverno	Expression d'intentions	complexe (21 tons, pas tous détaillés)	oui	assez élevé (changement de protocoles entre deux agents)	non (21 protocoles à développer)
Sian	Apprentissage et coopération	+ ou - (9 actes)	oui	élevé (l'avis de tous les agents est nécessaire)	oui
Reitz	Apprentissage et changement de rôle	simple (6 actes)	+ ou - (maître)	moyen (vérification par le Maître)	oui
Smith et Davis	Allocation de tâches	simple (4 actes)	oui	moyen (appel d'offre pour effectuer une sous-tâche)	oui
Durfee et Montgomery	Résolution de conflits avec hiérarchie sociale	non détaillés	oui (hiérarchie sociale variable)	élevé (échanges entre supérieurs et inférieurs)	non (décomposition comportements et hiérarchie)
Kreifelts et Von Martial	Coopération et négociation	+ ou - (12 actes)	non (agent coordination)	assez élevé (communication avec agent coordination)	moyenne (phase de re-négociation)
Chang et Woo	Coopération et négociation	complexe (22 actes)	+ ou - (autorité supérieure)	assez élevé (nombreux actes possibles entre deux agents)	non (6 phases)
Lander et Lesser	Coopération et négociation	simple (5 actes)	oui	élevé (avis de tous et	oui

	Particularités des protocoles	**Simplicité des actes de langage**	**Contrôle décentralisé**	**Coût possible en communication**	**Simplicité d'implémen--tation**
				relâchement d'exigences)	
Burmeister, Haddadi et Sundermeyer	Généricité (coopération et négociation)	au choix (7 actes détaillés)	oui	variable (selon les protocoles spécifiés)	oui

Tableau 3 : Comparaison de protocoles de communication

• La particularité des protocoles

Les protocoles étudiés sont représentatifs de nombreuses spécificités existantes : *expression d'intentions* [Campbell & D'Inverno, 90], *apprentissage* [Sian, 91], [Reitz, 92], *allocation de tâches* [Smith & Davis, 83], *résolution de conflits par coopération* en utilisant le respect de la hiérarchie sociale [Durfee & Montgomery, 90] ou en utilisant la négociation [Kreifelts & Von Martial, 91], [Chang & Woo, 92], [Lander & Lesser, 93]. Quant à la méthode de conception de protocoles proposée par [Burmeister & al., 93], nous pensons qu'elle doit permettre de construire des protocoles de types divers.

Nous noterons que le rôle des agents est variable. Ainsi, certains auteurs établissent des hiérarchies entre agents [Smith & Davis, 83], [Durfee & Montgomery, 90] qui peuvent varier : un agent peut être, à un moment donné, manager ou supérieur et à un autre moment, contractant ou inférieur. Pour d'autres auteurs, la hiérarchie est figée, comme dans le protocole de [Kreifelts & Von Martial, 91] où l'agent Coordination supervise tout le système. Enfin, d'autres auteurs considèrent qu'à tout moment les agents peuvent jouer le même rôle et l'avis de chaque agent a le même poids ; c'est le cas par exemple, du protocole de Sian. Nous pensons que ce type d'approche est adapté aux agents d'un système d'analyse dans la mesure où nous ne choisissons pas d'établir une hiérarchie entre agents.

• Les actes de langage

Parmi les protocoles étudiés, *le nombre d'actes de langage proposés varie entre 4* [Smith & Davis, 83] *et 22* [Chang & Woo, 92]. Nous pensons que, quelque que soit l'application visée, l'utilisation de plus d'une dizaine d'acte de langage est excessive : les actes ne sont pas tous indispensables, ils augmentent la complexité du système (au niveau de la conception des protocoles, des coûts en communication et de l'implémentation du système), et il devient difficile de déterminer lequel utiliser.

En ce qui concerne les actes présentés par les différents auteurs, nous établissons une distinction entre les actes de langage qui renvoient à une action (une requête, un appel d'offre, une commande,…) de ceux qui expriment une opinion sur une proposition (propose, confirme, suppose,…). Ainsi, le protocole de Sian par exemple, utilise des actes de langage qui permettent de fournir une opinion sur une proposition. Par opposition, le protocole de Smith et Davis ne contient que des actes de langage relatifs à une action. Mais il nous paraît intéressant de pouvoir utiliser ces deux types d'actes de langage ; c'est le cas, par exemple, dans le protocole d'apprentissage MOSCA où les actes de langages disponibles permettent à la fois d'effectuer une action ("*requête*") et de formuler une opinion ("*critique*").

Par contre, il nous semble important que cette distinction entre ces deux types d'actes soit exprimée clairement lors de la définition d'un langage de communication.

• Le type de contrôle

La majorité (6 sur 9) des protocoles étudiés utilisent un contrôle *décentralisé*, même si un agent peut à un moment donné être le supérieur d'un autre (comme dans [Smith & Davis, 83] ou [Durfee & Montgomery, 90]). Nous trouvons aussi judicieuse la méthode adoptée par [Chang & Woo, 92] qui consiste à utiliser en règle générale un contrôle décentralisé tout en instaurant un appel possible à une autorité supérieure pour régler un problème. Quant à [Reitz, 92], il propose un contrôle des hypothèses de l'Apprenti par le Maître. Par contre, nous ne voyons pas bien l'intérêt d'utiliser un contrôle totalement *centralisé* par l'agent Coordination de [Kreifelts & Von Martial, 91].

• Le coût en communication

Bien entendu, l'évaluation (des coûts possibles en communication des protocoles étudiés) que nous formulons, est intuitive[18].

Ainsi, nous pensons que lorsque l'avis de tous les agents du système est demandé sur une hypothèse (comme dans les protocoles de Sian et de Lander et Lesser), le coût en communication devient *élevé*. D'ailleurs, nous pensons qu'en général tous les agents d'un système n'ont pas les capacités (ou connaissances) pour exprimer un avis sur une proposition. De même, dans le protocole de Durfee et Montgomery, les envois de messages entre l'agent supérieur et tous les agents inférieurs du système doit engendrer un coût en communication élevé.

D'autres protocoles engendrent un coût en communication *assez élevé* même si la communication ne s'effectue qu'entre deux agents. C'est le cas des protocoles de [Chang & Woo, 92] et de [Campbell & D'Inverno, 90] dans la mesure où le nombre d'actes de langage ou de protocoles est considérablement élevé. De même, dans le protocole de [Kreifelts & Von Martial, 91], le fait que tout échange entre agents passe par l'agent Coordination doit augmenter le nombre de communication (devenant ainsi assez élevé).

Finalement, les protocoles respectant un coût *moyen* en communication sont : celui de Smith et Davis (qui ne comporte que quatre actes de langage mais qui nécessite au début un envoi de l'appel d'offre à tous les agents) et celui de [Reitz, 92] (qui utilise 6 actes de langage et une interaction entre agents variable qui dépend de la répartition des rôles).

• L'implémentation

L'évaluation que nous formulons quant à la simplicité d'implémentation est fonction de la complexité des protocoles.

Nous retiendrons que trois des protocoles sont relativement complexes à implémenter pour des raisons diverses : le protocole de Campbell et D'Inverno (comportant 21 protocoles qui ne sont, pour la plupart, pas détaillés), celui de Durfee et Montgomery (nécessitant une décomposition des comportements et de la hiérarchie sociale, et, de plus, les actes de langage n'étant pas détaillés), et enfin le protocole de Chang et Woo (comportant 6 phases complexes et 22 actes de langage).

Nous considérerons que la difficulté d'implémentation du protocole de Kreifelts et Von Martial est relative dans la mesure où il utilise un nombre d'actes moyen (12) et comporte deux phases (une phase de négociation initiale et une phase de re-négociation).

III-2-2-3) Conclusion

En résumé, cette synthèse des protocoles nous permet de dégager les conclusions suivantes :

---

[18] Par ailleurs, pour nous, l'évaluation du coût n'est pas essentielle dans un prototype de recherche.

- Les protocoles de Campbell et D'Inverno ne sont pas tous explicités et les 21 tons ne répondent pas à nos besoins.
- Le protocole de Sian paraît très intéressant pour permettre la coopération entre agents tout en restant simple : le protocole est flexible et il utilise des actes de langage génériques.
- Le protocole de Reitz est adapté à l'apprentissage de type "enseignant-élève" ; mais selon l'application, la spécification des connaissances incontestables de "l'enseignant" est difficile.
- Le protocole de Smith et Davis est intéressant mais nous ne l'utiliserons pas puisque les agents linguistiques connaissent les compétences des autres agents avec leurs accointances.
- Le protocole de Durfee et Montgomery suppose une hiérarchie sociale entre agents et un découpage hiérarchique des comportements suivant leurs six dimensions. Ce protocole est surtout utile pour des systèmes dans lesquels les agents ne savent pas avec qui ils vont être amenés à interagir, ce qui n'est pas le cas de notre système.
- Aucun des trois protocoles de négociation ([Kreifelts & Von Martial, 91], [Chang & Woo, 92] et [Lander & Lesser, 93]) n'est adapté à un système d'analyse automatique du français écrit du fait de leur trop grande complexité par rapport à nos besoins. De plus, d'une manière générale, les protocoles de négociation existants sont souvent inflexibles [Béron & al., 95].
- Les protocoles présentés par Burmeister, Haddadi et Sundermeyer sont très intéressants mais très génériques. En fait, ils proposent simplement la construction de protocoles plus ou moins complexes à partir d'actes de langage.

Cette étude comparative de différents protocoles de communication nous a permis de dégager certains impératifs concernant les protocoles que nos désirons développer pour TALISMAN II. A partir de ces constatations, nous allons maintenant présenter les protocoles de communication que nous avons définis pour le système d'analyse automatique du français écrit : nous les nommerons des "protocoles linguistiques".

III-3) Elaboration de protocoles linguistiques

Notre objectif est de développer des protocoles de communication pour TALISMAN II. Nous allons, dans un premier temps, récapituler les critères nécessaires à la définition de protocoles linguistiques. Puis, nous exposons les protocoles linguistiques que ces critères nous ont amené à développer.

III-3-1) Choix de critères pour l'élaboration de protocoles linguistiques

• Utilisation de protocoles de coopération

Les protocoles linguistiques dont nous avons besoin doivent permettre la coopération entre les agents. Ces derniers ont besoin d'échanger des informations de nature linguistique. Les méthodes classiques de résolution de conflits (négociation, vote, contrôle supérieur), ne sont donc pas utiles pour notre système. En effet, toutes les connaissances sont distribuées au sein des agents du système et les conflits linguistiques seront résolus seulement si les connaissances le permettent. Par conséquent, la coopération suffit à la résolution des conflits linguistiques dans la mesure où les conflits persistants (i.e. conflits qui n'ont pu être résolus par aucun agent du système) doivent être conservés.

- **Modification du protocole de Sian**

L'étude comparative des protocoles de communication, nous permet d'envisager l'utilisation du protocole de [Sian, 91]. En effet, le protocole nous paraissant le plus intéressant pour un système d'analyse du français écrit est celui développé par Sati Sian puisqu'il permet la coopération entre les agents tout en restant relativement flexible. De plus, il utilise des actes de langage génériques et son implémentation est simple. Ce protocole de communication est donc en adéquation avec nos besoins.

Dans sa version originale, tous les agents doivent donner leur opinion sur une hypothèse, ce qui entraîne de nombreux envois de messages. Cela ne nous paraît pas utile dans TALISMAN II puisque tous les agents ne sont pas capables de donner leur avis sur un problème donné ; de plus les agents disposent de connaissances sur les compétences des autres agents. Un agent peut donc demander directement leur avis, aux agents concernés. Nous allons donc modifier en fonction de nos besoins, le protocole de Sian, utilisé dans la première version de TALISMAN [Stéfanini, 93].

- **Adaptation de la décomposition des protocoles**

De plus, nous nous inspirerons des protocoles génériques développés par [Burmeister & al., 93], dans la mesure, où la décomposition des protocoles en fonction de l'acte de langage initial nous paraît pertinente. Nous décomposerons donc, pour plus de clarté, le protocole de [Sian, 91] en plusieurs protocoles de communication suivant l'intention de l'agent émetteur initial.

Nous sommes donc amené, dans la partie suivante, à introduire les protocoles de communication adaptés à TALISMAN II (permettant la coopération et la résolution de certains conflits) en fonction des critères de choix que nous venons de définir. Bien entendu, nous exposons préalablement le contexte communicatif de ces protocoles : les messages (mode d'envoi, mode de réception et format) et le langage de communication retenu.

III-3-2) Messages et langage de communication dans TALISMAN II

La communication est primordiale dans un système multi-agents ; en effet, c'est un des éléments qui va permettre d'assurer la cohérence du système.

Nous allons étudier la communication directe entre agents telle qu'elle est définie dans TALISMAN II, et plus particulièrement, le format de messages retenu. Nous développons ensuite, le langage de communication nécessaire aux agents.

III-3-2-1) Communication par envois de messages

- **Hypothèses de TALISMAN II**

Dans le modèle proposé, la communication par envois de messages est supposée sans erreurs. Ainsi, nous partons de l'hypothèse qu'il n'existe aucun problème de transmission : lorsqu'un message est envoyé, il est supposé toujours arriver à son destinataire ; et il n'est en aucun cas altéré par sa transmission.

Quant aux agents, nous supposons de leur sincérité par rapport aux messages envoyés et, de leur crédulité vis à vis des messages reçus.
D'autre part, l'envoi de messages est réalisé de façon asynchrone et les agents n'ont pas à délivrer d'accusés de réception. N'importe quel agent peut envoyer un message n'importe quand. Les agents reçoivent les messages dans leur boîte aux lettres où ils sont empilés ; le premier message reçu sera le premier traité. La réception d'un message par un agent inactif a pour effet de l'activer ou de le réactiver. Un agent actif qui reçoit un message interrompt la réalisation de sa tâche courante pour essayer de répondre au message.

- **Format des messages de TALISMAN II**

La communication par envois de messages nécessite la définition précise du contenu d'un message. Le format des messages que nous allons utiliser est une simplification de la structure du langage proposée par [Demazeau, 95].
Dans notre système, un agent désirant envoyer un message utilisera le format suivant :

(Routage, Acte, Contenu)

Le **routage** est composé de l'identificateur de l'agent expéditeur (=expéditeur) afin d'aider à la compréhension du message et de permettre au destinataire de lui répondre. L'agent expéditeur devra déterminer les agents destinataires (=destinataire qui peut être un agent, un groupe d'agents ou l'ensemble des agents du système) à l'aide des connaissances dont il dispose sur les autres agents. Pour l'instant, ces connaissances, déterminées a priori, ne peuvent faire l'objet d'aucun raisonnement par l'agent. Si l'agent ne dispose d'aucune information, il pourra toujours envoyer le message à l'ensemble des agents du système.
L'**acte** du message représente l'intention de l'agent émetteur ; il sera développé dans la partie suivante.
Le **contenu** propositionnel est formulé dans le langage de représentation des connaissances (à ne pas confondre avec le langage de communication) propre à l'agent et à son expertise linguistique. Dans le cas où l'agent possède plusieurs solutions pour un même problème, il envoie l'ensemble des solutions possibles ; la première solution étant celle qui devra être jugée par les autres agents. Nous avons choisi d'effectuer ces envois multiples, d'une part, par souci d'économie d'envois de messages, et d'autre part, afin que l'agent récepteur ait la possibilité d'évaluer la solution proposée en regard des autres solutions possibles.

III-3-2-2) Langage de communication

Nous considérons la communication comme une action sur le monde extérieur découlant d'une intention de l'agent émetteur que le destinataire doit pouvoir comprendre sans ambiguïtés. Nous rejoignons ainsi le courant qui, actuellement en intelligence artificielle distribuée, s'inspire des études sur les actes de langage (conçus par [Austin, 62] puis [Searle, 69]) pour la formulation de la communication dans les systèmes.

- **Actes de langage**

La plupart des approches permettant de formaliser la communication entre agents reposent sur la théorie des actes de langage [Cohen & Levesque, 90], [Singh, 92], [Vanderveken, 94],…
La théorie des actes de langage introduit la pragmatique dans le discours, toute parole a un but : faire quelque chose ou produire un comportement. Le langage est alors un comportement intentionnel régi par des règles. [Searle, 69] a proposé une catégorisation des actes de langage, il distingue : l'acte locutoire (l'énonciation de l'acte, le fait de dire quelque chose), l'acte illocutoire (l'acte réalisé quand on dit quelque chose) et l'acte perlocutoire (l'effet obtenu par le fait de dire quelque chose, ce dernier pouvant être différent de l'effet attendu).

Les actes illocutoires peuvent être classés en 5 grandes classes [Searle, 79] : assertifs (soutenir, affirmer, conclure,…), directifs (demander, ordonner, réclamer,…), promissifs (promettre,…), expressifs (remercier, féliciter, s’excuser,…), déclaratifs (déclarer, définir, désigner,…).

Nous rejoignons [Colineau & Moulin, 96] qui considèrent que l'acte illocutoire est défini à la fois par sa force, qui détermine quel type d'acte est réalisé (*Est-ce que j'informe ? Est-ce que je demande ?…*) et par son contenu propositionnel (*le sujet de l'information*, *le sujet de la demande*,…). Cette décomposition rejoint [Populaire & al., 93] qui considère qu'un destinataire, pour comprendre l'intention de l'émetteur, doit pouvoir séparer, dans le message, la partie intentionnelle et le contenu purement propositionnel.

• Langage de communication

Le langage de communication doit être compréhensible par tous les agents donc commun et il doit permettre d’exprimer toutes les intentions qu’ils peuvent avoir.

En général, on distingue 4 grands types de message [Gaspar, 91] :

- message de présentation (*present*) : permettant à un nouvel agent de se présenter auprès des autres agents du système,
- message d’information (*inform*) : utilisé par un agent pour envoyer une information à d'autres agents du système,
- message de demande (*request*) : permettant à un agent d'envoyer une demande d'information ou d'action à d'autres agents du système,
- message de réponse (*answer*) : utilisé par un agent pour répondre à une demande d'information ou d'action.

Mais ces types de messages sont insuffisants pour représenter toutes les intentions des agents. Nous rejoindrons donc [Populaire & al., 93] en associant, à ces types d'actes, des forces d’élocutions représentant les différents tons du dialogue. Les forces d'élocution que nous avons retenues sont adaptées du langage de communication utilisé par [Sian, 91]. Notre langage de communication diffère de ce dernier par la distinction du type et de la force d'élocution.

• Langage de communication de TALISMAN II

Nous avons donc défini les différents types d'acte et les forces illocutoires dont nous avons besoin.

Les différents **types d'acte** que nous avons retenu pour TALISMAN II sont : *informer* (pour envoyer une information), *demander* (pour demander de l'aide) et *répondre* (pour répondre à une demande). Nous n'avons pas retenu le type d'acte "*present*" permettant à un nouvel agent de se présenter dans la mesure où, pour l'instant, nous n'envisageons pas l'introduction d'un nouvel agent. D'autant plus, que ce type de messages est réellement utile quand de nouveaux agents sont susceptibles d'être créés fréquemment dans un système ; ce qui n'est pas le cas de notre application d'analyse du français.

Les **forces illocutoires** nécessaires à TALISMAN II sont :

- *affirme* représente une assertion non modifiable,
- *propose* permet de proposer une nouvelle hypothèse,
- *modifie* donne une version modifiée de l’hypothèse précédemment proposée,
- *ignore* n'exprime aucune opinion sur une hypothèse,
- *confirme* permet de confirmer l’hypothèse,
- *rejette* permet de rejeter l'hypothèse proposée.

Pour notre application, nous avons éliminé du langage de Sian, la force *accept* (accepte) qui nécessite l'accord de tous les agents. De même, nous n'avons pas retenu les forces *agreed* (accord) et *disagreed* (désaccord) qui nous paraissent redondantes, respectivement avec *confirm* (confirme) et *withdraw* (rejet), puisque, pour l'instant, les agents du système confirment ou infirment une hypothèse seulement s'ils disposent de compétences le permettant. L'introduction de ce type de forces sera nécessaire avec une approche plus fine des connaissances des agents.

Les combinaisons possibles, entre le type d'acte et la force illocutoire, sont indiquées dans le Tableau 4.

TYPE	FORCE	SIGNIFICATION
Informer	affirme	Envoi d'une information
Demander	propose	Demande de coopération sur une proposition
	[]	Demande d'information (sans proposition) ou d'action
Répondre	affirme	Réponse à une demande d'information (ou d'action)
	ignore	Aucune réponse possible
	rejette	Rejet de la proposition
	modifie	Modification de la proposition
	confirme	Confirmation de la proposition

Tableau 4 : Combinaisons possibles entre Type et Force

Ainsi, une conversation peut débuter avec les actes suivants : Informer (affirme), Demander (propose) ou Demander ([]). La fin d'une conversation est possible avec : Informer (affirme), Répondre (affirme), Répondre (ignore), Répondre (confirme) ou Répondre (rejette).

Le contexte d'utilisation de ces différents actes sera exposé dans la partie suivante lors de l'illustration par des exemples linguistiques du fonctionnement des protocoles de communication définis dans TALISMAN II.

III-3-3) Présentation des protocoles linguistiques

Dans le système, nous avons adapté le protocole de Sati Sian [Sian, 91] en fonction du langage de communication que nous avons défini pour la coopération entre agents. Nous avons donc simplifié ce protocole et, pour plus de clarté, nous l'avons décomposé en trois protocoles : un protocole d'affirmation (pour envoyer des informations sûres), un protocole de demande d'information et un protocole de demande de coopération (utilisé par un agent lorsqu'il a engendré plusieurs solutions et émet une hypothèse) [Warren & Stéfanini, 96].

III-3-3-1) Protocole d'affirmation

Ce protocole (cf. Figure 8) permet à un agent d'envoyer ses résultats partiels ou complets aux agents concernés.

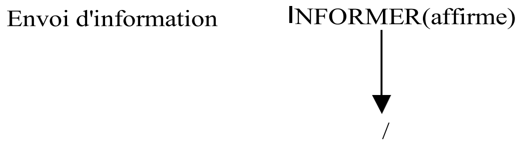


Figure 8 : Protocole d'affirmation

Exemple : "Sont-elles les anges ou les démons de cette histoire très compliquée ?"
L'agent Transf (expert, en particulier, en transformation d'interrogatives) reconnaît une question totale qu'il transforme (à l'aide des connaissances de l'agent Morph) et annote par QTO[19] ; il envoie ensuite ce résultat à l'ensemble des agents puisqu'ils sont tous concernés par la phrase à traiter :

```
nouveau_message (transf, broadcast, informer, affirme ,
   [phrase = "elles sont les anges ou les démons de cette histoire très compliquée", qto]).
```

III-3-3-2) Protocole de demande d'information

Ce protocole (cf. Figure 9) permet à un agent de poser une question précise à un (ou plusieurs) agent(s). Si l'agent destinataire sait répondre, il envoie un message du type "Répondre(affirme)", sinon il répond par "Répondre(ignore)".

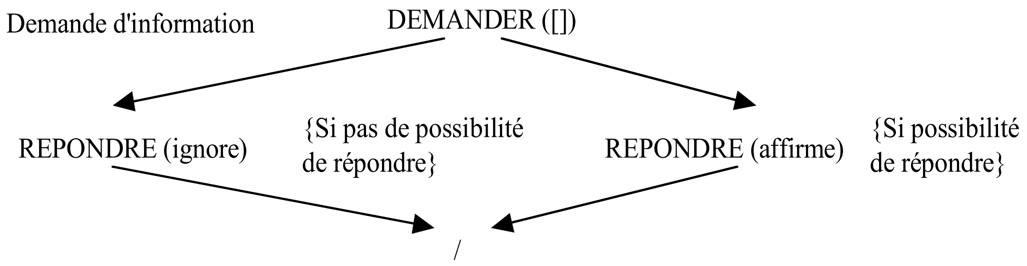


Figure 9 : Protocole de demande d'information

Exemple : "elles sont les anges ou les démons de cette histoire très compliquée"
L'agent Synt commence son analyse syntaxique de la phrase en se basant sur les formes non ambiguës de catégorie Verbe ; il va donc demander à la morphologie la liste des verbes non ambigus (cette liste pouvant être vide ou comporter un ou plusieurs verbes). Les messages suivants seront donc envoyés :

```
nouveau_message (synt, morph, demander, [], [liste-verb-non-amb=?]).
message (morph, synt, repondre, affirme, [liste-verb-non-amb= ["sont"]]).
```

III-3-3-3) Protocole de demande de coopération

Ce protocole (cf. Figure 10) permet à un agent de demander à un ou plusieurs autres agents de coopérer avec lui dans le but de résoudre le conflit qu'il a créé : il a engendré plusieurs solutions pour un même problème et les autres agents doivent confirmer ou non ses hypothèses (dans le cas d'une confirmation et du rejet d'une hypothèse par différents agents, le rejet de l'hypothèse sera retenu).

---

[19] QTO représente une question totale.

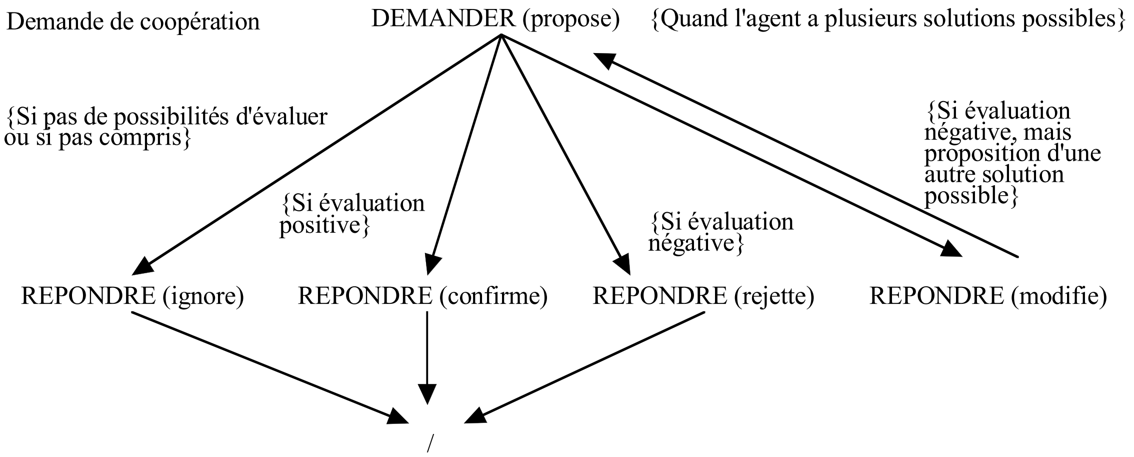

Demande de coopération
DEMANDER (propose)
{Quand l'agent a plusieurs solutions possibles}
{Si pas de possibilités d'évaluer ou si pas compris}
{Si évaluation positive}
{Si évaluation négative}
{Si évaluation négative, mais proposition d'une autre solution possible}
REPONDRE (ignore)
REPONDRE (confirme)
REPONDRE (rejette)
REPONDRE (modifie)
/

Figure 10 : Protocole de demande de coopération

Exemple : "il (Y) détient (V) le (Y/D) pouvoir (F/V) . (T)"
L'agent Morph (après analyse morphologique flexionnelle et désambiguïsation) envoie ses résultats (mentionnés ci-dessus) aux agent Segm et Synt. Cette phrase comporte deux verbes potentiels mais aucun introducteur de propositions. L'agent Segm propose donc à l'agent Synt de tenter l'analyse de cette phrase avec une seule proposition. L'agent Synt obtient l'analyse syntaxique en constituants et fonctionnelle de cette proposition ; il envoie donc un message de confirmation à l'agent Segm.
Ainsi, l'enchaînement des interactions se déroulera de la manière suivante :

```
nouveau_message (morph, [segm, synt], informer, affirme,
            phrase = ["il"= y, "détient"= v, "le"= [y,d], "pouvoir"= [f,v], "."= t]).
nouveau_message (segm, synt, demander, propose, nb_proposition = 1).
message (synt, segm, repondre, confirme, nb_proposition = 1).
```

Cette première partie nous a permis de présenter la problématique linguistique et informatique de notre étude. Nous avons présenté l'architecture TALISMAN sur laquelle nous nous sommes fondé. Nous avons étudié les méthodes classiques de résolution de conflits linguistiques, ce qui nous a amené à justifier nos choix : utilisation d'une architecture distribuée permettant la multi-expertise et la définition d'heuristiques. Nous avons présenté les protocoles de communication comme moyen de gérer les conflits linguistiques dans un système multi-agents. Aucun des protocoles de communication étudiés ne répondent aux besoins de TALISMAN II, nous avons donc été amené, à introduire nos propres protocoles linguistiques de communication.
Dans la seconde partie, nous étudions la gestion, voire la résolution de conflits linguistiques par les seules compétences d'un agent et par coopération entre agents communiquants via les protocoles de communication définis.

PARTIE II :

RESOLUTION DE SOLUTIONS MULTIPLES EN MORPHO-SYNTAXE

CHAPITRE III : RESOLUTION INTERNE AUX AGENTS MORPH ET SYNT

Les ambiguïtés inhérentes à la langue posent des problèmes à tout système de Traitement Automatique de la Langue (TAL), aussi bien au niveau linguistique qu'informatique. Les ambiguïtés provoquent des "conflits linguistiques" (i.e. plusieurs solutions linguistiques pour un même problème) qui nécessitent une méthodologie appropriée.

Cette seconde partie a pour objectif l'étude des ambiguïtés morpho-syntaxiques de la langue comme source de conflits. Nous montrons les solutions multiples qui sont résolues dans TALISMAN II : nous nous attachons, dans ce chapitre, aux solutions parasites éliminées par les agents Morph et Synt ; dans le chapitre suivant, nous étudions la résolution coopérative dans la société d'agents de notre architecture.

Introduction

D'après [Fuchs, 87b], "L'ambiguïté : une expression qui signifie plusieurs choses". [Fuchs, 96] distingue l'ambiguïté virtuelle, de l'ambiguïté effective en contexte. L'ambiguïté virtuelle peut être désambiguïsée par la suite de la phrase alors que l'ambiguïté effective persiste après l'analyse de la totalité de la phrase. L'auteur précise que l'ambiguïté varie suivant le type de récepteur : humain, linguiste ou machine.

Dans cette étude, notre objectif est de résoudre certaines ambiguïtés virtuelles de la machine. En d'autres termes, nous voulons éliminer certaines des solutions créées par un système de TAL et non attestées par la langue : notre objectif n'étant pas de résoudre les ambiguïtés linguistiques effectives en contexte. Les ambiguïtés virtuelles (ou "ambiguïtés machines" engendrées par le système) proviennent du modèle linguistique ou informatique adopté. Les ambiguïtés virtuelles liées au modèle linguistique sont la conséquence, entre autre, de la structuration du dictionnaire et de l'utilisation d'une grammaire "générative"[20] qui fournit trop de structures là où le récepteur humain ne perçoit pas l'ambiguïté (dans la mesure où il comprend la phrase).

Nous dirons qu'il y a ambiguïté, chaque fois que plusieurs solutions sont possibles pour un même problème ou lorsqu'une solution relève de plusieurs interprétations possibles. L'ensemble des exemples d'ambiguïtés est issu d'un corpus journalistique. Ces ambiguïtés apparaissent à chaque niveau différent conceptuel de représentation : morphologique, syntaxique, sémantique et pragmatique. Nous verrons aussi que certaines ambiguïtés peuvent interagir sur plusieurs niveaux en même temps, en particulier, avec les phénomènes linguistiques complexes.

---

[20] Une grammaire générative est capable de prévoir (i.e. d'énumérer et de décrire) toutes les phrases possibles de la langue. On dit qu'une telle grammaire (indispensable pour le traitement de la langue en général) engendre un ensemble infini de phrases.

Nous considérerons que l'analyse d'une langue écrite se décompose en quatre niveaux de représentation des connaissances : morphologie, syntaxe, sémantique et pragmatique. Nous allons illustrer, par des exemples issus d'un corpus journalistique, différents types d'ambiguïtés pouvant intervenir à différents niveaux de l'analyse. Nous présentons, ici, les ambiguïtés relevant des niveaux morphologique et syntaxique ; certaines ambiguïtés des niveaux sémantique et pragmatique sont présentées dans le chapitre 6.
Nos objectifs sont de déterminer l'origine des ambiguïtés et de montrer comment les différents cas d'ambiguïtés sont résolus (ou tout au moins gérés), dans TALISMAN II. Nous développons des cas d'ambiguïtés pouvant être résolus par les compétences d'un seul agent, notamment des agents Morph et Synt.

I) Etude du corpus

• Etude de phrases
Nous avons retenu 77 phrases d'un corpus journalistique[21]. Ces phrases comprennent au total 1301 formes qui représentent seulement 617 formes différentes. Nous avons choisi des phrases relativement simples (45 de ces phrases ne comportent qu'une seule proposition) ; en d'autres termes, pas trop longues. En effet, chacune des phrases comporte en moyenne environ 17 formes (1301/77 = 16,89 formes par phrase).
Voici quelques exemples de phrases :

```
Phrase 1 : La seule incertitude concerne l'ordre hiérarchique dans
           lequel les deux hommes seront sans doute appelés à
           travailler ensemble.
Phrase 2 : Forte de ses positions majoritaires, l'UC, pour
           l'instant, fait la sourde oreille.
Phrase 3 : Toute l'opposition, du centre droit à l'extrême gauche,
           est visée.
Phrase 4 : La veille, des unités de la garde présidentielle
           s'étaient emparées, souvent brutalement, d'une
           quarantaine de responsables et de militants politiques.
Phrase 5 : Les rues de Port-au-Prince sont restées calmes dimanche
           et aucun dispositif militaire particulier n'a été mis en
           place.
...
```

L'ensemble de ces phrases sont présentées dans les annexes (cf. Annexe - Phrases tests pour la désambiguïsation morphologique).

• Etude des formes du corpus
Sur 617 formes différentes, la plupart (représentant 507 formes) n'apparaissent qu'une fois dans le corpus. Ainsi, seulement 110 formes sont présentes plus d'une fois (entre 2 et 77 fois) dans les phrases étudiées. En d'autres termes, 82% des formes apparaissent une fois et 18 % des formes plus d'une fois.
Il est important de noter que ces 110 formes différentes représentent plus de 60 % des formes des phrases. Et plus précisément, seulement 44 formes différentes permettent de représenter la moitié (ou 49,96 %) du nombre total de formes.

---

[21] Ce corpus, issu du journal Le Monde, a été utilisé par notre équipe de recherche dans le cadre du projet Européen d'évaluation d'étiqueteurs grammaticaux, le projet "Grace" (cf. [Bertier & Lallich, 97]).

Voici les 44 formes qui permettent d'exprimer la moitié des phrases du corpus (cf. Tableau 5).

Formes	Nombre d'occurrences	Nombre d'occurrences cumulées	Pourcentage par rapport au nombre total de formes	Pourcentage cumulé par rapport au nombre total de formes
.	77	77	5,92%	5,92%
de	71	148	5,46%	11,38%
,	63	211	4,84%	16,22%
la	41	252	3,15%	19,37%
les	31	283	2,38%	21,75%
à	28	311	2,15%	23,90%
l'	24	335	1,84%	25,75%
le	20	355	1,54%	27,29%
est	19	374	1,46%	28,75%
des	17	391	1,31%	30,05%
dans	16	407	1,23%	31,28%
d'	15	422	1,15%	32,44%
en	15	437	1,15%	33,59%
et	15	452	1,15%	34,74%
un	15	467	1,15%	35,90%
une	13	480	1,00%	36,89%
a	12	492	0,92%	37,82%
pour	11	503	0,85%	38,66%
du	10	513	0,77%	39,43%
s'	10	523	0,77%	40,20%
deux	9	532	0,69%	40,89%
ces	8	540	0,61%	41,51%
été	8	548	0,61%	42,12%
il	8	556	0,61%	42,74%
plus	8	564	0,61%	43,35%
qui	8	572	0,61%	43,97%
sont	8	580	0,61%	44,58%
sur	6	586	0,46%	45,04%
être	5	591	0,38%	45,43%
ne	5	596	0,38%	45,81%
ont	5	601	0,38%	46,20%
par	5	606	0,38%	46,58%
pas	5	611	0,38%	46,96%
ans	4	615	0,31%	47,27%
c'	4	619	0,31%	47,58%
entre	4	623	0,31%	47,89%
leur	4	627	0,31%	48,19%
nous	4	631	0,31%	48,50%
que	4	635	0,31%	48,81%
au	3	638	0,23%	49,04%
aussi	3	641	0,23%	49,27%
cette	3	644	0,23%	49,50%
depuis	3	647	0,23%	49,73%
elle	3	650	0,23%	49,96%

Tableau 5 : Formes fréquentes du corpus

• Etude d'ambiguïtés catégorielles du corpus

Nous nous sommes intéressé à ces 44 formes représentant environ 50 % des formes (ou 650 occurrences de formes) présentes dans le corpus. Comme le montre le tableau suivant, parmi ces formes, certaines correspondent à plusieurs catégories morphologiques, d'autres pas.

Formes	Nombre d'occurrences	Catégories non ambiguës	Nombre d'occurrences cumulées de formes mono-catégorielles	Catégories ambiguës	Nombre d'occurrences cumulées de formes pluri-catégorielles
.	77	T	77		
de	71			P,Q	71
,	63	T	140		
la	41			D,Y,F	112
les	31			D,Y	143
à	28	P	168		
l'	24			D,Y	167
le	20			D,Y	187
est	19			V,F	206
des	17	P(de) + D(les)	185		
dans	16			P,F	222
d'	15			(de) P,Q	237
en	15			P,Y	252
et	15	C	200		
un	15			D,F	267
une	13			D,F	280
a	12	V	212		
pour	11			P,F	291
du	10	P(de) + D(le)	222		
s'	10	(se) Y	232		
deux	9			F,D	300
ces	8	D	240		
été	8	F	248		
il	8	Y	256		
plus	8			W,F	308
qui	8	Q	264		
sont	8	V	272		
sur	6	P	278		
être	5			V,F	313
ne	5	G	283		
ont	5	V	288		
par	5	P	293		
pas	5			W,F	318
ans	4	F	297		
c'	4	Y	301		
entre	4			V,P	322
leur	4			D,Y	326
nous	4			Y,F	330
que	4	Q	305		
au	3	P(à) + D(le)	308		
aussi	3	W	311		
cette	3	D	314		
depuis	3	P	317		
elle	3			Y,F	333
TOTAUX	650		317		333

Tableau 6 : Catégories morphologiques des formes les plus fréquentes du corpus

Parmi ces 650 occurrences de formes, presque la moitié (pour être exact 317) correspondent à une seule catégorie. Nous nous attacherons, dans une perspective de désambiguïsation, aux 333 occurrences des formes les plus fréquentes de catégories ambiguës ; elles sont constituées de 20 formes différentes.

Formes	Catégories ambiguës	Nombre d'occurrences par type de catégories ambiguës
un	D,F	37
une	D,F	
deux	D,F	
les	D,Y	79
l'	D,Y	
le	D,Y	
leur	D,Y	
la	D,Y,F	41
dans	P,F	27
pour	P,F	
de (ou d')	P,Q	86
en	P,Y	15
est	V,F	24
être	V,F	
entre	V,P	4
plus	W,V	8
pas	W,F	5
nous	Y,F	7
elle	Y,F	
TOTAL		333

Tableau 7 : Catégories ambiguës parmi les formes les plus fréquentes

Les 20 formes ambiguës les plus fréquentes du corpus peuvent être regroupées en 11 classes de catégories ambiguës possibles.

Ce constat nous a conduit à développer, d'une part, des règles de désambiguïsation selon les catégories possibles (cf. § II-1-1) :

- Règles pour le choix entre D et F (concernant les formes "*un*", "*une*", "*deux*"),
- Règles pour le choix entre D et Y (pour les formes "*les*", "*l'*", "*le*", "*leur*"),
- Règles pour le choix entre P et F (regroupant les formes "*dans*", "*pour*"),
- Règles pour le choix entre V et F (concernant formes "*est*", "*être*"),
- Règles pour le choix entre Y et F (regroupant les formes "*nous*", "*elle*"),

D'autre part, nous avons développé aussi des règles concernant des formes spécifiques, lorsque seule une forme est concernée par les catégories ambiguës (cf. § II-1-2). C'est le cas des formes : "*la*", "*de*", "*en*", "*entre*", "*plus*", "*pas*".

• Synthèse

Nous avons étudié un corpus journalistique composé de 77 phrases comportant 1301 formes. Nous avons recensé 617 formes différentes et nous avons constaté que seulement 44 formes différentes permettent de représenter la moitié (soit 49,96%) du nombre total de formes du corpus. Nous avons étudié les formes les plus fréquentes de catégories ambiguës ; elles sont constituées de 20 formes différentes pouvant être regroupées en 11 classes de catégories ambiguës (cf. Tableau 7).

Ces types de catégories ambiguës ont conduit à étudier les suites de catégories possibles dans une perspective de désambiguïsation de catégories multiples. Notre objectif est de développer, d'une part, des règles de désambiguïsation sur les catégories ambiguës lorsqu'elles concernent plusieurs formes. D'autre part, de proposer des règles de désambiguïsation sur des formes particulières, lorsque les catégories ambiguës qu'elles représentent sont spécifiques à ces formes. Nous verrons qu'il est possible de traiter au niveau morphologique d'autres types d'ambiguïtés relevant par exemple de valeurs de variables différentes (cf. § II-1) .

Notons que le corpus comprend au départ 1301 formes, mais il en comprend 1354 après les différentes régularisations. Cela est dû aux règles de régularisation du type : "*du*" est transformé en "*de*" + "*le*", "*laquelle*" devient "*que*" + "*la*" + "*%*",… Ce type de règles ajoute donc des formes au nombre total de formes avant prétraitement ; mais, aucune ambiguïté n'est créée puisque les formes sont alors catégorisées correctement.
Avant l'analyse morphologique, le corpus comprend donc 1354 formes représentant 2043 entrées lexicales possibles. Parmi, ces formes, 525 relèvent de plusieurs entrées lexicales (cf. Tableau 8).

	Nombre de formes	**Formes relevant d'une seule entrée lexicale**	**Formes relevant de plusieurs entrées lexicales**
Au total	1354	829 (soit 61%)	525 (soit 39%)
Par phrases (moyenne)	17,58	10,76	6,81

Tableau 8 : Synthèse des formes du corpus avant désambiguïsation

Cette étude du corpus nous a permis de reconnaître les ambiguïtés catégorielles les plus fréquentes ; nous verrons comment ces ambiguïtés sont traitées dans TALISMAN II.

II) Agent MORPH

Nos travaux sur l'agent Morph concernent la désambiguïsation morphologique : les règles linguistiques (présentées dans ce chapitre) et la coopération avec les autres agents du système, notamment avec l'agent Synt (présentée dans le chapitre suivant).

II-1) Analyse morphologique dans TALISMAN II

L'analyse morphologique a pour objectif l'identification des formes. Elle pose les problèmes suivants : délimiter les frontières entre les formes et retrouver les entrées lexicales correspondantes ainsi que leurs catégories et les valeurs de leurs variables associées.
Les ambiguïtés relevant du niveau de l'analyse morphologique concernent essentiellement, dans TALISMAN II, les agents Regul (anciennement Pret, cf. Chapitre I) et Morph.
L'agent Regul a pour but de représenter une chaîne de caractères comme une liste de formes. Il effectue les traitements suivants : éclatements d'amalgames orthographiques et morphologiques, traitement de formes composées, traitement de l'apostrophe, des majuscules, repérage du trait d'union.

L'agent Morph a pour but d'analyser une forme dans son contexte, c'est-à-dire de trouver la ou les bonnes analyses morphologiques (i.e. attestées par la langue). Il effectue les traitements suivants : analyse morphologique flexionnelle, traitement du trait d'union, désambiguïsation à l'aide de règles linguistiques.

II-2) Ambiguïtés relevant du niveau morphologique

Nous distinguerons les différents types d'ambiguïtés du niveau morphologique suivant leur origine linguistique : typographique, lexicale, morphologique flexionnelle, morphologique dérivationnelle.

II-2-1) Ambiguïtés d'origine typographique

Les ambiguïtés d'origine typographique sont bien souvent des ambiguïtés virtuelles ; elles apparaissent dès le début de l'analyse mais peuvent généralement être résolues au niveau morphologique. Nous allons étudier certaines ambiguïtés liées à la capitalisation et à la ponctuation. Ces phénomènes ont été étudiés dans le cadre de la correction d'erreurs par [Froissart, 92].

• Cas des majuscules

Le problème de la capitalisation apparaît en présence de titres mis en majuscule, de mise en relief par capitalisation et d'initiale de nom propre ou de phrase.

Une majuscule en tête de phrase peut représenter une initiale de nom propre ou non. Elle doit être mise en minuscule. Mais les voyelles en majuscules posent de problème de leur accentuation. En particulier, la réduction n'est pas univoque sur les caractères "*A*" et "*E*".

Exemples :

"Etes…" = "êtes" (seule accentuation correcte)

"… l'Etat…" = nom propre "état" (seule accentuation correcte)

"A…" = préposition "à" ou auxiliaire "a"

"Pe'kin…" = nom propre "pékin" (n'est pas un nom commun en début de phrase)

• Cas des ponctuations

Les ponctuations[22] ne sont pas toujours faciles à traiter [Froissart, 92]. Le point n'est pas toujours une ponctuation forte, la virgule une ponctuation faible et l'apostrophe un séparateur de formes.

Exemples : "M. ", "3,7", "aujourd'hui"

---

[22] Nous distinguons différents types de ponctuation (cf. [Berrendonner, 83]) : les ponctuations fortes marquant une frontière de phrase (comme le point, le point d'interrogation, le point d'exclamation), les ponctuations faibles séparant deux membres de la phrase (comme la virgule, le point virgule, le deux-points) et les ponctuations balancées (parenthèses, guillemets,…).

II-2-2) Ambiguïtés d'origine lexicale

Ces ambiguïtés d'origine lexicale[23] sont inhérentes à la langue dans la mesure où une forme peut relever de catégories différentes ; elles deviennent virtuelles lorsqu'elles persistent malgré la prise en compte de l'ensemble du contexte syntaxique sauf pour certains cas d'école (ex. :"il détient le pouvoir et *le contrôle*.", où "*le*" reste déterminant ou préverbal et "*contrôle*" reste nom ou verbe). Ce type d'ambiguïté est lié à la structuration du dictionnaire adopté : notre dictionnaire comporte, pour une forme, au moins autant d'entrées lexicales (homonymes[24] ou différentes[25]) que de catégories distinctes associées.
Ce type d'ambiguïté est résolu soit dès le niveau morphologique (cf. § II-4) soit au niveau syntaxique (cf. § III-4-3) ; sauf évidemment pour les cas d'ambiguïté linguistique (comme le cas d'école exposé ci-dessus).
Nous allons étudier des ambiguïtés portant sur la catégorie morphologique d'une forme.

• Ambiguïtés sur la catégorie morphologique d'une forme
De nombreuses formes peuvent relever de plusieurs catégories morphologiques. Les catégories utilisées sont issues des travaux d'Alain Berrendonner [Berrendonner, 83] : verbes (V), nominaux (F), pronoms préverbaux (Y), déterminants (D), conjonctions de subordination (Q), conjonctions de coordination (C), adverbes (W), prépositions (P), pro-phrases "oui" et "si" (H), "ne" et "non" (G), ponctuations (T).
Exemples :

# la * la	# doute * doute
D la	F doute
Y la	V douter
F la	

II-2-3) Ambiguïtés d'origine morphologique flexionnelle

Ces ambiguïtés d'origine morphologique flexionnelle sont en général des ambiguïtés virtuelles. Ce type d'ambiguïté apparaît dès le niveau morphologique ; des désambiguïsations peuvent être réalisées par l'agent Morph (cf. § II-4-3 et § II-4-4).

• Ambiguïtés sur les variables de la catégorie d'une forme
Ce problème comprend le problème linguistique de l'homonymie : par exemple, le nom "*voile*" possède deux "comportements morphologiques", l'un au féminin et l'autre au masculin.
Une forme d'une catégorie donnée peut correspondre à plusieurs possibilités de valeurs des variables. Les variables des catégories précédemment citées, sont aussi inspirées de [Berrendonner, 83] (cf. Tableau 9).

---

[23] Notons que nous ne traitons pas l'erreur lexicale.
[24] La forme "le" renvoie à deux entrées lexicales homonymes : "le" Préverbal catégorie Y et "le" Déterminant catégorie D.
[25] La forme "avions" renvoie à deux entrées lexicales différentes : "avion" Nom catégorie F et "avoir" Verbe catégorie V.

Nom	Signification	Catégories concernées	Valeurs et signification
NA	Type nominal	F	NOM (nom), ADJ (adjectif), NAN (nom ou adjectif)
NN	Sous-type nominal	F	PRP (propre), COM (commun), PRO (pronominal)
PA	Participe	F	PPA (participe passé), PPR (participe présent)
VB	Type verbal	V	INF (infinitif), FIN (forme finie)
VX	Auxiliarité	V	AUX (auxiliaire), ORD (ordinaire)
MD	Mode	V	IPF (imparfait), ISU (indicatif, subjonctif),…
IN	Valeur de pronoms	Y	IN1 (sujet), I2A (accusatif), I2B (datif),…
GR	Genre	Y, F ou D	MAS(masculin), FEM (féminin), GRN (non marqué)
NB	Nombre	Y, F ou D	SNG (singulier), PLU (pluriel), NBN (non marqué)
PE	Personne	Y ou V	PE1 (1^{ère} personne sing.), PE2, PE3, PE4, PE5, PE6
NG	Polarité négative	W ou F	NEG (négatif), NNG (non négatif)
VP	Valeur de ponctuation	T	VP1 (forte), VP2 (faible), VP3 (autres)

Tableau 9 : Exemples de variables de catégories morphologiques

Exemples :

# dynamique * dynamique {2 entrées lexicales - 2 solutions morphologiques}
dynamique F GRN SNG *ADJ*… (adjectif)
dynamique F FEM SNG *NOM* COM… (nom)

# doute * doute {2 entrées lexicales en tant que verbe -
5 solutions morphologiques en tant que verbe}
douter *PE2 IPF* PST ORD… (2ème personne de l'impératif)
douter *P13 ISU* PST ORD… (1 ère ou 3 ème personne de l'indicatif ou du subjonctif ;
ce qui représente 4 solutions différentes)

# voile * voile {4 entrées lexicales en tout - 7 solutions morphologiques en tout}
voile F *MAS* SNG NOM…(nom masculin)
voile F *FEM* SNG NOM…(nom féminin)
voiler V *PE2 IPF* PST ORD… (2ème personne de l'impératif)
voiler V *P13 ISU* PST ORD… (1 ère ou 3 ème personne de l'indicatif ou du subjonctif)

II-2-4) Ambiguïtés d'origine morphologique dérivationnelle

Ces ambiguïtés du niveau morphologique sont bien souvent virtuelles et leur résolution diffère selon les cas. Nous allons étudier des ambiguïtés sur le découpage de formes multiples[26] : des formes composées[27] et des formes comportant un trait d'union.

---

[26] Nous ne traitons pas ici, le découpage par un analyseur morphologique dérivationnel de formes simples telles : "*parachutiste*" découpé en "*para*" + "*chut*" + "*iste*" (cf. [Clavier, 96a]).

- **Ambiguïtés sur le découpage des formes**

Le repérage des formes est en général facilité par la présence d'un espace avant et après chaque forme. Cependant certains cas restent ambigus : le trait d'union pose le problème du découpage ou non d'une forme en deux. Quant aux formes composées, leur regroupement facilite l'analyse syntaxique, mais elles ne peuvent pas toutes être regroupées systématiquement[28].

Exemples :

- formes avec trait d'union : "Peut-on", "Port-Mariane", "Entre-temps", "vis-à-vis", "rendez-vous"
- mots composés : "20 000", "grâce à", "en effet", "coup de filet"

Nous avons recensé un certain nombre d'ambiguïtés relevant du niveau d'analyse morphologique. Dans la section suivante, nous présentons l'agent Morph ; ceci nous permettra (dans la section II-4) de développer des règles de levée d'ambiguïtés permettant de résoudre certaines des ambiguïtés présentées ci-dessus.

II-3) Présentation de l'agent Morph

L'objectif de l'agent Morph(ologie) est de trouver pour chaque forme les entrées lexicales, et les bonnes catégories morphologiques ainsi que les valeurs de leurs variables. Pour cela, il dispose des **connaissances** suivantes :

- le dictionnaire du français courant construit à partir du DELAS,
- les dictionnaires de spécialité suivant l'application,
- le dictionnaire de certains mots composés,
- la liste des modèles flexionnels des noms, des adjectifs et des verbes,
- la liste des flexions ainsi que leur compatibilité mutuelle,
- les régularisations de formes et de bases.

Ces connaissances permettent à l'agent Morph de réaliser les traitements suivants appelés **compétences** :

- l'analyseur morphologique flexionnel,
- le module de désambiguïsation à l'aide de règles linguistiques,
- le traitement de mots inconnus : nous proposons ici un traitement spécifique des mots inconnus (comme des néologismes, mais nous n'avons pas intégré la correction d'erreurs dans l'architecture). Nous nous sommes fondé sur les travaux de [Lasaunière, 94], qui a effectué une étude sur la reconnaissance du vocabulaire de spécialité dans une perspective d'indexation automatique. Ces recherches ont montré que bien souvent ces

---

[27] Par formes composées nous entendons :
- les noms composés : "pomme de terre",
- les locutions adverbiales : "de temps en temps",
- les locutions conjonctives : "du fait que",
- les locutions verbales : "avoir lieu",
- les articles composés : "la plupart des".

[28] Nous ne voulons pas étiqueter l'ensemble des formes composées dans un dictionnaire comme par exemple dans l'approche de [Gross, 90] ; nous préférons utiliser différentes stratégies de détection de formes composées.

mots étaient des noms ou adjectifs[29] et parfois des adverbes ; reconnaissables à leur suffixe. D'autres mots inconnus peuvent être reconnus en ôtant leur préfixe. L'utilisation d'un analyseur dérivationnel est fort utile. Pour l'heure, nous avons implémenté la stratégie suivante : l'agent Morph propose différentes catégories possibles à l'agent Synt, la première catégorie proposée étant le nominal F.

La mise en œuvre de ses compétences est réalisée selon un ordre préétabli ; nous avons défini la **stratégie générale** de l'agent Morph de la manière suivante :

1. choix du dictionnaire,
2. analyse morphologique flexionnelle,
3. traitement des mots inconnus (coopération avec l'agent Synt),
4. désambiguïsation avec des règles linguistiques contextuelles,
5. coopération avec Synt pour certains cas d'ambiguïtés, puis retour en 4.

II-4) La désambiguïsation linguistique

Une des compétences de l'agent Morph est la désambiguïsation avec des règles linguistiques contextuelles.
Nous sommes parti du principe qu'une désambiguïsation pouvait être mise en œuvre en utilisant des règles basées sur des critères linguistiques (l'utilisation de ces règles avait été proposée dans [Stéfanini, 93]). Ces règles doivent permettre l'élimination rapide de solutions parasites. Nous avons étudié les ambiguïtés issues de notre corpus et analysé différents types d'ambiguïtés morphologiques selon leur origine linguistique.
Nous avons donc défini différents types de règles linguistiques contextuelles de désambiguïsation selon les critères suivants :

- des règles selon le contexte d'apparition de catégories différentes concurrentes,
- des règles spécifiques à certaines formes fréquentes du corpus,
- des règles permettant de choisir entre deux catégories identiques concurrentes en fonction des variables associées à chacune de ces catégories,
- des règles utilisant d'autres types d'information sur la nature de la phrase (telle que le mode impératif, la position des formes dans la phrase).

Dans la section suivante, nous présentons la plupart des règles[30] développées ainsi que des exemples d'utilisation issus de notre corpus.

II-4-1) Règles de choix de catégories différentes concurrentes

Ce type de règles permet de lever des ambiguïtés morphologiques d'origine lexicale (cf. § II-2-2).

---

[29] Ces résultats concordent avec notre corpus dans lequel parmi les trois mots non reconnus par le dictionnaire, deux sont des nominaux ("*autoproduites*" et "*disc-jockeys*"), et le troisième est une erreur d'accentuation ("*maitrisables*").

[30] La notation utilisée pour les règles est la suivante :
$\alpha 1$, $\alpha 2$, $\alpha 3$ représente l'ensemble des variables associées à une catégorie,
Z désigne une catégorie quelconque, et {Z} désigne une liste de catégories quelconques,
[Z1 / Z2] signifie que la forme peut relever des deux catégories Z1 et Z2.

Nous présentons certaines des règles que nous avons développées selon les catégories concurrentes :

- **Déterminant vs préverbal**

L'ambiguïté entre un déterminant et un préverbal est courante. Elle concerne les formes "*les*", "*l'*", "*le*", et "*leur*". Nous avons défini deux règles permettant de déterminer la catégorie correcte en fonction du contexte d'apparition de ce type d'ambiguïté.

```
REGLE : [D(α1) / Y(α2)] + [F(α3)] → [D(α1)] + [F(α3)]
    où ppr[31] ∉ α3 /* pour "le faisant", "le montant",… */
           genre.α1 = genre.α3 et nombre.α1 = nombre.α3
    Exemples : "l'ordre" (ph 1), "l'instant" (ph 2)

REGLE : [D / Y] + [V] → [Y] + [V]
    Exemples : "leur permettait" (ph 48), "l'entendent" (ph 72)
```

- **Verbe vs nominal**

L'ambiguïté entre verbe et nominal est très courante, puisque de nombreux noms sont dérivés de verbes. Exemples : "*doute*" vs "*douter*" (ph1), "*centre*" vs "*centrer*" (ph3), "*veille*" vs "*veiller*" (ph4),…

```
REGLE : [D(α1)] + [V(α2) / F(α3)] → [D(α1)] + [F(α3)]
    où     genre.α1 = genre.α3 et nombre.α1 = nombre.α3
    Exemples : "cette usine" (ph 18), "leurs luttes" (ph 59)

REGLE : [Y] + [V / F] → [Y] + [V]
    Exemples : "il est" (ph 41), "c'est" (ph 49)
```

- **Préposition vs nominal**

Nous avons trouvé aussi de nombreuses ambiguïtés dues à des formes correspondant à la catégorie P (préposition) ou F (nominal). C'est le cas des formes "*pour*", "*dans*",…

```
REGLE : [D(α1)] + [P(α2) / F(α3)] → [D(α1)] + [F(α3)]
    où     genre.α1 = genre.α3 et nombre.α1 = nombre.α3
    Exemples :"ces dans (de judo)"

REGLE : [P / F] + [D] → [P] + [D]
    Exemples : "dans la" (ph 56)
```

- **Déterminant vs nominal**

Cette ambiguïté catégorielle concerne d'une part, les formes "un" et "une" pouvant être déterminant D, nom F(nom) ou adjectif F(adj) ; et, d'autre part, les formes "deux", "trois"… relevant des catégories déterminant D et nominal F(nan) avec la variable numérique NUM.

```
REGLE : [D(α1)] + [D(α2) / F(α3) / F(α4)] → [D(α1)] + [F(α3)]
    où     genre.α1 = genre.α3 et nombre.α1 = nombre.α3
           nom ∈ α3, adj ∈ α4
    Exemple :"la une (du journal)"

REGLE : {Z(α1)} + [D(α2)/F(α3)] + [F(α4)]→ {Z(α1)} + [D(α2)] + [F(α4)]
    où     D ∉ Z et num ∈ α2 et num ∈ α3
           genre.α2 = genre.α4 et nombre.α2 = nombre.α4
    Exemple :"de deux parties" (ph 36)
```

---

[31] "ppr" signifie participe présent.

II-4-2) Règles sur certaines formes

Nous avons défini des règles portant sur des formes spécifiques fréquentes dans le corpus (cf. section I).

- **La forme "la" (Y/D/F)**

```
REGLE "la" : {Z}+[Y(α1)/D(α2)/F(α3)]+[F(α4)] → {Z} + [D(α2)] + [F(α4)]
    où    D ∉ Z et F ∉ Z (contre exemple : "le beau la")
          ppr ∉ α4 (contre exemple : "la jouant")
          genre.α2 = genre.α4 et nombre.α2 = nombre.α4
    Exemple : "la filiale" (ph 17)
```

- **La forme "de"[32] (P/Q)**

```
REGLE "de" : [P(α1) / Q(α2)] + [V(α3)] → [Q(α1)] + [V(α3)]
    où    inf ∈ α3
    Exemples : "de mettre" (ph 7), "d'embellir" (ph 20)

REGLE "de" : [P(α1) / Q(α2)] + [D(α3)] → [P(α1)] + [D(α3)]
    Exemples : "de ses" (ph 2), "de ces" (ph 33)
```

- **La forme "en" (P/Y)**

```
REGLE "en" : [P(α1) / Y(α2)] + [F(α3)] → [P(α1)] + [F(α3)]
    où    ppr ∉ α3 /* contre exemple : "en lançant" */
    Exemple : "en amont" (ph 22)

REGLE "en" : [P(α1) / Y(α2)] + [V(α3)] → [Y(α1)] + [V(α3)]
    Exemple : "en a (pris note)" (ph 57)
```

- **La forme "pas" (W/F)**

```
REGLE "pas" : [V(α1)] + [W(α2) / (F(α3)] → [V(α1)] + [W(α2)]
    Exemple : "suffisent pas" (ph 35)
```

II-4-3) Règles sur le choix de variables avec catégories identiques

Ce type de règles concerne les ambiguïtés d'origine morphologique flexionnelle (cf. § II-2-3).

Nous avons défini des règles portant sur des variables de catégories identiques. Elles sont utilisées quand il existe pour une même forme deux entrées lexicales de type nominal : l'une étant considérée comme un adjectif, l'autre comme un nom.

```
REGLE "Nom/Adj" : [F(α1)] + [F(α2) / F(α3)] → [F(α1)] + [F(α3)]
    où     nom ∈ α1, nom ∈ α2, adj ∈ α3
    Exemples : "dispositif militaire" (ph 5),
               "aristocratie financière (ph 41)

REGLE "Nom/Adj" : [F(α1) / F(α2)] + F[(α3)] → [F(α1)] + [F(α3)]
    où adj ou nan ∈ α1, nom ∈ α2, nom ∈ α3
    Exemples : "bonne dizaine" (ph 60)
```

---

[32] Ces règles concernent aussi la forme "*d'*" qui est transformée en "*de*" par l'agent Regul.

II-4-4) Règles sur certains critères linguistiques particuliers

Nous avons défini une règle qui permet d'éliminer les verbes au mode impératif si la phrase n'est pas à ce mode (cette règle résout un cas particulier d'ambiguïté d'origine morphologique flexionnelle présenté au paragraphe II-2-2).
Nous utilisons aussi une règle qui permet de préférer un syntagme nominal en tête de phrase à un syntagme verbal dans le cas d'une phrase assertive.

```
REGLE  : syntagme nominal en tête de phrase assertive (D/Y)
[D(α1) / Y(α2)]pos1 + [V(α3) / F(α4)]pos2 → [D(α1)] + [F(α4)]
    où    i2a[33] ∈ α2 et inf ∉ α3 /* pour "Il fait" et "Le prendre" */
          genre.α1 = genre.α4 et nombre.α1 = nombre.α4
    Exemples : "Les rues" (ph 5), "Les poissons" (ph 15)

REGLE : syntagme nominal en tête de phrase assertive (D/Y/F)
[D(α1)/Y(α2)/F(α3)]pos1 + [V(α4)/F(α5)]pos2 → [D(α1)] + [F(α5)]
    où    i2a ∈ α2 et inf ∉ α4
          genre.α1 = genre.α5 et nombre.α1 = nombre.α5
    Exemples : "La veille" (ph 4), "La demande" (ph 10)
```

II-5) Bilan

Ce bilan nous permet d'analyser les résultats fournis par l'agent Morph pour le corpus choisi (cf. Annexe - Phrases tests pour la désambiguïsation morphologique) et les ambiguïtés que nous avons étudiées. Nous proposons ensuite quelques améliorations possibles.

II-5-1) Apports

- **Résultats par rapport au corpus**

Avant l'analyse morphologique, nous avons vu que le corpus de 77 phrases comprenait 1354 formes représentant 2043 entrées lexicales possibles. Il y avait 525 formes avec entrées lexicales multiples.
Les résultats du premier passage du module de levée d'ambiguïtés montrent que 423 règles ont été appliquées ; les 1354 formes ne correspondent alors plus qu'à 1533 entrées lexicales. Les règles ont permis d'éliminer 510 solutions parasites (2043-1533=510) ; soit 25% des entrées lexicales sont éliminées. En d'autres termes, le corpus qui comprenait 525 formes relevant de plusieurs entrées lexicales n'en contient plus que 137.
Ceci étant réalisé sans aucune erreur de désambiguïsation.

---

[33] "i2a" correspond à un préverbal accusatif.

Sur 1354 formes	**Formes relevant d'une seule entrée lexicale**	**Formes relevant de plusieurs entrées lexicales**
Avant désambiguïsation	829 (soit 61%)	525 (soit 39%)
Après une première désambiguïsation	1217 (soit 90 %)	137 (soit 10%)

Tableau 10 : Synthèse des formes du corpus après une première levée d'ambiguïtés

Les règles de désambiguïsation que nous avons développées représentent une première étape. Il nous semble que ces règles pourraient être affinées avec un corpus plus important. Par ailleurs, les 10 % d'ambiguïtés persistantes pourraient faire l'objet d'une coopération avec un module statistique de levée d'ambiguïtés ; il nous semblerait intéressant d'étudier les résultats.

- **Résultats relatifs aux différents types d'ambiguïtés du niveau morphologique**

Les règles définies permettent de résoudre certaines ambiguïtés d'origine lexicale et morphologique flexionnelle. Nous montrons au paragraphe III que les stratégies syntaxiques de désambiguïsation permettent de résoudre d'autres ambiguïtés de ce type.
Des améliorations des agents du niveau morphologique peuvent toutefois être envisagées.

II-5-2) Perspectives

Nous proposons des perspectives pour la résolution interne des agents du niveau morphologique : les agents Regul et Morph.

II-5-2-1) Ambiguïtés d'origine typographique

Nous présentons des ambiguïtés d'origine typographique pouvant être résolues par l'agent Regul.

- **Traitement de la majuscule en début de forme (non en début de phrase)**

L'agent Regul reconnaît cette forme comme étant un nom propre ; la majuscule sera conservée et la forme sera étiquetée en tant que nom propre. Le problème de l'accentuation des voyelles en majuscules n'est pas traité[34].

- **Traitement de certaines ponctuations**

L'agent Regul reconnaît les différents types de ponctuation.
La virgule est considérée comme une ponctuation faible dans les cas où il existe un espace après. Dans le cas inverse (ex. :"*3,7*"), il ne s'agit pas d'une ponctuation séparatrice de formes. De même, un point entouré de chiffres n'engendrera aucun traitement (la forme "*3.7*" restera une seule et même forme).
En ce qui concerne l'apostrophe, certains cas sont traités : par exemple "*d'*" devient "*de*", par contre "*l'*" n'est pas modifié car il peut correspondre aux formes "*le*" ou "*la*". Dans

---

[34] D'ailleurs, nous ne traitons pas non plus les voyelles minuscules inaccentuées puisque cela relève du domaine de la correction d'erreurs [Froissart, 92].

certains cas l'apostrophe n'est pas considérée comme un séparateur de formes (comme pour "*aujourd'hui*", "*presqu'île*", "*grand'mère*",…) ; ces formes ne seront donc pas modifiées[35]. Le point d'interrogation ou le point d'exclamation sont supprimés au cours du traitement, et la phrase est annotée respectivement par "interrogative" ou "impérative" (ou "exclamative").

II-5-2-2) Ambiguïtés d'origine morphologique dérivationnelle

Nous développons un type d'ambiguïté d'origine morphologique dérivationnelle pouvant être résolu par les seules compétences de l'agent Morph.

• Traitement de formes composées

L'agent Morph possède une liste (non exhaustive) des formes composées ("*coup de filet*", "*en effet*", "*au fur et à mesure*", "*petit à petit*", "*grâce à*",…). Ces formes composées peuvent être ambiguës ("*grâce à lui*" vs "*la grâce à Lourdes*"). Donc, deux solutions seront conservées : ces suites de formes seront analysées forme par forme et elles seront interprétées comme une seule forme (lorsqu'elles sont présentes dans le lexique).

Exemple :

Une forme "grâce à" : Préposition VSDeux formes : "grâce" Nom et "à" Préposition

Nous avons présenté l'agent Morph et les ambiguïtés qu'il est capable de résoudre. Nous verrons dans le chapitre suivant que cet agent résout d'autres ambiguïtés par coopération avec les autres agents du système.

III) Agent SYNT

Nous avons développé un agent Synt(axe) capable de réaliser l'analyse syntaxique en constituants et l'analyse fonctionnelle d'une proposition. Ces travaux nous ont permis de mesurer l'ampleur de la problématique d'un analyseur syntaxique. Nous allons montrer comment est réalisé cet analyseur et nous étudierons les ambiguïtés qu'il traite.

III-1) Analyse syntaxique dans TALISMAN II

L'analyse syntaxique renvoie aux propriétés formelles des constructions linguistiques ; elle doit permettre la structuration de la phrase en constituants fonctionnels hiérarchisés. L'analyse syntaxique nécessite, dans notre modèle linguistique, la segmentation de la phrase en propositions et en syntagmes, ainsi que l'association d'une fonction syntaxique à ces syntagmes et propositions.

Dans TALISMAN II, les ambiguïtés du niveau d'analyse syntaxique concernent essentiellement les agents Synt et Segm.

---

[35] Remarque : dans certains cas, l'un ou l'autre des segments n'est jamais autonome (c'est le cas de "hui" dans "aujourd'hui") ; dans d'autres cas, les deux segments peuvent être autonomes (exemple : "grand'mère"). Ce problème rejoint celui du lexique de formes composées (vu au premier chapitre).

L'agent Synt a pour but de construire la ou les structures syntaxiques (en constituants et structures fonctionnelles) correctes (i.e. attestées par la langue) d'une suite de formes. Il effectue les traitements suivants : regroupement de morphèmes discontinus, détection des indicateurs de structure grammaticaux[36] et lexicaux[37], vérification des accords, analyse syntaxique en constituants de la proposition et détermination du rôle fonctionnel des syntagmes régis par le verbe.

L'agent Segm a pour but de segmenter une phrase en propositions. Il effectue les traitements suivants : prédiction sur le nombre de propositions, détection des propositions principales, subordonnées, et construction arborescente de la structure de la phrase en propositions.

La grammaire en constituants hors-contexte utilisée engendre des ambiguïtés, comme toute grammaire du même type. En effet, cette grammaire est constituée d'un ensemble de règles tel que l'on puisse engendrer tous les énoncés potentiels acceptables de la langue. La grammaire est donc surproductrice de solutions (dont des solutions parasites), mais elle rend compte des différentes possibilités ; l'utilisation d'un analyseur fonctionnel doit permettre d'éliminer de nombreuses solutions parasites. L'analyseur utilise le dictionnaire des schémas syntaxiques des verbes afin de déterminer le rôle fonctionnel des compléments. Suivant [Blank, 87] et [Blank & Palermiti, 87], nous distinguons différents types de compléments :

C0 : Complément se présentant sous forme d'un SN (Syntagme Nominal) non précédé par une préposition et anaphorisable en "*elle*" ou "*il*" ; appelé "sujet".

C1 : Complément se présentant sous forme d'un SN (non précédé par une préposition) et anaphorisable par les pronoms "*le*", "*la*", "*les*" ; appelé complément d'objet direct ou accusatif.

C2 : Complément introduit par la préposition "*à*" et suivi d'un SN. Ils sont anaphorisables par les pronoms "*lui*", "*leur*" et sont appelés complément d'objet indirect ou datif.

C3 : Complément introduit par la préposition "*à*" ou d'autres prépositions locatives "*vers*", "*sur*", "*en*", "*chez*",… et suivi d'un SN. Ils sont anaphorisables par le pronom "*y*" et sont appelés complément d'objet indirect ou locatif.

C4 : Complément introduit par la préposition "*de*" et suivi d'un SN. Ils sont anaphorisables par le pronom "*en*" et sont appelés complément d'objet indirect ou ablatif.

C5 : Regroupe les compléments introduits par une préposition et non anaphorisables et sont appelés circonstants.

III-2) Ambiguïtés relevant du niveau syntaxique

Nous distinguons différents types d'ambiguïtés du niveau syntaxique : celles liées aux grammaires du syntagme, de la proposition et de la phrase.

---

[36] Les indicateurs de structure grammaticaux (ISG) indiquent avec certitude le début d'un syntagme :
- les déterminants (D) qui permettent de repérer les débuts de syntagmes nominaux (SN),
- les prépositions (P) qui introduisent les syntagmes prépositionnels (SP),
- les pronoms préverbaux (Y) et les verbes (V) qui sont initiaux de syntagme verbal (SV).

[37] Les indicateurs de structure lexicaux (ISL) correspondent aux verbes, les noms et adjectifs dérivés de verbe, quelques adverbes, noms et adjectifs non déverbaux ou régissant des compléments. Un lexique donne pour chacun d'eux un ou plusieurs schémas syntaxiques.

III-2-1) Ambiguïtés liées à la grammaire du syntagme

• Ambiguïtés sur le rattachement de syntagmes

Le rattachement d'un syntagme n'est pas toujours facile à réaliser puisqu'il n'est pas toujours lié au syntagme précédent (i.e. au nœud le plus proche).

Exemples de rattachement d'un adjectif :

1. "… sa gestion devrait être naturellement confiée à la ville *propriétaire*."
2. "En un siècle, près de la moitié des 5 millions de mètres cubes de galets … ont été extraits pour alimenter les chantiers de construction *locaux* ou les fabricants de céramique."
3. "Au cours d'une conférence de presse *commune*,…"

Dans le premier exemple, le rattachement de l'adjectif "*propriétaire*" ne pose aucun problème. Par contre, dans le second exemple, le rattachement adjectival de "*locaux*" nécessite une vérification d'accord afin d'empêcher son rattachement à "*construction*"[38]. Quant au dernier exemple, le rattachement de l'adjectif "*commune*" (à "*presse*" ou à "*conférence*") reste ambigu au niveau syntaxique.

III-2-2) Ambiguïtés liées à la grammaire de la proposition

• Ambiguïtés sur la nature des propositions

La nature des propositions subordonnées peut être ambiguë. Ainsi, un même introducteur de proposition (comme "*qui*") peut introduire une proposition relative (comme dans le premier exemple) ou une proposition interrogative indirecte (comme dans le second exemple).

Exemples :

1. " Une fois de plus en ce siècle, cette instabilité [***qui*** prépare la guerre] prévaudra."
2. "Il a indiqué qu'il se rendrait bientôt à Bucarest pour y avoir des contacts directs avec les dirigeants et "voir à [***qui*** on a affaire]" ."

III-2-3) Ambiguïtés liées à la grammaire de la phrase

• Ambiguïtés sur la segmentation de la phrase en propositions

Le découpage d'une phrase en propositions est un problème complexe nécessitant parfois des connaissances très diverses sur la coordination, la négation,…

Exemples :

---

[38] Notons que si l'adjectif est placé après, c'est sans doute parce que "*chantiers de construction*" est considéré par l'auteur comme une forme composée. Le problème qui se pose alors est le suivant : est-il possible de recenser toutes les formes composées, ou doit-on les traiter comme des formes distinctes.

1. "La chorégraphe n'entend pas, Dieu merci, faire le tour d'une question qui, de toute éternité, a divisé *les théologiens* ***et*** *les moralistes*."
2. "On entendra *du jazz* ***et*** *des auteurs* liront des textes."

Dans chacun des deux exemples, le coordonnant "*et*" est entouré de deux syntagmes identiques ; mais seulement dans le second exemple, ce coordonnant permet de délimiter deux propositions. Le problème devient plus complexe lorsque la phrase comporte des verbes ambigus (ex. : "*il détient le pouvoir et le contrôle*" où, en l'occurrence, "*contrôle*" peut être un nom ou un verbe).

III-3) Présentation de l'agent Synt

L'agent Synt(axe) doit produire les bonnes analyses syntaxiques d'une phrase. Il dispose des **connaissances** suivantes :

- le dictionnaire des indicateurs de structures lexicaux notés ISL, (correspondant aux différents schémas syntaxiques des verbes),
- la liste des indicateurs de structures grammaticaux, notés ISG, (permettant de reconnaître la présence d'un début de syntagme nominal, verbal ou prépositionnel),
- la grammaire générale de base hors-contexte (grammaire de reconnaissance et de construction de l'analyse en constituants d'une proposition),
- les grammaires satellites (ou locales[39]) qui permettent la reconnaissance de syntagmes notamment des syntagmes nominaux complexes (avec ou sans nom propre).

Les connaissances de l'agent Synt lui permettent d'activer ses **compétences** :

- le regroupement des morphèmes discontinus (notamment des verbes conjugués avec un auxiliaire),
- l'analyse syntaxique en constituants de la proposition standard,
- l'analyse syntagmatique pour la reconnaissance des syntagmes nominaux, verbaux et prépositionnels : la stratégie consiste à déterminer la nature du syntagme (grâce aux ISG), puis à activer les grammaires satellites correspondantes (grammaire du syntagme verbal, prépositionnel ou nominal — standard ou complexe),
- la vérification des accords (le but étant de désambiguïser, et non pas de corriger des erreurs),
- la réalisation de l'analyse fonctionnelle à partir de la détermination du rôle fonctionnel des syntagmes régis par le verbe,
- la reconnaissance des circonstanciels (les compléments circonstants étant définis comme tout syntagme prépositionnel non régi par le schéma syntaxique des verbes),
- l'association entre les constituants syntagmatiques (reconnaissance des différents types de syntagmes) et le schéma syntaxique du verbe (rôle fonctionnel des différents types de syntagmes).

Nous proposons une première **stratégie générale** intégrant les grammaires satellites, afin que l'agent Synt puisse atteindre son but :

---

[39] Les grammaires locales (proposées par A. Berrendonner) sont définies par un ensemble de règles de grammaire activables à partir de certains indices présents dans le texte.

1. La construction des différentes suites possibles de catégorie morphologique (appelées "trajectoires") en cas d'analyse morphologique multiple pour une ou plusieurs formes de la proposition. Ainsi, si les ISG ne sont pas désambiguïsés (comme la forme "*le*" pouvant être un introducteur-Y de syntagme verbal ou un introducteur-D de syntagme nominal), l'analyse devra fournir différents types d'arbres syntagmatiques. De même, si le verbe de la proposition est ambigu, il faut tester l'ensemble des verbes possibles et leurs schémas syntaxiques associés. Les stratégies utilisées pour cette préparation à l'analyse en constituants de la proposition sont détaillées au paragraphe III-4-3.
2. La construction de l'analyse en constituants de la proposition à l'aide de la grammaire générale. Si cette dernière échoue, l'appel aux grammaires satellites (comme la grammaire du syntagme nominal complexe) est réalisé. En général, plusieurs arbres syntagmatiques de la proposition sont obtenus.
3. L'association entre syntagmes et compléments du verbe : il s'agit d'associer à un verbe donné, pouvant comporter différents schémas syntaxiques possibles, les structures en constituants déjà repérées dans la proposition.
4. Le rattachement des propositions en une phrase (en coopération avec ses accointances : Coord, Segm,…).

III-4) Les différentes compétences de Synt

Nous avons étudié une soixantaine de propositions issues des phrases du corpus journalistique présentées au début de ce chapitre. Certaines de ces propositions (une vingtaine environ) sont totalement désambiguïsées localement par Morph grâce à ses règles linguistiques contextuelles de désambiguïsation. Pour d'autres propositions, certaines ambiguïtés subsistent même après l'action de l'agent Morph (certaines propositions seront totalement désambiguïsées par coopération entre agents). Voici un petit échantillon de ces propositions :

```
Proposition 1 : La seule incertitude concerne l'ordre hiérarchique.
Proposition 2 : Des échanges ont lieu entre les deux Allemagnes.
Proposition 3 : L'opposition est visée.
Proposition 4 : Une culture s'épanouissait discrètement en vase
                 clos.
```

Dans la section suivante, nous exposons la grammaire syntaxique en constituants de la proposition que nous testerons sur les propositions du corpus. Nous présentons ensuite l'analyse fonctionnelle de la proposition que nous avons élaborée. Enfin, nous proposons des stratégies linguistiques pour éliminer certaines solutions parasites (dans le cas où Morph n'a pas pu lever toutes les ambiguïtés).

III-4-1) Analyse en constituants

Nous présentons succinctement la grammaire syntaxique de la proposition que nous avons développée, en l'illustrant par des exemples.

III-4-1-1) Grammaire syntaxique en constituants de la proposition

Nous avons utilisé une grammaire de la proposition standard proposée par A. Berrendonner (cf. Annexe - La grammaire hors-contexte de la proposition assertive).
Cette grammaire comprend des règles de reconnaissance de la proposition, du syntagme verbal, du syntagme nominal et du syntagme prépositionnel. Une implémentation est présentée au chapitre V et les règles de cette grammaire sont détaillées dans les annexes. Nous illustrons ici par quelques règles de la grammaire, la reconnaissance des constituants d'une proposition ; puis nous montrons la représentation arborescente de la structure d'une proposition telle qu'elle est créée par le système.

• Exemple d'analyse de la proposition

Les règles utilisées pour la reconnaissance de la proposition ("*L'opposition est visée.*") sont les suivantes :

```
REGLES DE RECONNAISSANCE D'UNE PROPOSITION :
REGLE 3 : PP''-> PP'           Proposition se réécrit Proposition
REGLE 8 : PP'-> PP             Proposition se réécrit Proposition
REGLE 10 : PP-> SN + SV        Proposition se réécrit :
                                     Syntagme Nominal
                                     + Syntagme Verbal

REGLES DE RECONNAISSANCE D'UN SYNTAGME NOMINAL :
REGLE 31 : SN -> D' + N'       Syntagme Nominal se réécrit :
                                     Déterminant
                                     + Nominal
REGLE 57 : D' -> D             Déterminant est un déterminant
REGLE 40 : N' -> N             Nominal se réécrit Nominal
REGLE 41 : N -> F[NOM]         Nominal est un nom

REGLES DE RECONNAISSANCE D'UN SYNTAGME VERBAL :
REGLE 13 : SV -> V''           Syntagme Verbal se réécrit SV
REGLE 18 : V'' -> V'' + A''    SV se réécrit :
                                     SV
                                     + Adjectival
REGLE 25 : V'' -> V'           SV se réécrit Verbal
REGLE 28 : V' -> V             Verbal est un verbe
REGLE 50 : A'' -> A'           Adjectival se réécrit Adjectival
REGLE 52 : A' -> A             Adjectival se réécrit Adjectival
REGLE 53 : A -> F[ADJ]         Adjectival est un adjectif
```

L'échantillon de la grammaire ci-dessus permet de construire une représentation syntaxique en constituants de la proposition : "*L'opposition est visée.*"

```
            proposition
                 |
      +---------^----------+
      sn                   sv
      |                    |
 +----^----+          +---^----+
 d        fnom        v       fadj
 |         |          |        |
"l'"  "opposition"  "e^tre"  "viser"
```

La proposition ci-dessus n'engendre qu'un seul arbre syntaxique. Cependant la plupart des propositions testées par la grammaire fournissent plusieurs solutions concurrentes (comme le montre l'exemple suivant).

• Exemple d'analyse multiple de la proposition

La proposition suivante (qui représente une phrase complète du corpus) engendre deux arbres syntaxiques différents : "*Ses frontières sont difficilement franchissables.*"

Solution N° 1 :

```
                  proposition
                     |
       +-------------^-------------+
       sn                          sv
       |                           |
  +----^----+              +-------^--------+
  d        fnom                           fadj
  |         |        +-----^------+         |
"ses"  "frontie`re"  v            w  "franchissable"
                     |            |
                  "e^tre"  "difficilement"
```

Solution N° 2 :

```
               proposition
                   |
       +-----------^-----------+
       sn                      sv
       |                       |
  +----^----+            +------^------+
  d        fnom          v
  |         |            |     +-------^--------+
"ses"  "frontie`re"  "e^tre"  w               fadj
                              |                |
                       "difficilement"  "franchissable"
```

La différence entre ces deux solutions réside dans le rattachement de l'adverbe "*difficilement*" au verbe "*être*" ou à l'adjectif "*franchissable*". La solution syntaxiquement correcte est la seconde dans laquelle l'adverbe "*difficilement*" se rapporte à la forme "*franchissable*".

- **Remarques**

Il nous faut cependant noter que de nombreuses propositions fournissent des solutions attestées par la grammaire mais non par la langue. En effet, la grammaire est surproductrice de solutions puisqu'elle doit permettre de reconnaître toutes les constructions syntaxiques d'une proposition attestées par la langue. L'utilisation de grammaires satellites doit permettre l'élimination de certaines de ces solutions parasites.

Evidemment, le nombre de solutions produites par l'analyseur croît avec le nombre d'ambiguïtés persistantes (i.e. même après désambiguïsation à l'aide de règles linguistiques contextuelles). Le système doit disposer de stratégies syntaxiques d'élimination pour certaines solutions (cf. § III-4-3). Par ailleurs, nous verrons que certaines solutions produites par cette grammaire doivent être éliminées, d'une part, par l'utilisation de grammaires satellites[40] ou de méta-grammaires[41] ; et d'autre part, par les schémas syntaxiques des verbes de l'analyse fonctionnelle.

III-4-1-2) Grammaires satellites et méta-grammaires

---

[40] Nous définissons les grammaires satellites (ou locales) comme un ensemble de règles de grammaire activables à partir de certains indices présents dans le texte.

[41] Nous définissons une méta-grammaire comme une grammaire utilisant une autre grammaire. Pour plus de détail sur les métaconnaissances se référer à [Pitrat, 90].

En présence de phrases "complexes" (i.e. non reconnues par la grammaire si elles comprennent des phénomènes linguistiques complexes, ou reconnues par la grammaire mais avec un nombre considérable de solutions), l'utilisation de méta-grammaires et de grammaires satellites est indispensable.

- **Intérêt pour les analyses syntaxiques multiples d'une phrase**

Les grammaires satellites pour l'analyse de syntagmes, doivent permettre de traiter certains cas d'ambiguïtés (en particulier, celles liées à la grammaire du syntagme exposées au § III-2-1). Ainsi, pour le problème du rattachement d'un syntagme, ce type de grammaire doit, par exemple, effectuer des vérifications d'accords.
Par exemple, le syntagme "*les chantiers de construction locaux*" n'est pas traité par la grammaire standard de la proposition dans la mesure où il n'y a pas d'accord entre les deux derniers nominaux (cf. les stratégies d'éliminations syntaxiques § III-4-3). Ce syntagme doit être traité par une grammaire satellite privilégiant le rattachement de "*locaux*" à "*chantiers*". Ainsi, le système évite de créer des rattachements de syntagmes erronés. Par contre, pour le syntagme "*la conférence de presse commune*", les deux rattachements possibles de l'adjectif sont créés par la grammaire standard.

```
        sn                                             sn
        |                                              |
   +----^-----+                                  +-----^-----+
   d                                             d
   |     +----^----+                             |     +-----^-----+
"la"  fnom         sp                          "la"              fadj
          |         |                               +----^----+       |
   "confe'rence"                                   fnom       sp  "commun"
               +--^--+                               |         |
               p     sn                      "confe'rence"
               |     |                                     +--^--+
              "de"                                         p     sn
                +----^----+                                |     |
               fnom      fadj                             "de"  fnom
                 |         |                                     |
             "presse"  "commun"                               "presse"
```

- **Utilité pour les phrases non reconnues par la grammaire**

Dans le cas d'une phrase non reconnue par la grammaire, comme la proposition comportant une coordination de syntagmes "*L'ouverture est un espoir et une crainte*", la grammaire standard (de l'analyse en constituants de la proposition) ne pourra pas analyser cette proposition. Par contre, elle peut analyser le début de la proposition "*l'ouverture est un espoir*". Il devient nécessaire de définir une grammaire de reconnaissance du syntagme "*une crainte*" suivant le coordonnant : une *grammaire satellite*. Ensuite, l'analyse syntaxique correcte de cette phrase nécessite une *méta-grammaire* de la coordination permettant de marquer la coordination des deux syntagmes nominaux et de réaliser le rattachement syntaxique correct des segments de cette phrase. Nous étudierons plus en détail, dans le chapitre suivant, l'utilisation de ces différentes grammaires.

III-4-2) Analyse fonctionnelle

Nous commençons par présenter la stratégie générale de l'analyseur fonctionnel ; ensuite, nous illustrons les résultats par des exemples.

III-4-2-1) Présentation générale

L'analyse fonctionnelle utilise les résultats de l'analyse en constituants. Elle va tenter d'associer, à chaque solution syntaxique, le ou les schémas associés aux verbes.
Les schémas syntaxiques des verbes ont été exposés au chapitre II. Nous voulons simplement expliciter ici succinctement comment la reconnaissance des différents composants de la proposition s'effectue.
Un C0 (sujet) ou un C1 (complément d'objet direct) est soit un préverbal appartenant à une liste finie ("*je*", "*tu*",… pour le C0 et "*le*", "*la*",… pour le C1) soit un syntagme nominal de la proposition.
Les compléments C2, C3 et C4 correspondent soit à un préverbal appartenant à une liste finie ("*lui*", "*leur*" pour le C2 et "*y*" pour le C3 et "*en*" pour le C4) soit à un syntagme prépositionnel (introduit par "*à*" pour le C2, par une préposition locative "*dans*", "*sur*",… pour le C3 et par "*de*" pour le C4).
Quant au C5, il est forcément représenté par un syntagme prépositionnel (qui est non anaphorisable).
Par ailleurs, les verbes peuvent relever de plusieurs schémas syntaxiques, comme le montreront les exemples suivants.

III-4-2-2) Exemples d'analyse fonctionnelle

Nous allons étudier deux exemples d'association de la structure syntaxique et fonctionnelle. Dans le premier exemple, le verbe est supposé accepter un seul schéma. Par contre, dans le second exemple, le schéma du verbe est reconnu comme étant à constructions multiples.

A) Schéma du verbe unique

La proposition ("*Le propos devient manichéen.*") ne comprend aucune ambiguïté après désambiguïsation morphologique. Le seul verbe possible est le verbe "*devenir*". L'analyse en constituants de la proposition donne le résultat unique suivant :

```
          proposition
               |
     +---------^---------+
     sn                  sv
     |                   |
 +---^---+           +---^---+
 d      fnom         v       sn
 |       |           |        |
"le"  "propos"  "devenir"   fnom
                              |
                         "maniche'en"
```

A partir de ce résultat, l'analyse fonctionnelle va comparer le schéma syntaxique du verbe "*devenir*". Ce schéma est unique et il est de la forme : C0 V C1.

```
** Schemas ISL =
 -> forme([cat("devenir",v,[pe3,ind,pst,for,ord])],3,3) -
[complements(c0(obl,[sn(nnh,[])]),c1(obl,[sn(nnh,[])]),[],[],[],[])]
```

L'analyse fonctionnelle fournit alors le résultat suivant signifiant que les C0 et C1 sont deux syntagmes nominaux qui ont pu être identifiés :

```
** Structure fonctionnelle =
Verbe : "devenir"
([c0(obl,[sn(nnh,[])]),c1(obl,[sn(nnh,[])]),[],[],[],[]])
C0 : [d[le],fnom[propos]]
C1 : fnom[maniche'en]
C2 : []
C3 : []
C4 : []
C5 : []
```

B) Schéma du verbe à constructions multiples

Nous allons étudier les résultats de la proposition : "*La séparation interdit la syndication.*" Dans ce cas, la proposition est totalement et correctement désambiguïsée par coopération de Morph et Synt. Le verbe est finalement non ambigu ("*interdire*") mais il admet plusieurs schémas syntaxiques différents.

L'analyseur syntaxique en constituants fournit une seule solution arborescente pour cette phrase :

```
                proposition
                  |
      +-----------^-----------+
      sn                      sv
      |                       |
 +----^-----+             +---^----+
 d         fnom           v        sn
 |          |             |        |
"la"  "se'paration"  "interdire"
                              +----^-----+
                              d         fnom
                              |          |
                             "la"  "syndication"
```

Le verbe "*interdire*" est défini comme admettant deux schémas syntaxiques possibles :
C0 V C1 et C0 V C1C2.

```
** Schemas ISL =
 -> forme([cat("interdire",v,[pe3,ind,pps,for,ord])],3,3) -
[complements(c0(obl,[sn(nnh,[])]),c1(obl,[sn(nnh,[])]),[],[],[],[]),
complements(c0(obl,[sn(nnh,[])]),c1(obl,[sn(nhm,[])]),c2(vd,[sn(hum,
["a`"])]),[],[],[])]
```

La solution de l'analyseur fonctionnel fournit pour l'arbre ci-dessus est la suivante :

```
** Structure fonctionnelle =
Verbe : "interdire"
([c0(obl,[sn(nnh,[])]),c1(obl,[sn(nnh,[])]),[],[],[],[]])
C0 : [d[la],fnom[se'paration]]
C1 : [d[la],fnom[syndication]]
C2 : []
C3 : []
C4 : []
C5 : []

 -> Fin d'etude des schemas syntaxiques du verbe
```

La mise en œuvre du second schéma syntaxique du verbe C0 V C1C2, n'a pas abouti, dans la mesure où le complément d'objet indirect C2 n'a pas pu être trouvé dans la phrase.

Dans les exemples présentés ci-dessus, les résultats de l'analyse syntaxique sont uniques. Mais, bien souvent, les solutions fournies par l'analyseur en constituants sont très nombreuses. L'analyse fonctionnelle permet alors d'éliminer certaines arborescences dont les compléments détectés ne répondent pas aux schémas syntaxiques du verbe ; ce qui permet d'éliminer des ambiguïtés créées par la grammaire du syntagme (les analyses erronées de rattachements de syntagmes).

III-4-3) Stratégies syntaxiques de désambiguïsation

Lorsque la proposition comporte des ambiguïtés, toutes les suites possibles (ou trajectoires) de catégories morphologiques seront testées ; mais les trajectoires ainsi construits devront faire l'objet de vérifications minutieuses. Les stratégies syntaxiques que nous utilisons sont de différents types. En particulier, nous utilisons, des stratégies de "courte distance" : vérification d'accords de formes contiguës et élimination des suites de catégories morphologiques impossibles. Nous avons aussi développé des stratégies de "longue distance" pouvant s'appliquer à la fois de gauche à droite et de droite à gauche. Ces règles concernent, entre autre, les stratégies de choix du verbe de la proposition.

III-4-3-1) Stratégies de choix du verbe de la proposition

Nous partons de l'hypothèse de base qu'il y a forcément un et un seul verbe conjugué au sein d'une proposition dans le cadre de notre corpus littéraire. Plusieurs cas peuvent se présenter :

1. Le verbe de la phrase n'est pas ambigu, mais il existe d'autres formes comportant des entrées lexicales renvoyant à un verbe conjugué. Ces dernières seront supprimées.
2. Il n'existe qu'un seul verbe conjugué potentiel ambigu ; il sera considéré comme le verbe et les autres catégories possibles pour cette forme seront éliminées.
3. S'il existe plusieurs verbes conjugués potentiels, chacun sera tour à tour testé et les autres catégories verbes possibles seront éliminées.

Dans le cas d'une désambiguïsation de verbe, un appel à l'agent Morph est réalisé afin qu'il puisse de nouveau effectuer une désambiguïsation morphologique ; l'analyseur syntaxique ne sera appliqué qu'après cette nouvelle intervention de Morph.

Exemple d'utilisation pour la proposition "*L'UC fait la sourde oreille.*" :

```
-> Le verbe de la proposition est ambigu
...[42]
** Liste des formes ambigues de categorie verbe =
 -> forme([cat("faire",v,[pe3,ind,pst,for,ord]),
cat("fait",f,[mas,sng,nom,for,com]),
cat("faire",f,[mas,sng,adj,ppa,for])],3,3)
 -> forme([cat("sourde",f,[fem,sng,nan,for]),
cat("sourdre",v,[pe3,ind,pst,for,ord])],5,5)

** Choix du verbe = forme([cat(faire,v,[pe3,ind,pst,for,ord])],3,3)
...
-> Necessite d'une nouvelle levee d'ambiguites
...
** Choix du verbe = forme([cat(sourdre,v,[pe3,ind,pst,for,ord])],5,5)
```

---

[42] Dans le cas de résultats fournis par le prototype, nous insérons des point de suspension lorsque nous supprimons certains résultats du système pour une meilleure lisibilité.

```
...
-> Necessite d'une nouvelle levee d'ambiguites
...
```

En fait, cette stratégie permet d'éliminer certaines ambiguïtés d'origine morphologique lexicale.

III-4-3-2) Elimination de suites de catégories impossibles

Lorsque la proposition comporte des ambiguïtés que le système n'arrive pas à lever, toutes les suites possibles de catégories pour cette proposition vont être testées. Dans ce cas, une première vérification consiste à éliminer certaines suites de catégories impossibles. Ainsi, nous éliminons les cas les plus fréquents : un déterminant D suivi d'un verbe V, et un préverbal Y suivi d'un nominal F.

Exemples d'utilisation :

```
    -> Differents choix possibles pour la forme : la
          -> Possibilite d'analyse morphologique des formes =
[d,l',[grn,sng,def,fai],f,UC,[grn,nbn,nom,prp],f,fait,[mas,sng,nom,f
or,com],d,la,[fem,sng,def,fai],v,sourdre,[pe3,ind,pst,for,ord],f,ore
ille,[fem,sng,nom,for,com]]

 -> Solution rejetee syntaxiquement (d + v)

    -> Differents choix possibles pour la forme : poste
          -> Possibilite d'analyse morphologique des formes =
[d,l',[grn,sng,def,fai],f,architecte,[mas,sng,nom,for,com],y,s',[grn
,nbn,pe3,pe6,i20,fai],v,e^tre,[pe3,ind,pst,for,aux],f,voir,[mas,sng,
adj,ppa,for],f,re'cuse',[mas,sng,adj,ppa,for],p,pour,[apo,fai],y,ce,
[grn,nbn,in1,fai],f,poste,[fem,sng,nom,for,com]]

 -> Solution rejetee syntaxiquement (y + f)
```

L'élimination de suites de catégories impossibles permet la résolution de certaines ambiguïtés d'origine morphologique lexicale.

III-4-3-3) Vérification d'accords

De même, nous effectuons une vérification d'accords afin d'éliminer rapidement des solutions parasites. Les vérifications mettant en œuvre des nominaux et/ou déterminants (Exemples : D + F, F + F) s'effectuent sur le genre et le nombre ; les vérifications des accords sur des préverbaux et/ou verbes (ex. : Y + V) sont réalisées en fonction de la personne. En cas d'échec des accords, la solution est immédiatement rejetée.

Exemples d'utilisation :

```
    -> Differents choix possibles pour la forme : ce
    -> Differents choix possibles pour la forme : poste
          -> Possibilite d'analyse morphologique des formes =
[d,l',[grn,sng,def,fai],f,architecte,[mas,sng,nom,for,com],y,s',[grn
,nbn,pe3,pe6,i20,fai],v,e^tre,[pe3,ind,pst,for,aux],f,voir,[mas,sng,
adj,ppa,for],f,re'cuse',[mas,sng,adj,ppa,for],p,pour,[apo,fai],d,ce,
[mas,sng,def,fai],f,poste,[fem,sng,nom,for,com]]

 -> Solution rejetee syntaxiquement (non accord de d + f)
...

    -> Differents choix possibles pour la forme : fil
          -> Possibilite d'analyse morphologique des formes =
[d,ce,[mas,sng,def,fai],f,fil,[mas,plu,nom,for,com],p,de,[apo,fai],f
,pasteur,[mas,sng,nom,for,com],v,organiser,[p13,isu,pst,for,ord],p,d
e,[fai,apo],d,les,[grn,plu,def,fai],f,exposition,[fem,plu,nom,for,co
m]]

 -> Solution rejetee syntaxiquement (non accord de d + f)
...
```

Remarque : Lorsqu'il n'y a pas d'accord entre deux nominaux, la solution est rejetée par la grammaire standard de la proposition. Un appel à la grammaire satellite du syntagme nominal doit permettre d'analyser correctement un syntagme comme "*les chantiers de construction locaux*" (avec comme solution unique le rattachement de l'adjectif au nominal "*chantiers*"). Par contre, pour le syntagme "*ce fils*" (exemple ci-dessus) la grammaire satellite doit aboutir à un échec ; la grammaire élimine alors une solution parasite liée à une ambiguïté d'origine morphologique flexionnelle.

III-4-3-4) Stratégies de longue distance

Nous utilisons des règles de longue distance : à partir de certaines informations, on déduit d'autres informations de nature linguistique qui doivent alors être vérifiées. Ces règles peuvent aussi bien être définies de droite à gauche que de gauche à droite sur la proposition. Ainsi, nous avons créé une règle qui vérifie la présence d'un nominal F à droite de tout déterminant D.

Exemple d'utilisation :

```
     -> Differents choix possibles pour la forme : 1992
          -> Possibilite d'analyse morphologique des formes =
[d,une,[fem,sng,nnu,fai],f,premier,[fem,sng,nan,for],f,tranche,[fem,
sng,nom,for,com],v,e^tre,[pe3,ind,fut,for,aux],f,commercialiser,[fem
,sng,adj,ppa,for],p,en,[aau,loc,fai],d,1992,[grn,plu,num]]
 -> Solution rejetee syntaxiquement (d sans f apres)
```

III-4-3-5) Elimination des solutions arborescentes identiques

L'objectif est d'éliminer les arbres syntaxiques possibles lorsqu'ils ont déjà été trouvés. Cette redondance se produit lorsque la grammaire peut utiliser des règles différentes pour arriver aux mêmes constituants de la proposition.

C'est le cas des règles suivantes de reconnaissance d'un syntagme prépositionnel "*du projet*" dans la proposition "*La dimension du projet interdit toute échéance précise.*", qui après regularisations devient "*la dimension de le projet interdit toute e'che'ance pre'cise.*".

Ces deux suites de règles permettant la reconnaissance de "de le projet"	
Regle 37 : N' -> N + SP	Regle 40 : N' -> N
Regle 41 : N -> F[NOM]	Regle 43 : N -> N + SP
Regle 47 : SP -> P + SN	Regle 41 : N -> F[NOM]
Regle de reconnaissance d'une preposition	Regle 47 : SP -> P + SN
Regle 31 : SN -> D' + N'	Regle de reconnaissance d'une preposition
Regle 57 : D' -> D	Regle 31 : SN -> D' + N'
Regle de reconnaissance d'un determinant	Regle 57 : D' -> D
Regle 40 : N' -> N	Regle de reconnaissance d'un determinant
Regle 41 : N -> F[NOM]	Regle 40 : N' -> N
	Regle 41 : N -> F[NOM]

L'arbre créé par ces deux suites de règles est identique ; mais il n'est possible de s'en apercevoir qu'après la construction de cette arborescence. La solution n'est pas retenue et l'analyse fonctionnelle n'est pas réalisée à nouveau. Voici une des solutions proposées par l'analyseur deux fois : la seule différence est l'application des règles mentionnées ci-dessus.

```
                   proposition
                      |
      +---------------^----------------+
      sn                               sv
      |                                |
 +----^----+                  +--------^---------+
 d                                               sn
 |     +---^-----+       +----^-----+            |
"la"  fnom      sp       v          w
       |        |        |          |      +-----^------+
  "dimension"       "interdire"  "toute"  fnom         fadj
             +--^--+                       |            |
             p     sn                 "e'che'ance"  "pre'cis"
             |     |
            "de"
               +---^---+
               d      fnom
               |       |
              "le"  "projet"
```

Cette stratégie permet l'élimination d'ambiguïtés virtuelles (au niveau syntaxique) liées à la grammaire de la proposition standard.

III-5) Bilan

Nous avons prototypé un analyseur syntaxique permettant l'analyse en constituants et l'analyse flexionnelle de propositions. Cet analyseur fournit encore trop de solutions. Mais nous pensons qu'une étude approfondie des règles de la grammaire et des schémas des verbes permettrait de résoudre certains problèmes ; de même, le développement de grammaires satellites devrait faciliter l'élimination de solutions parasites. De plus, l'intégration des agents Sema(ntique) et Pragma(tique) devrait permettre d'éliminer de nombreuses solutions parasites.

Une des qualités de l'analyseur que nous avons présenté est sa flexibilité. Nous étudierons dans le chapitre suivant dans quelle mesure il peut être utilisé pour le traitement de la coordination.

III-5-1) Apports

Nous avons vu que les règles syntaxiques de désambiguïsation permettent essentiellement d'éliminer des ambiguïtés relevant du niveau d'analyse morphologique.

En ce qui concerne les ambiguïtés du niveau syntaxique, elles nous semblent bien plus complexes à traiter. L'agent Synt doit être capable de résoudre certaines ambiguïtés liées à la grammaire du syntagme à l'aide de vérification d'accords et de la mise en place de grammaires satellites. Par ailleurs, lors de l'échec de l'analyse fonctionnelle, des analyses en constituants peuvent être éliminées.

III-5-2) Perspectives

La résolution interne des agents du niveau syntaxique (notamment des agents Segm et Synt) doit être améliorée pour rendre compte d'ambiguïtés liées à la grammaire de la proposition et à la grammaire de la phrase.

• Ambiguïtés sur la nature des propositions

Les ambiguïtés sur la nature des propositions devraient être résolues au niveau syntaxique.

Exemples :

1. " Une fois de plus en ce siècle, cette instabilité [***qui*** prépare la guerre] prévaudra."
2. "Il a indiqué qu'il se rendrait bientôt à Bucarest pour y avoir des contacts directs avec les dirigeants et "voir à [***qui*** on a affaire]" ."

La désambiguïsation sur la nature d'une proposition relative ou interrogative, introduite par "*qui*" comme dans les exemples ci-dessus, doit tenir compte des critères suivants :

- dans le premier exemple, la forme "*qui*" possède un antécédent "*cette instabilité*",
- dans le second exemple, la forme "*qui*" n'a pas d'antécédent, et le verbe "*voir*" peut régir une interrogative directe ; cette information est donnée par le dictionnaire syntaxique des verbes indiquant non seulement les arguments nominaux mais aussi les arguments propositionnels qu'un verbe peut régir.

• Certaines ambiguïtés sur la segmentation de la phrase en propositions

L'agent Segm prédit le nombre de propositions en fonction du nombre d'introducteurs de propositions et du nombre de verbes d'une phrase. Un introducteur de propositions est un "indicateur fort" de catégorie Q comme "*que*", "*qui*" (conjonctions de subordination). Segm utilise aussi des indicateurs dits "faibles" : la catégorie C comme "*et*", "*ou*" (conjonctions de coordination), certains adverbes W comme "*ainsi*", "*puis*" ou encore la catégorie T comme la virgule. Ainsi, si la phrase comporte n (n≥0) introducteurs de propositions, et que chaque introducteur potentiel devient un véritable introducteur de propositions, elle comporte n+1 propositions. L'agent Segm vérifie ensuite que la phrase possède le nombre de verbes conjugués nécessaires (en accord avec Morph) en fonction du nombre de propositions ; puis il effectue le découpage de la phrase en propositions.

Exemple :

"[Elle *visait* également à démythifier une formation] [**dont** la perspective *angoissait* ces femmes, âgées pour la plupart de trente-cinq à cinquante ans,] [**et dont** beaucoup *gardent* de l'école une image négative.]"

Dans ce chapitre, nous avons étudié les compétences des agents Morph et Synt dans une perspective de résolution de solutions multiples liées aux ambiguïtés d'origine morphologique et syntaxique. Le chapitre suivant, nous permettra d'étudier des exemples de résolution coopérative entre différents agents du système.

CHAPITRE IV : RESOLUTION COOPERATIVE

Dans ce chapitre, notre objectif est d'illustrer les résultats obtenus par coopération entre les agents du système TALISMAN II que nous avons implémenté.

Nous commençons par présenter des exemples d'analyse morpho-syntaxique dans TALISMAN II. Ensuite, nous proposons une stratégie coopérative de traitement de la coordination. Pour cela, nous étudions la problématique de la coordination, nous développons les agents Coord(ination) et Segm(entation) permettant l'analyse de phrase comportant un coordonnant ; nous illustrons la stratégie d'analyse de certains cas de coordination par des exemples. Enfin, nous proposons l'élaboration d'une stratégie de traitement d'une forme inconnue.

I) Exemples d'analyse coopérative dans TALISMAN II

Nous allons illustrer par des exemples issus de notre corpus, le déroulement de l'analyse morpho-syntaxique, en détaillant plus particulièrement, la coopération entre agents par l'intermédiaire des envois de messages.

I-1) Un cas simple

Nous allons montrer les résultats fournis par le système lors de l'analyse de la phrase suivante : "*Le propos devient manichéen.*"

Les interactions nécessaires à l'analyse morpho-syntaxique de cette phrase sont présentées dans la figure suivante :

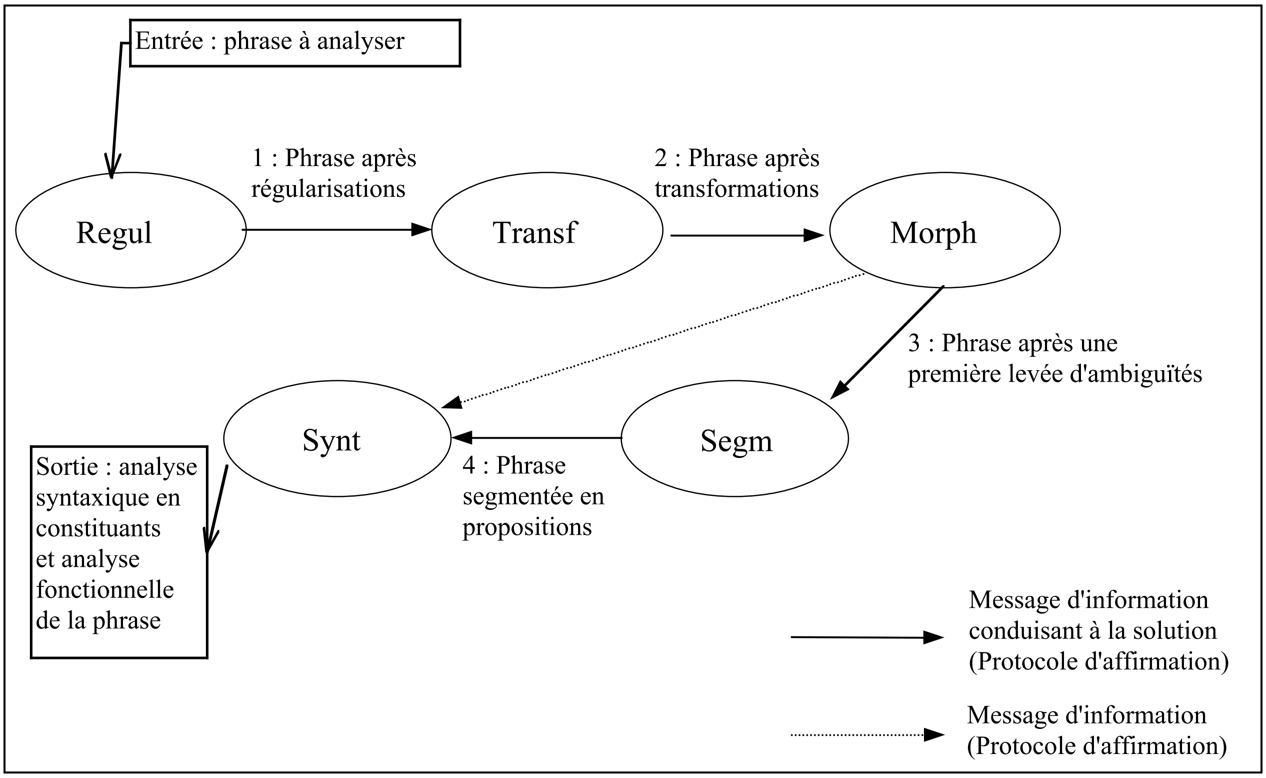

Entrée : phrase à analyser
Regul
1 : Phrase après régularisations
Transf
2 : Phrase après transformations
Morph
3 : Phrase après une première levée d'ambiguïtés
Synt
Segm
4 : Phrase segmentée en propositions
Sortie : analyse syntaxique en constituants et analyse fonctionnelle de la phrase
Message d'information conduisant à la solution (Protocole d'affirmation)
Message d'information (Protocole d'affirmation)

Figure 11 : Analyse d'une phrase "simple"

L'agent Regul fournit les résultats suivants (cf. flèche n°1) :

```
** Apres regularisations =
 -> forme("le","le",1,1)
 -> forme("propos","propos",2,2)
 -> forme("devient","devient",3,3)
 -> forme("maniche'en","maniche'en",4,4)
 -> forme(".",".",5,5)
```

L'agent Transf reconnaît une phrase assertive et envoie le message suivant qui signifie qu'aucune transformation n'a été effectuée (cf. flèche n°2) :

```
message : inform,assert,[transf([],[],[])])
```

L'agent Morph active ses compétences pour effectuer l'analyse morphologique (avec désambiguïsation) :

```
** Apres acces dictionnaire =
 -> forme([cat("le",y,[mas,sng,pe3,i2a,fai]),cat("le",d,[mas,sng,def,
fai])],1,1)
 -> forme([cat("propos",f,[mas,nbn,nom,for,com])],2,2)
 -> forme([cat("devenir",v,[pe3,ind,pst,for,ord])],3,3)
 -> forme([cat("maniche'en",f,[mas,sng,nan,for,com])],4,4)
 -> forme([cat(".",t,[vp1])],5,5)

** Application des regles de levee d'ambiguites =
Regle 1a : y/d + f -> d + f

** Apres levee d'ambiguites =
 -> forme([cat("le",d,[mas,sng,def,fai])],1,1)
 -> forme([cat("propos",f,[mas,nbn,nom,for,com])],2,2)
 -> forme([cat("devenir",v,[pe3,ind,pst,for,ord])],3,3)
 -> forme([cat("maniche'en",f,[mas,sng,nan,for,com])],4,4)
 -> forme([cat(".",t,[vp1])],5,5)
```

La phrase ne comporte plus d'ambiguïtés ; le verbe n'est pas ambigu, ce qui facilite la suite de l'analyse. L'agent Morph envoie un message à l'agent Segm l'informant de ces résultats (cf. flèche n°3 de la Figure 11).

L'agent Segm reconnaît une seule proposition dans la phrase :

```
** Introducteurs potentiels de proposition =
 -> []

** Nombre de propositions = 1

** Apres debut de segmentation =
 -> prin([forme([cat("le",d,[mas,sng,def,fai])],1,1),
forme([cat("propos",f,[mas,nbn,nom,for,com])],2,2),
forme([cat("devenir",v,[pe3,ind,pst,for,ord])],3,3),
forme([cat("maniche'en",f,[mas,sng,nan,for,com])],4,4),
forme([cat(".",t,[vp1])],5,5)])
```

L'agent Synt (informé par l'agent Segm : cf. flèche n°4 de la Figure 11) va procéder à l'analyse syntaxique en constituants et à l'analyse fonctionnelle :

```
** Analyse syntaxique de la proposition =

          Solution arborescente N°1

** Liste des regles de la grammaire utilisees :
Regle 3 : PP''-> PP'
Regle 8 : PP'-> PP
Regle 10 : PP-> N'' SV
Regle 31 : SN -> D' + N'
Regle 57 : D' -> D
Regle de reconnaissance d'un determinant
Regle 40 : N' -> N
Regle 41 : N -> F[NOM]
Regle 13 : SV -> V''
Regle 15 : V'' -> V'' + SN
Regle 25 : V'' -> V'
Regle 28 : V' -> V
Regle de reconnaissance d'un verbe
Regle 36 : SN -> N'
Regle 40 : N' -> N
Regle 41 : N -> F[NOM]

** Representation syntaxique :

         proposition
              |
     +--------^---------+
     sn                 sv
     |                  |
 +---^---+          +---^---+
 d      fnom        v       sn
 |       |          |       |
"le"  "propos"  "devenir"  fnom
                            |
                       "maniche'en"

** Schemas ISL =
 -> forme([cat("devenir",v,[pe3,ind,pst,for,ord])],3,3) -
[complements(c0(obl,[sn(nnh,[])]),c1(obl,[sn(nnh,[])]),[],[],[],[])]
```

```
** Structure fonctionnelle =
Verbe : "devenir"
([c0(obl,[sn(nnh,[])]),c1(obl,[sn(nnh,[])]),[],[],[],[]])
C0 : [d[le],fnom[propos]]
C1 : fnom[maniche'en]
C2 : []
C3 : []
C4 : []
C5 : []
```

L'agent Synt ne trouve que cette solution.

I-2) Analyse d'une phrase comportant des verbes ambigus

La phrase que nous allons étudier est la suivante : "*L'UC fait la sourde oreille*."
Le schéma suivant résume les échanges de message indispensables à l'obtention de l'analyse de cette phrase :

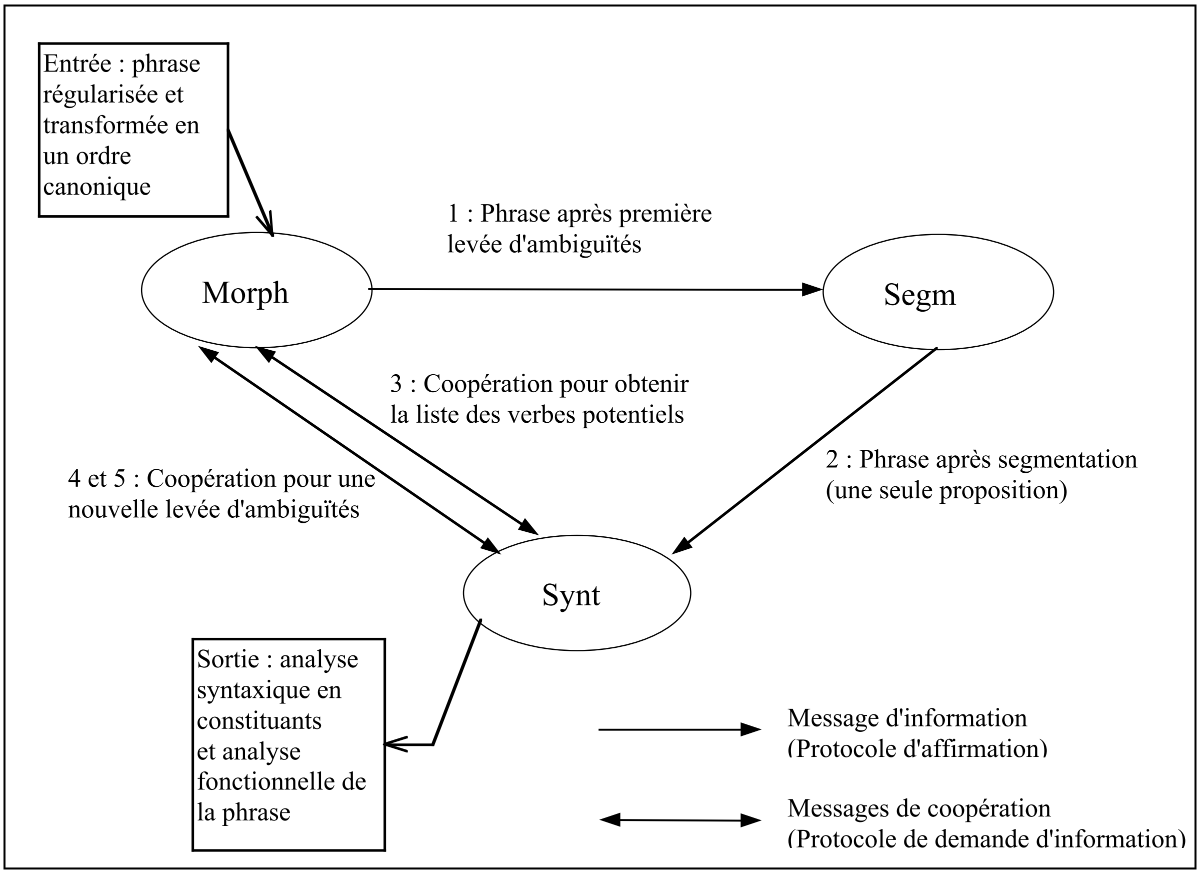

Entrée : phrase régularisée et transformée en un ordre canonique
Morph
1 : Phrase après première levée d'ambiguïtés
Segm
3 : Coopération pour obtenir la liste des verbes potentiels
2 : Phrase après segmentation (une seule proposition)
4 et 5 : Coopération pour une nouvelle levée d'ambiguïtés
Synt
Sortie : analyse syntaxique en constituants et analyse fonctionnelle de la phrase
Message d'information (Protocole d'affirmation)
Messages de coopération (Protocole de demande d'information)

Figure 12 : Analyse d'une phrase comportant des verbes ambigus

Après une première levée d'ambiguïtés, Morph a analysé les formes comme suit :

```
** Apres acces dictionnaire =
-> forme([cat("l'",y,[grn,sng,pe3,i2a,fai]),
cat("l'",d,[grn,sng,def, fai])],1,1)
-> forme([cat("UC",f,[grn,nbn,nom,prp])],2,2)
-> forme([cat("faire",v,[pe3,ind,pst,for,ord]),
cat("fait",f,[mas,sng,nom,for,com]),
cat("faire",f,[mas,sng,adj,ppa,for])],3,3)
-> forme([cat("la",y,[fem,sng,pe3,i2a,fai]),
cat("la",d,[fem,sng,def,fai]),
cat("la",f,[mas,sng,nom,for,com])],4,4)
-> forme([cat("sourde",f,[fem,sng,nan,for]),
cat("sourdre",v,[pe3,ind,pst,for,ord])],5,5)
-> forme([cat("oreille",f,[fem,sng,nom,for,com])],6,6)
-> forme([cat(".",t,[vp1])],7,7)

** Application des regles de levee d'ambiguites =
Regle 1a : y/d + f -> d + f

** Apres levee d'ambiguites =
-> forme([cat("l'",d,[grn,sng,def,fai])],1,1)
-> forme([cat("UC",f,[grn,nbn,nom,prp])],2,2)
-> forme([cat("faire",v,[pe3,ind,pst,for,ord]),
cat("fait",f,[mas,sng,nom,for,com]),
cat("faire",f,[mas,sng,adj,ppa,for])],3,3)
-> forme([cat("la",y,[fem,sng,pe3,i2a,fai]),
cat("la",d,[fem,sng,def,fai]),
cat("la",f,[mas,sng,nom,for,com])],4,4)
-> forme([cat("sourde",f,[fem,sng,nan,for]),
cat("sourdre",v,[pe3,ind,pst,for,ord])],5,5)
-> forme([cat("oreille",f,[fem,sng,nom,for,com])],6,6)
-> forme([cat(".",t,[vp1])],7,7)
```

Cette phrase comporte, après une première levée d'ambiguïtés par Morph, encore trois formes relevant de plusieurs entrées lexicales. L'agent Morph envoie un message d'information à l'agent Segm (cf. flèche n°1 de la Figure 12).

L'agent Segm suppose que cette phrase ne comporte qu'une seule proposition dans la mesure où il ne détecte aucun introducteur de propositions ; il en informe l'agent Synt (cf. flèche n°2 de la Figure 12).

L'agent Synt tente alors d'effectuer l'analyse syntaxique de cette phrase. Il ne trouve pas de verbes non ambigus (i.e. toutes les catégories verbes de la phrase sont couplées avec d'autres catégories possibles) ; il envoie donc un message à Morph pour lui demander la liste des verbes potentiels (cf. flèche n°3 de la Figure 12).

```
message(request,[],donner_formes_ambigues_contenant_cat(v,[forme([ca
t("l'",d,[grn,sng,def,fai])],1,1),forme([cat("UC",f,[grn,nbn,nom,prp
])],2,2),forme([cat("faire",v,[pe3,ind,pst,for,ord]),cat("fait",f,[m
as,sng,nom,for,com]),cat("faire",f,[mas,sng,adj,ppa,for])],3,3),form
e([cat("la",y,[fem,sng,pe3,i2a,fai]),cat("la",d,[fem,sng,def,fai]),c
at("la",f,[mas,sng,nom,for,com])],4,4),forme([cat("sourde",f,[fem,sn
g,nan,for]),cat("sourdre",v,[pe3,ind,pst,for,ord])],5,5),forme([cat(
"oreille",f,[fem,sng,nom,for,com])],6,6),forme([cat(".",t,[vp1])],7,
7)],_10825),_10820)
```

La réponse de Morph est la suivante (cf. flèche n°3 de la Figure 12) :

```
message(answer,assert,donner_formes_ambigues_contenant_cat(...,
[forme([cat("faire",v,[pe3,ind,pst,for,ord]),cat("fait",f,[mas,sng,n
om,for,com]),cat("faire",f,[mas,sng,adj,ppa,for])],3,3),forme([cat("
sourde",f,[fem,sng,nan,for]),cat("sourdre",v,[pe3,ind,pst,for,ord])]
,5,5)]),_11470)
```

En d'autres termes :

```
** Liste des formes ambigues de categorie verbe =
 -> forme([cat("faire",v,[pe3,ind,pst,for,ord]),
cat("fait",f,[mas,sng,nom,for,com]),
cat("faire",f,[mas,sng,adj,ppa,for])],3,3)
 -> forme([cat("sourde",f,[fem,sng,nan,for]),
cat("sourdre",v,[pe3,ind,pst,for,ord])],5,5)
```

L'agent Synt va alors tester les deux verbes potentiels. Lorsqu'un verbe est testé les autres verbes potentiels sont éliminés (i.e. les catégories V pour ces verbes ambigus sont supprimées) ; des ambiguïtés du niveau morphologique d'origine lexicale sont alors traitées par Synt. Il essaie d'abord d'accomplir son but avec le premier verbe : "*faire*".

```
** Choix du verbe = forme([cat(faire,v,[pe3,ind,pst,for,ord])],3,3)
```

L'agent Synt élimine alors les autres verbes potentiels et demande à l'agent Morph d'effectuer une nouvelle levée d'ambiguïtés (cf. flèche n°4 de la Figure 12). Synt envoie le message suivant :

```
message(request,[],lever_ambiguites(_mode,[forme([cat("l'",d,[grn,sn
g,def,fai])],1,1),forme([cat("UC",f,[grn,nbn,nom,prp])],2,2),forme([
cat("faire",v,[pe3,ind,pst,for,ord])],3,3),forme([cat("la",y,[fem,sn
g,pe3,i2a,fai]),cat("la",d,[fem,sng,def,fai]),cat("la",f,[mas,sng,no
m,for,com])],4,4),forme([cat("sourde",f,[fem,sng,nan,for])],5,5),for
me([cat("oreille",f,[fem,sng,nom,for,com])],6,6),forme([cat(".",t,[v
p1])],7,7)],_12350),_12345)
```

L'agent Morph active son module de levée d'ambiguïtés qui permet alors d'éliminer des solutions parasites. D'autres ambiguïtés d'origine lexicale et morphologique flexionnelle peuvent être résolues lors de cette seconde levée d'ambiguïtés grâce à la coopération de Morph et de Synt.

L'agent Morph, après la nouvelle levée d'ambiguïtés (qui lui a permis de désambiguïser la forme "*la*"), envoie le message suivant à l'agent Synt :

```
message(answer,assert,lever_ambiguites(_mode,
[forme([cat("l'",d,[grn,sng,def,fai])],1,1),forme([cat("UC",f,[grn,n
bn,nom,prp])],2,2),forme([cat("faire",v,[pe3,ind,pst,for,ord])],3,3)
,forme([cat("la",y,[fem,sng,pe3,i2a,fai]),cat("la",d,[fem,sng,def,fa
i]),cat("la",f,[mas,sng,nom,for,com])],4,4),forme([cat("sourde",f,[f
em,sng,nan,for])],5,5),forme([cat("oreille",f,[fem,sng,nom,for,com])
],6,6),forme([cat(".",t,[vp1])],7,7)],
/* liste résultat : */
[forme([cat("l'",d,[grn,sng,def,fai])],1,1),forme([cat("UC",f,[grn,n
bn,nom,prp])],2,2),forme([cat("faire",v,[pe3,ind,pst,for,ord])],3,3)
,forme([cat("la",d,[fem,sng,def,fai])],4,4),forme([cat("sourde",f,[f
em,sng,nan,for])],5,5),forme([cat("oreille",f,[fem,sng,nom,for,com])
],6,6),forme([cat(".",t,[vp1])],7,7)]),_13352)
```

La proposition est alors totalement désambiguïsée. L'agent Synt produit trois analyses possibles (variant par l'interprétation de "*sourde oreille*", la forme "*sourde*" pouvant être traité comme un nom ou un adjectif), dont la solution suivante :

```
** Representation syntaxique :
      proposition
           |
    +------^------+
    sn            sv
    |             |
 +--^--+       +--^---+
 d    fnom     v      sn
 |     |       |      |
"l'"  "UC"  "faire"
                 +----^-----+
                 d
                 |     +----^-----+
                "la"  fadj       fnom
                       |           |
                    "sourde"  "oreille"

** Schemas ISL =
 -> forme([cat("faire",v,[pe3,ind,pst,for,ord])],3,3) -
[complements(c0(obl,[sn(nnh,[])]),c1(obl,[sn(nnh,[])]),[],[],[],[]),
complements(c0(obl,[sn(nnh,[])]),c1(obl,[sn(nnh,[])]),c2(vd,[sn(hum,
["a`"])]),[],[],[]),
complements(c0(obl,[sn(nnh,[])]),c1(obl,[sn(nnh,[])]),[],c3(obl,[sn(
hum,["a`"])]),[],[]),
complements(c0(obl,[sn(nnh,[])]),c1(obl,[sn(nhm,[])]),[],[],c4(obl,[
sn(nhu,["de"])]),[])]

** Structure fonctionnelle =
Verbe : "faire"
([c0(obl,[sn(nnh,[])]),c1(obl,[sn(nnh,[])]),[],[],[],[]])
C0 : [d[l'],fnom[UC]]
C1 : [d[la],[fadj[sourde],fnom[oreille]]]
C2 : []
C3 : []
C4 : []
C5 : []
-> Fin d'etude des schemas syntaxiques du verbe
```

L'agent Synt va alors traiter le deuxième verbe potentiel : "*sourdre*". L'agent Synt demande à l'agent Morph d'effectuer une nouvelle levée d'ambiguïtés (cf. flèche n°5 de la Figure 12). Ensuite, l'agent Synt essaiera de reconnaître les syntagmes de la proposition ; puis de leur associer une structure fonctionnelle.

```
** Choix du verbe =
forme([cat(sourdre,v,[pe3,ind,pst,for,ord])],5,5)
```

Mais, dans ce cas, la proposition comporte encore des ambiguïtés (que Morph n'a pas réussi à lever lors de sa seconde tentative d'éliminations de solutions parasites). En effet, les formes "*fait*" et "*la*" sont encore ambiguës.

```
-> Proposition comportant des ambiguites =
    -> Differents choix possibles pour la forme : fait
    -> Differents choix possibles pour la forme : la
```

Ensuite, toutes les suites possibles de catégories morphologiques sont étudiées. Certaines sont éliminées (grâce aux stratégies syntaxiques d'élimination), comme la suivante :

```
          -> Possibilite d'analyse morphologique des formes =
[d,l',[grn,sng,def,fai],f,UC,[grn,nbn,nom,prp],f,faire,[mas,sng,adj,
ppa,for],d,la,[fem,sng,def,fai],v,sourdre,[pe3,ind,pst,for,ord],f,
oreille,[fem,sng,nom,for,com]]
 -> Solution rejetee syntaxiquement (d + v)
```

En fait, les stratégies syntaxiques utilisées ici permettent de résoudre certaines ambiguïtés d'origine lexicale ou morphologique flexionnelle. Au total, six solutions possibles sont proposées (considérant le verbe "*sourdre*"), toutes erronées. Par exemple, la suite suivante d'analyse morphologique des formes de la phrase est analysée par l'agent Synt qui fournit une analyse fonctionnelle possible :

```
[d,l',[grn,sng,def,fai],f,UC,[grn,nbn,nom,prp],f,faire,[mas,sng,adj,
ppa,for],y,la,[fem,sng,pe3,i2a,fai],v,sourdre,[pe3,ind,pst,for,ord],
f,oreille,[fem,sng,nom,for,com]]

** Structure fonctionnelle =
Verbe : "sourdre" ([c0(obl,[sn(nnh,[])]),[],[],[],[],[]])
C0 : [d[l'],[fnom[UC],fadj[faire]]]
C1 : []
C2 : []
C3 : []
C4 : []
C5 : []
```

Le schéma unique du verbe "*sourdre*" est C0 V. L'analyse ci-dessus montre que le préverbal "*la*" et le syntagme nominal composé de la seule forme "*oreille*", n'ont donc pas pu être interprétés par l'analyse fonctionnelle ; cette solution doit donc être éliminée. L'analyse fonctionnelle doit disposer de critères d'élimination d'une solution possible. Elle doit, par ailleurs, prendre en compte la présence de compléments circonstanciels (de temps, lieu,…) non régis par le verbe, i.e. non répertoriés dans le schéma syntaxique du verbe. La seule solution correcte fournie par l'agent Synt est celle présentée dans la page précédente qui considère le verbe "*faire*" comme le verbe de la proposition.

II) Stratégie de traitement de la coordination

Nous exposons succinctement la problématique des phénomènes linguistiques complexes. Ensuite, nous présentons une première stratégie d'analyse de coordination (inter ou intra-proposition) que nous avons développée ; nous détaillons donc les agents Segm et Coord indispensables à ce traitement. Enfin, nous illustrons par deux exemples les stratégies que nous avons prototypées.

II-1) Cas particulier des phénomènes linguistiques complexes

Les phénomènes linguistiques complexes sont souvent source d'ambiguïtés relevant de différents niveaux (morphologique, syntaxique,…) ; nous les qualifions de problèmes "transversaux". L'analyse de phénomènes linguistiques complexes dans un système multi-agents a déjà été présentée dans [Stéfanini & Warren, 96a]. Notre objectif est d'étudier des stratégies de résolution de ces problèmes faisant intervenir des connaissances de natures diverses ; i.e. des connaissances sur la segmentation de phrases en propositions, l'analyse syntaxique, l'analyse morphologique, la coordination, les ellipses,…

Nous avons étudié les phénomènes linguistiques complexes suivants (ayant déjà fait ou faisant l'objet d'études dans notre équipe[43]) : les transformations[44] (phrases interrogatives, impératives,…), les coordinations, les anaphores, les ellipses, les négations (cf. chapitre 6). Nous proposons ici une étude plus approfondie de l'agent transversal Coord dont le but est de construire des structures coordonnées inter ou intra-propositions. Il doit reconnaître les coordinations de formes et de syntagmes ainsi que les énumérations et il construit les structures coordonnées. Nous avons formulé des heuristiques possibles sur la reconnaissance de la coordination et sur la détermination de sa portée. L'objectif étant de fournir un cadre d'agent transversal pour faciliter la modélisation des linguistes de phénomènes linguistiques complexes.

II-2) Problématique de la coordination

La coordination est un phénomène linguistique très fréquent et pourtant relativement peu étudié en TAL car sa mise en œuvre informatique est complexe [Kac & Rindflesch, 88], [Lavelli & Stock, 90]. Dans notre équipe, une étude syntaxique de la coordination dans une grammaire LFG du français est en cours [Brun, 96].
Une coordination est signalée par la présence d'un coordonnant catégorisé C[45], ou d'une virgule.
Nous distinguons différents types de coordination :

• Les coordinations inter-phrastiques
Nous dirons qu'une coordination de deux phrases est caractérisée par la présence d'un coordonnant au début de la seconde phrase. Ce type de coordination est une connection argumentative jouant un rôle pragmatique ; il s'agit de connecteurs textuels qui permettent d'assurer la cohérence inter-phrastique et/ou la progression thématique.

Exemple :

"Aucune de ces perspectives n'a provoqué de réactions vraiment visibles. ***Et*** même l'annonce par Saint-Gobain d'un bénéfice de 4,3 milliards de francs, en hausse de 6 %, a entraîné vendredi une dépréciation du titre."

---

[43] Cette catégorisation n'est pas limitative ; elle représente l'état de nos connaissances actuelles pour certains phénomènes linguistiques complexes.

[44] Nous nous sommes fondé sur les travaux d'Alain Berrendonner, notamment sur l'impersonnel [Berrendonner, 90]. Ce qui rejoint l'idée de "grammaire transformationnelle" proposée par Chomsky. Nous utilisons des règles de transformations permettant de ramener des phrases à un ordre canonique. Un modèle transformationnel est exposé en détail dans [Ruwet, 70].

[45] Nous ne considérons "comme conjonctions de coordination les mots qui sont capables de connecter non seulement deux phrases, mais aussi deux termes de phrase" [Berrendonner, 83]. Les formes "et", "ni", "ou", "soit", "tantôt",… sont donc catégorisées C.
Par contre, "tout connecteur qui ne peut lier que deux phrases successives, ou bien deux expressions prédicatives, sera traité comme un adverbe anaphorique". Les formes "or", "ainsi", "donc", "mais", "puis", "car",… ne sont donc pas catégorisées C.

• Les coordinations intra-phrastiques

Une coordination intra-phrastique est caractérisée par la présence d'un coordonnant à l'intérieur d'une phrase[46]. Nous distinguons les coordinations inter-propositions des coordinations intra-propositions.

- Coordinations inter-propositions

Une coordination inter-propositions consiste à coordonner deux propositions au sein d'une même phrase.

Exemple :

"La pièce est composée de deux parties trop distinctes ***et*** le propos devient manichéen." (*P1 et P2*)

- Coordinations intra-propositions de syntagmes ou de formes

Ce type de coordination apparaît souvent entre constituants (formes ou syntagmes) de même catégorie.

Exemple de coordination de syntagmes :

"Nous devons beaucoup à des pays comme *la Tchécoslovaquie ou la Pologne*, dont on n'oublie pas le rôle joué par Solidarnosc." (*SN ou SN*)

Exemple de coordination de formes[47] :

"Au cours des deux dernières années, a-t-il notamment déclaré, les autorités centrales ont cherché à agir avec *patience et réflexion*, tentant de résoudre des problèmes difficiles exclusivement par des moyens *pacifiques et politiques*." (*Fnom et Fnom* puis *Fadj et Fadj*)

Bien entendu, il existe d'autres types de coordinations : des coordinations de constituants de natures différentes, des coordinations doubles (comme [*ou…ou…*]), des énumérations ; et le problème de la détermination de la portée de la coordination peut être complexe (comme le montrent les exemples suivants).

Exemples :

1. "Logiquement, elles devraient se redéployer ***en France, en Grande-Bretagne, aux Pays-Bas et en Belgique***, mais il est peu probable que la volonté politique impose cette logique militaire." → (*SP, SP, SP et SP*)

2. "Ces BCD, dotées dès leur ouverture de cinq cents livres, seront ouvertes non seulement ***[pendant le temps scolaire]*** *mais aussi* ***[pendant les inter-classes],*** **[** ***les mercredis*** **]** ***et*** **[*****durant les petites vacances***], puisqu'elles seront placées sous la responsabilité d'un animateur de la Ville." → (*SP mais aussi* (*SP* , *SN et SP)*)

---

[46] i.e. dans une phrase mais non en début de phrase.

[47] Ce sont souvent des syntagmes réduits à une seule forme.

3. "Un concours de circonstances, auquel [[***l'enquête et le livre***] ***d'un journaliste***] ***et*** [***la détermination*** [***d'un juge*** [***d'instruction***]]] ne sont pas totalement étrangers, a fait éclater il y a quelques mois, à Lyon, une grave affaire de proxénétisme dans laquelle était impliqué un certain nombre de policiers." → (*SN et SN)SP et (SN SP SP*)

II-3) Les agents Segm et Coord

Nous présentons ces deux agents indispensables au traitement d'un coordonnant.

II-3-1) Agent Segm

L'agent Segm(entation) permet la segmentation d'une phrase en propositions ; en particulier, il reconnaît les propositions principales, subordonnées et coordonnées.

Pour atteindre son objectif, l'agent Segm dispose des **connaissances** suivantes :

- une liste des introducteurs potentiels de proposition (les introducteurs de propositions subordonnées de catégorie Q, les coordonnants C,…),
- un compteur utilisé pour compter le nombre de verbes et le nombre d'introducteurs de proposition.

Les **compétences** de l'agent Segm sont les suivantes :

- la reconnaissance de la proposition principale,
- la détection du début possible[48] d'une proposition subordonnée,
- la détection du début d'un segment coordonné (proposition ou syntagme),
- la prédiction du nombre de propositions en fonction du nombre de verbes et du nombre d'introducteurs potentiels de proposition.

L'agent Segm utilise ses compétences pour atteindre son but en respectant sa **stratégie générale** :

1. La détection du nombre n1 d'introducteurs potentiels de proposition ; le nombre n2 de propositions possibles est alors égale n1 + 1 (= n2).
2. Le comptage du nombre n3 de verbes conjugués (donnés par Morph) de la phrase ; ce nombre correspond aussi au nombre possible de propositions de la phrase.
3. La prédiction du nombre de propositions par comparaison des nombres n2 et n3 :
    - n2 = n3 : le nombre de propositions est n2 (=n3),
    - n2 ≠ n3 : le nombre de propositions est d'abord supposé égal au nombre de verbes conjugués (n3) ; le nombre de verbes conjugués étant considéré comme un indice plus fort que le nombre potentiel d'introducteurs. Si l'analyse syntaxique de la phrase

---

[48] La présence de la catégorie Q n'indique pas forcément une proposition subordonnée ; comme dans la phrase (42) suivante où la forme "*que*" correspond à la fin d'une négation restrictive : "*Cette possibilité n'a été accordée que pour les obligations d'une durée inférieure à six mois.*"

échoue avec cette possibilité du nombre de propositions, le nombre de propositions sera supposé égal au nombre d'introducteurs incrémenté de 1 (= n2).

- n3 = 0 : dans ce cas (où la proposition ne comporte que des verbes ambigus), l'agent Segm se basera sur le nombre d'introducteurs de proposition (=n2).

4. Le découpage de la phrase en propositions en fonction du nombre de propositions et des introducteurs repérés. Ensuite, l'agent Segm coopère avec l'agent Coord (si un coordonnant a été détecté) ou sinon avec l'agent Synt ; si la coopération de ces derniers aboutit à un échec, retour en 3 (la prédiction de l'agent Segm sur le nombre de propositions doit être erronée).

La stratégie proposée pour la prédiction du nombre de propositions doit être affinée, entre autre, pour prendre en considération, d'une part, les cas où certains verbes de la phrase sont ambigus, et d'autre part, la nature des indicateurs de propositions (indicateurs "forts" ou "faibles").

II-3-2) Agent Coord

L'agent Coord(ination) doit reconnaître la portée des coordonnants en construisant la structure syntaxique des phrases comportant une coordination de propositions ou de syntagmes.

Cet agent utilise ses **connaissances** :

- la liste des indicateurs de coordination ("*et*", "*ou*", "*ni*", "*soit*", "*tantôt*" de catégorie C et la virgule de catégorie T),
- une méta-grammaire de la coordination (sa mise en œuvre est indispensable dans la mesure où la grammaire de la proposition standard ne reconnaît pas la catégorie C).

Ces connaissances permettent à l'agent Coord d'effectuer ses **compétences** :

- la détection et la construction de propositions et de syntagmes coordonnées,
- la détection et la construction d'énumérations.

L'agent met en œuvre ses compétences en utilisant la **stratégie générale** suivante :

1. Reconnaissance du coordonnant.
2. Détection des segments coordonnés (à gauche et à droite du coordonnant).
3. Reconnaissance de la portée de la coordination (grâce aux informations de l'agent Segm) et traitement approprié :

1er cas : la construction de la coordination inter-propositions (si le nombre de propositions indiqué par Segm est supérieur à 1) :

- coopération avec Synt pour l'obtention des arborescences de chacune des propositions (le découpage en propositions est fourni par Segm),
- construction de l'arbre des propositions coordonnées.

2ème cas : la construction de la coordination intra-proposition (si le nombre de propositions indiqué par Segm est égal à 1) :

- coopération avec Synt pour l'obtention de l'arborescence de la proposition principale et du syntagme coordonné (le découpage étant proposé par Segm).
- recherche du syntagme à gauche du coordonnant (i.e. à partir de la droite de la proposition principale) ; pour l'instant, nous considérons que les syntagmes coordonnés doivent être du même type (ex. : deux syntagmes nominaux ou deux syntagmes prépositionnels),

- construction de l'arbre de la proposition comportant des syntagmes coordonnés.

4. Si échec, retour en 3. La prédiction de l'agent Segm doit être erronée.

Cette stratégie permet de reconnaître des cas triviaux de coordination inter et intra-proposition (comme le montrent les exemples § II-3). La prochaine étape est de considérer, entre autre, les traitements suivants (repris section IV-3) :

- Coopération avec Segm : en particulier, lorsque plusieurs solutions sont possibles pour le nombre ou la délimitation des propositions.
- Coopération avec l'agent Ellip : pour permettre une coordination inter-propositions avec ellipse du sujet dans la seconde.
- Amélioration des capacités de l'agent Coord : il doit être capable de reconnaître la coordination de syntagmes de nature différente, d'effectuer différents rattachements possibles du syntagme coordonné, et de traiter les énumérations.

II-4) Exemples de traitement d'un coordonnant

Nous illustrons par deux exemples issus de notre corpus le traitement de la coordination dans TALISMAN II : le premier exemple concerne une coordination entre deux propositions (non elliptiques) ; le second exemple comprend une coordination entre deux syntagmes nominaux.

II-4-1) Traitement d'une coordination inter-propositions

La phrase que nous allons étudier est la suivante : "*La pièce a deux parties et le propos devient manichéen*."
Les envois de message les plus importants pour l'aboutissement de l'analyse sont présentés dans le schéma suivant :

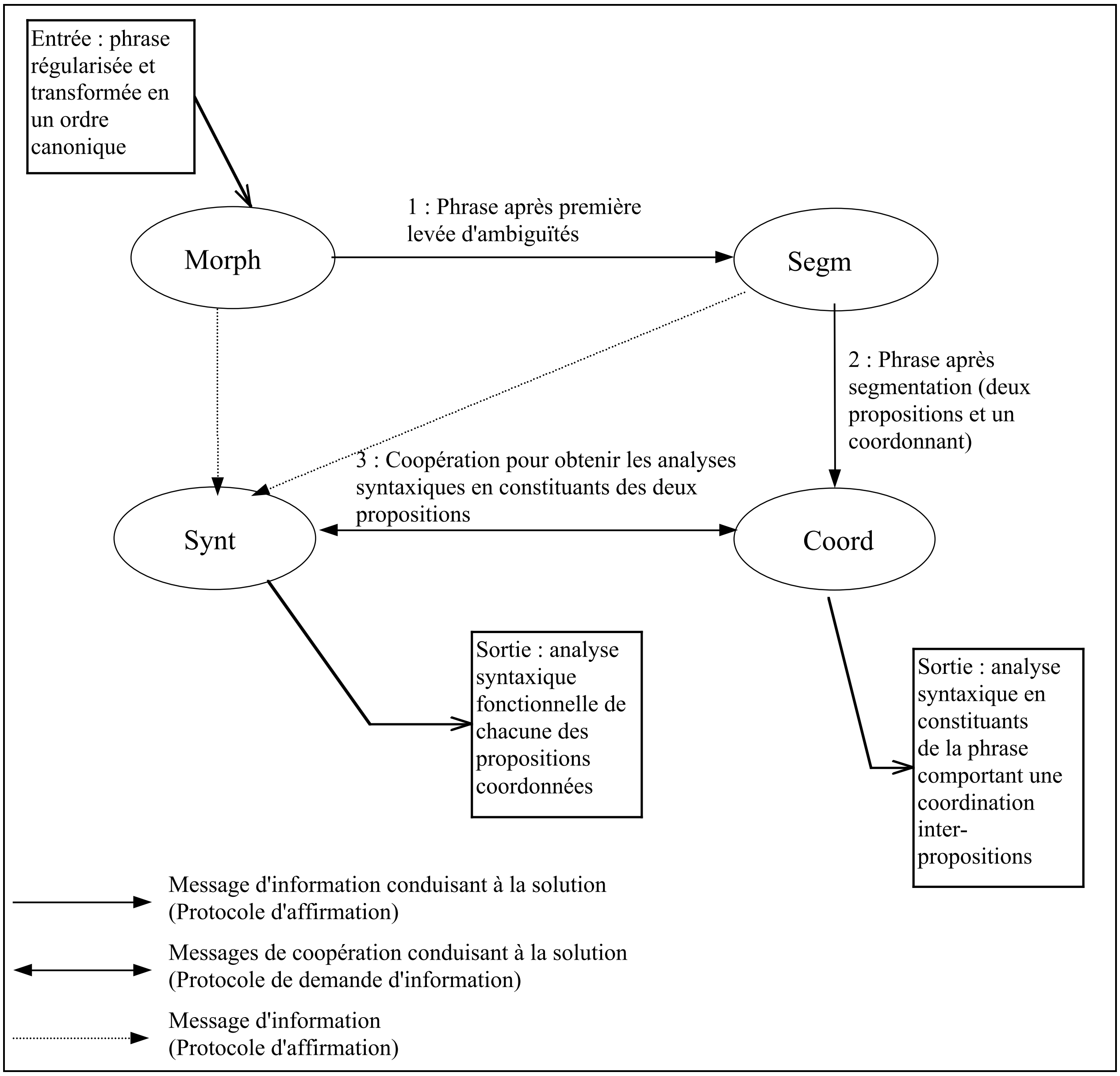

Entrée : phrase régularisée et transformée en un ordre canonique
Morph
1 : Phrase après première levée d'ambiguïtés
Segm
2 : Phrase après segmentation (deux propositions et un coordonnant)
3 : Coopération pour obtenir les analyses syntaxiques en constituants des deux propositions
Synt
Coord
Sortie : analyse syntaxique fonctionnelle de chacune des propositions coordonnées
Sortie : analyse syntaxique en constituants de la phrase comportant une coordination inter-propositions
Message d'information conduisant à la solution (Protocole d'affirmation)
Messages de coopération conduisant à la solution (Protocole de demande d'information)
Message d'information (Protocole d'affirmation)

Figure 13 : Analyse d'une coordination inter-propositions

Après désambiguïsation des catégories par l'agent Morph, ses résultats sont les suivants : la phrase est totalement désambiguïsée et elle comporte deux verbes conjugués.

```
** Apres levee d'ambiguites =
 -> forme([cat("la",d,[fem,sng,def,fai])],1,1)
 -> forme([cat("pie`ce",f,[fem,sng,nom,for,com])],2,2)
 -> forme([cat("avoir",v,[pe3,ind,pst,for,aux])],3,3)
 -> forme([cat("deux",d,[grn,plu,num])],4,4)
 -> forme([cat("partie",f,[fem,plu,nom,for,com])],5,5)
 -> forme([cat("et",c,[])],6,6)
 -> forme([cat("le",d,[mas,sng,def,fai])],7,7)
 -> forme([cat("propos",f,[mas,nbn,nom,for,com])],8,8)
 -> forme([cat("devenir",v,[pe3,ind,pst,for,ord])],9,9)
 -> forme([cat("maniche'en",f,[mas,sng,nan,for,com])],10,10)
 -> forme([cat(".",t,[vp1])],11,11)
```

```
** Liste des formes non ambigues de categorie verbe =
 -> forme([cat("avoir",v,[pe3,ind,pst,for,aux])],3,3)
 -> forme([cat("devenir",v,[pe3,ind,pst,for,ord])],9,9)
```

L'agent Morph envoie un message d'information contenant ses résultats à l'agent Segm (cf. flèche n°1 de la Figure 13).

L'agent Segm examine la phrase et conclut qu'elle comporte deux propositions et un coordonnant ; il procède donc à un début de segmentation.

```
** Introducteurs potentiels de proposition =
 -> forme([cat("et",c,[])],6,6)

** Nombre de propositions = 2
 -> Detection d'un coordonnant

** Apres debut de segmentation =
 -> deb_prin([forme([cat("la",d,[fem,sng,def,fai])],1,1),
forme([cat("pie`ce",f,[fem,sng,nom,for,com])],2,2),forme([cat("avoir
",v,[pe3,ind,pst,for,aux])],3,3),forme([cat("deux",d,[grn,plu,num])]
,4,4),forme([cat("partie",f,[fem,plu,nom,for,com])],5,5)])
 -> deb_coord([forme([cat("et",c,[])],6,6),
forme([cat("le",d,[mas,sng,def,fai])],7,7),forme([cat("propos",f,[ma
s,nbn,nom,for,com])],8,8),forme([cat("devenir",v,[pe3,ind,pst,for,or
d])],9,9),forme([cat("maniche'en",f,[mas,sng,nan,for,com])],10,10),f
orme([cat(".",t,[vp1])],11,11)])
```

L'agent Segm informe l'agent Coord de ses résultats (cf. flèche n°2 de la Figure 13). L'agent Coord conclut que la phrase est composée de deux propositions coordonnées.

```
 -> Presence d'une coordination inter-proposition
```

Coord demande donc à l'agent Synt (cf. flèche n°3 de la Figure 13) d'effectuer l'analyse syntaxique en constituants des deux propositions qu'il lui transmet. L'agent Synt effectue donc l'analyse en constituants puis l'analyse fonctionnelle de ces deux propositions :

```
Proposition =
[forme([cat(la,d,[fem,sng,def,fai])],1,1),forme([cat(pie`ce,f,[fem,s
ng,nom,for,com])],2,2),forme([cat(avoir,v,[pe3,ind,pst,for,aux])],3,
3),forme([cat(deux,d,[grn,plu,num])],4,4),forme([cat(partie,f,[fem,p
lu,nom,for,com])],5,5)]
 -> Proposition totalement desambiguisee

** Analyse syntaxique de la proposition =

         Solution arborescente N°1

** Liste des regles de la grammaire utilisees :
Regle 3 : PP''-> PP'
Regle 8 : PP'-> PP
Regle 10 : PP-> N'' SV
Regle 31 : SN -> D' + N'
Regle 57 : D' -> D
Regle de reconnaissance d'un determinant
Regle 40 : N' -> N
Regle 41 : N -> F[NOM]
Regle 13 : SV -> V''
Regle 15 : V'' -> V'' + SN
Regle 25 : V'' -> V'
Regle 28 : V' -> V
Regle de reconnaissance d'un verbe
Regle 31 : SN -> D' + N'
Regle 57 : D' -> D
Regle de reconnaissance d'un determinant
Regle 40 : N' -> N
Regle 41 : N -> F[NOM]

** Representation syntaxique :

        proposition
             |
     +-------^--------+
     sn               sv
     |                |
 +---^---+         +--^---+
 d      fnom       v      sn
 |       |         |      |
"la"  "pie`ce"  "avoir"
                      +---^----+
                      d       fnom
                      |        |
                    "deux"  "partie"

** Schemas ISL =
 -> forme([cat("avoir",v,[pe3,ind,pst,for,aux])],3,3) -
[complements(c0(obl,[sn(nnh,[])]),c1(obl,[sn(nnh,[])]),[],[],[],[]),
complements(c0(obl,[sn(hum,[])]),c1(obl,[sn(nhu,[])]),[],c3(obl,[sn(
nhu,loc)]),[],[]),
complements(c0(obl,[sn(nnh,[])]),c1(obl,[sn(nnh,[])]),[],[],[],c5(ob
l,[sn(nnh,["pour","contre"])])))]

** Structure fonctionnelle =
Verbe : "avoir"
([c0(obl,[sn(nnh,[])]),c1(obl,[sn(nnh,[])]),[],[],[],[]])
C0 : [d[la],fnom[pie`ce]]
C1 : [d[deux],fnom[partie]]
C2 : []
C3 : []
C4 : []
C5 : []

 -> Fin d'etude des schemas syntaxiques du verbe
```

Proposition =
[forme([cat(le,d,[mas,sng,def,fai])],7,7),forme([cat(propos,f,[mas,nbn,nom,for,com])],8,8),forme([cat(devenir,v,[pe3,ind,pst,for,ord])],9,9),forme([cat(maniche'en,f,[mas,sng,nan,for,com])],10,10),forme([cat(.,t,[vp1])],11,11)]

```
 -> Proposition totalement desambiguisee

** Analyse syntaxique de la proposition =

         Solution arborescente N°1

** Liste des regles de la grammaire utilisees :
Regle 3 : PP''-> PP'
Regle 8 : PP'-> PP
Regle 10 : PP-> N'' SV
Regle 31 : SN -> D' + N'
Regle 57 : D' -> D
Regle de reconnaissance d'un determinant
Regle 40 : N' -> N
Regle 41 : N -> F[NOM]
Regle 13 : SV -> V''
Regle 15 : V'' -> V'' + SN
Regle 25 : V'' -> V'
Regle 28 : V' -> V
Regle de reconnaissance d'un verbe
Regle 36 : SN -> N'
Regle 40 : N' -> N
Regle 41 : N -> F[NOM]

** Representation syntaxique :

         proposition
              |
     +--------^---------+
     sn                 sv
     |                  |
 +---^---+          +---^---+
 d      fnom        v       sn
 |       |          |       |
"le"  "propos"  "devenir"  fnom
                            |
                      "maniche'en"

** Schemas ISL =
 -> forme([cat("devenir",v,[pe3,ind,pst,for,ord])],9,9) -
[complements(c0(obl,[sn(nnh,[])]),c1(obl,[sn(nnh,[])]),[],[],[],[])]

** Structure fonctionnelle =
Verbe : "devenir"
([c0(obl,[sn(nnh,[])]),c1(obl,[sn(nnh,[])]),[],[],[],[]])
C0 : [d[le],fnom[propos]]
C1 : fnom[maniche'en]
C2 : []
C3 : []
C4 : []
C5 : []

 -> Fin d'etude des schemas syntaxiques du verbe
```

A la réception des résultats de l'agent Synt (cf. flèche n°3 de la Figure 13), Coord réalise la construction de la phrase (grâce à sa méta-grammaire de la coordination) comportant deux propositions coordonnées :

```
** Representation syntaxique de la phrase :

                          phrase
                            |
                            c
                            |
                            et
                            |
             +--------------^---------------+
         principale                     principale
             |                              |
     +-------^--------+            +--------^---------+
     sn               sv           sn                 sv
     |                |            |                  |
 +---^---+         +--^---+    +---^---+          +---^---+
 d      fnom       v      sn   d      fnom        v       sn
 |       |         |      |    |       |          |       |
"la"  "pie`ce"  "avoir"       "le"  "propos"  "devenir"  fnom
                      +---^----+                          |
                      d       fnom                   "maniche'en"
                      |        |
                    "deux"  "partie"
```

II-4-2) Traitement d'une coordination intra-proposition

Nous allons étudier un exemple de coordination de deux syntagmes nominaux au sein d'une proposition. La phrase est la suivante : "*L'ouverture est un espoir et une crainte*."
Le schéma suivant résume les échanges de message indispensables à l'obtention de l'analyse de cette phrase :

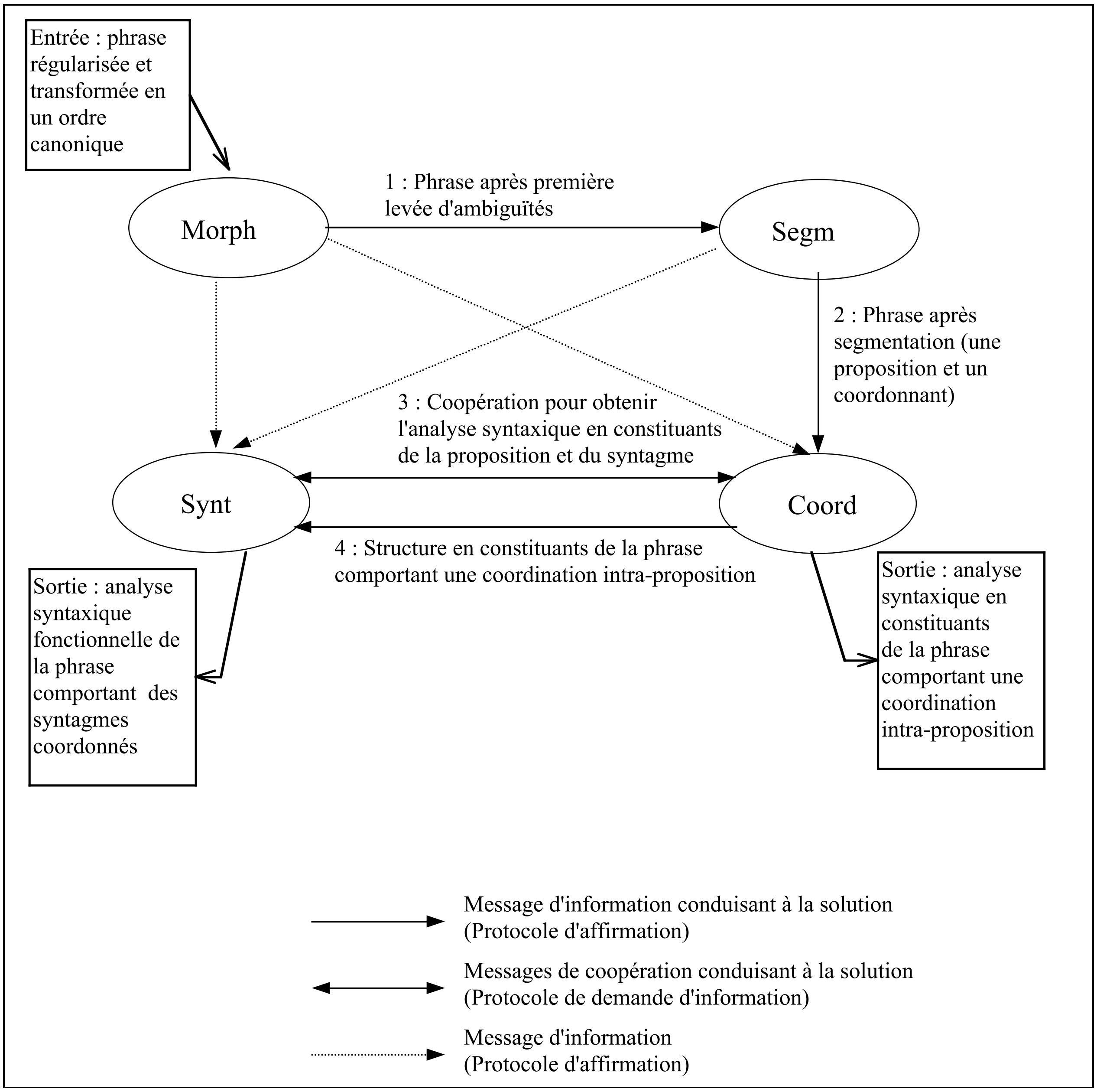

Entrée : phrase régularisée et transformée en un ordre canonique
Morph
1 : Phrase après première levée d'ambiguïtés
Segm
2 : Phrase après segmentation (une proposition et un coordonnant)
3 : Coopération pour obtenir l'analyse syntaxique en constituants de la proposition et du syntagme
Synt
Coord
4 : Structure en constituants de la phrase comportant une coordination intra-proposition
Sortie : analyse syntaxique fonctionnelle de la phrase comportant des syntagmes coordonnés
Sortie : analyse syntaxique en constituants de la phrase comportant une coordination intra-proposition
Message d'information conduisant à la solution (Protocole d'affirmation)
Messages de coopération conduisant à la solution (Protocole de demande d'information)
Message d'information (Protocole d'affirmation)

Figure 14 : Analyse d'une coordination intra-proposition

L'agent Morph désambiguïse correctement cette phrase et en informe l'agent Segm (cf. flèche n°1) :

```
** Apres levee d'ambiguites =
 -> forme([cat("l'",d,[grn,sng,def,fai])],1,1)
 -> forme([cat("ouverture",f,[fem,sng,nom,for,com])],2,2)
 -> forme([cat("e^tre",v,[pe3,ind,pst,for,aux])],3,3)
 -> forme([cat("un",d,[mas,sng,nnu,fai])],4,4)
 -> forme([cat("espoir",f,[mas,sng,nom,for,com])],5,5)
 -> forme([cat("et",c,[])],6,6)
 -> forme([cat("une",d,[fem,sng,nnu,fai])],7,7)
 -> forme([cat("crainte",f,[fem,sng,nom,for,com])],8,8)
 -> forme([cat(".",t,[vp1])],9,9)

** Liste des formes non ambigues de categorie verbe =
 -> forme([cat("e^tre",v,[pe3,ind,pst,for,aux])],3,3)
```

L'agent Segm détecte un coordonnant, mais un seul verbe : l'agent Segm conclut que la phrase ne comporte qu'une seule proposition (il en informe l'agent Coord : cf. flèche n°2 de la Figure 14).

```
** Introducteurs potentiels de proposition =
 -> forme([cat("et",c,[])],6,6)

** Nombre de propositions = 1
 -> Detection d'un coordonnant

** Apres debut de segmentation =
 -> deb_prin([forme([cat("l'",d,[grn,sng,def,fai])],1,1),
forme([cat("ouverture",f,[fem,sng,nom,for,com])],2,2),forme([cat("e^
tre",v,[pe3,ind,pst,for,aux])],3,3),forme([cat("un",d,[mas,sng,nnu,f
ai])],4,4),forme([cat("espoir",f,[mas,sng,nom,for,com])],5,5)])
 -> deb_coord([forme([cat("et",c,[])],6,6),
forme([cat("une",d,[fem,sng,nnu,fai])],7,7),forme([cat("crainte",f,[
fem,sng,nom,for,com])],8,8),forme([cat(".",t,[vp1])],9,9)])
```

L'agent Coord (après analyse des résultats de Segm) conclut que la phrase comporte une coordination intra-proposition.

```
 -> Presence d'une coordination intra-proposition
```

L'agent Coord demande donc à l'agent Synt (cf. flèche n°3 de la Figure 14) d'effectuer l'analyse de la proposition précédant le coordonnant ("*l'ouverture est un espoir*") et d'effectuer l'analyse du syntagme suivant le coordonnant ("*une crainte*").

L'agent Synt fournit les résultats suivants :

```
 -> Proposition totalement desambiguisee
...
** Representation syntaxique :

          proposition
               |
      +--------^---------+
      sn                 sv
      |                  |
 +----^----+          +--^---+
 d        fnom        v      sn
 |         |          |      |
"l'"  "ouverture"  "e^tre"
                         +---^---+
                         d      fnom
                         |       |
                        "un"  "espoir"

** Reconnaissance d'un syntagme nominal = [d,une,[fem,sng,nnu,fai],
f,crainte,[fem,sng,nom,for,com]]

          Solution arborescente N°1

** Liste des regles de la grammaire utilisees :
Regle 31 : SN -> D' + N'
Regle 57 : D' -> D
Regle de reconnaissance d'un determinant
Regle 40 : N' -> N
Regle 41 : N -> F[NOM]
```

```
** Representation syntaxique :
      sn
      |
 +---^----+
  d       fnom
  |         |
"une"  "crainte"
```

L'agent Synt a analysé le début de la proposition principale "*l'ouverture est un espoir*" ; et il a reconnu que le syntagme "*une crainte*" était de type nominal (grâce aux grammaires satellites du syntagme : la grammaire de reconnaissance de la nature du syntagme, puis d'analyse du syntagme).

L'agent Coord tente alors de retrouver un syntagme de même nature (i.e. nominal) en parcourant de droite à gauche la liste des syntagmes composant la proposition précédant le coordonnant. L'agent Coord trouve le syntagme nominal "*un espoir*" et il réalise la coordination de ce dernier syntagme avec celui suivant le coordonnant : "*une crainte*".

L'agent Coord construit donc l'arborescence suivante :

```
** Representation syntaxique de la phrase :
          proposition
               |
      +--------^---------+
      sn                 sv
      |                  |
 +----^----+          +--^--+
 d         fnom       v     sn
 |           |        |     |
"l'"  "ouverture"  "e^tre"  c
                            |
                            et
                            |
                    +-------^--------+
                    sn               sn
                    |                |
                +---^---+        +---^----+
                d      fnom      d       fnom
                |        |       |         |
               "un"  "espoir"  "une"  "crainte"
```

L'agent Coord informe l'agent Synt (cf. flèche n°4 de la Figure 14) de ces résultats sur la structure en constituants de la proposition. L'agent Synt peut alors réaliser l'analyse fonctionnelle de cette proposition ; il fournit les résultats suivants :

```
** Schemas ISL =
 -> forme([cat("e^tre",v,[pe3,ind,pst,for,aux])],3,3) -
[complements(c0(obl,[sn(nnh,[])]),[],[],[],[],[]),
complements(c0(obl,[sn(nnh,[])]),c1(obl,[sn(nhm,[])]),[],[],[],[]),
complements(c0(obl,[sn(nnh,[])]),[],c2(obl,[sn(nhm,[])]),[],[],[]),
complements(c0(obl,[sn(nnh,[])]),[],[],[],[],c5(obl,[sn(nhu,[de])]))]

** Structure fonctionnelle =
Verbe : "e^tre" ([c0(obl,[sn(nnh,[])]),[],[],[],[],[]])
C0 : [d[l'],fnom[ouverture]]
C1 : []
C2 : []
C3 : []
C4 : []
C5 : []
```

```
** Structure fonctionnelle =
Verbe : "e^tre"
([c0(obl,[sn(nnh,[])]),c1(obl,[sn(nhm,[])]),[],[],[],[]])
C0 : [d[l'],fnom[ouverture]]
C1 : [c[et[[sn[[d[un],fnom[espoir]]],sn[[d[une],fnom[crainte]]]]]]]
C2 : []
C3 : []
C4 : []
C5 : []

 -> Fin d'etude des schemas syntaxiques du verbe
```

Sur les quatre schémas possibles du verbe "*être*", seulement deux ont permis l'analyse fonctionnelle ; ici encore, la première solution proposée par l'analyseur (avec C0V) pourrait être éliminée.

III) Traitement d'un mot inconnu

Nous avons présenté dans le chapitre précédent la stratégie adoptée par l'agent Morph lors de la détection d'une forme inconnue par le dictionnaire.
Nous allons étudier l'analyse morpho-syntaxique (correspondant à une première étape dans l'élaboration d'une stratégie d'analyse de formes inconnues par Morph) d'une phrase comportant une forme ("*disc-jockeys*") absente du dictionnaire. Cette phrase est la suivante : "*Les disc-jockeys acceptaient ces cassettes*."
Les envois de message les plus importants pour l'obtention de l'analyse morpho-syntaxique de cette phrase sont présentés dans le schéma suivant :

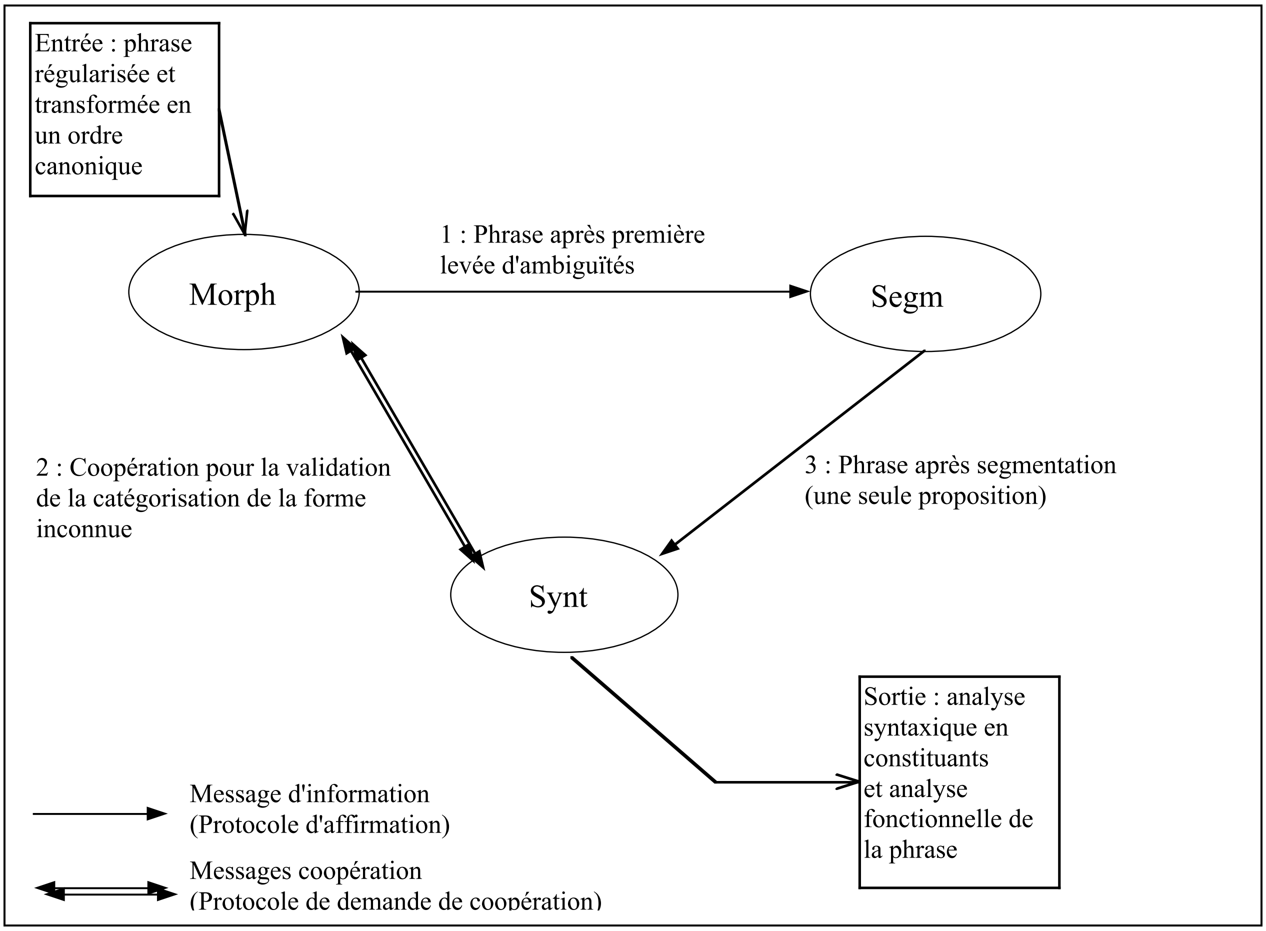

Entrée : phrase régularisée et transformée en un ordre canonique
Morph
1 : Phrase après première levée d'ambiguïtés
Segm
2 : Coopération pour la validation de la catégorisation de la forme inconnue
3 : Phrase après segmentation (une seule proposition)
Synt
Sortie : analyse syntaxique en constituants et analyse fonctionnelle de la phrase
Message d'information (Protocole d'affirmation)
Messages coopération (Protocole de demande de coopération)

Figure 15 : Analyse d'une phrase comportant une forme inconnue

L'agent Morph va tenter une catégorisation de cette forme inconnue par F (nominal) ; puis il effectue la levée d'ambiguïtés.

```
** Detection d'une forme absente du dictionnaire =
 -> disc-jockeys
    -> Essai de categorisation par f

** Apres acces dictionnaire =
 -> forme([cat("les",y,[grn,plu,i2a,pe3,fai]),
cat("les",d,[grn,plu,def,fai])],1,1)
 -> forme([cat("disc-jockeys",f,[grn,nbn,nan])],2,2)
 -> forme([cat("accepter",v,[pe6,ind,imp,for,ord])],3,3)
 -> forme([cat("ces",d,[grn,plu,def,fai])],4,4)
 -> forme([cat("cassette",f,[fem,plu,nom,for,com])],5,5)
 -> forme([cat(".",t,[vp1])],6,6)

** Application des regles de levee d'ambiguites =
Regle 1a : y/d + f -> d + f

** Apres levee d'ambiguites =
 -> forme([cat("les",d,[grn,plu,def,fai])],1,1)
 -> forme([cat("disc-jockeys",f,[grn,nbn,nan])],2,2)
 -> forme([cat("accepter",v,[pe6,ind,imp,for,ord])],3,3)
 -> forme([cat("ces",d,[grn,plu,def,fai])],4,4)
 -> forme([cat("cassette",f,[fem,plu,nom,for,com])],5,5)
 -> forme([cat(".",t,[vp1])],6,6)
```

L'agent Segm (informé par l'agent Morph des résultats de l'analyse morphologique : cf. flèche n°1 de la Figure 15) reconnaît que cette phrase ne comporte qu'une seule proposition. L'agent Morph envoie ses résultats à Synt en lui précisant qu'il n'est pas sûr de ses catégorisations : le message est du type "`request propose`" (cf. flèche n°2 de la Figure 15). L'agent Synt répond selon les résultats de son analyse syntaxique par "`answer([modify, confirm, desagree, no_opinion])`".

L'agent Synt est informé par l'agent Segm que la phrase ne comporte qu'une seule proposition (cf. flèche n°3 de la Figure 15).

```
*********************************************************
                        AGENT : syntaxe

Lecture du message suivant emis par l'agent  morphologie

message(request,answer([modify,confirm,desagree,no_opinion]),[ph_lv_
amb([forme([cat("les",d,[grn,plu,def,fai])],1,1),forme([cat("disc-
jockeys",f,[grn,nbn,nan])],2,2),forme([cat("accepter",v,[pe6,ind,imp
,for,ord])],3,3),forme([cat("ces",d,[grn,plu,def,fai])],4,4),forme([
cat("cassette",f,[fem,plu,nom,for,com])],5,5),forme([cat(".",t,[vp1]
)],6,6)]),form_namb_v([forme([cat("accepter",v,[pe6,ind,imp,for,ord]
)],3,3)])],_8471)
...
 -> Le verbe de la proposition n'est pas ambigu
...
-> Proposition totalement desambiguisee

** Analyse syntaxique de la proposition =

          Solution arborescente N°1

** Liste des regles de la grammaire utilisees :
Regle 3 : PP''-> PP'
Regle 8 : PP'-> PP
Regle 10 : PP-> N'' SV
Regle 31 : SN -> D' + N'
Regle 57 : D' -> D
Regle de reconnaissance d'un determinant
Regle 40 : N' -> N
Regle 42 : N -> F[NAN]
Regle 13 : SV -> V''
Regle 15 : V'' -> V'' + SN
Regle 25 : V'' -> V'
Regle 28 : V' -> V
Regle de reconnaissance d'un verbe
Regle 31 : SN -> D' + N'
Regle 57 : D' -> D
Regle de reconnaissance d'un determinant
Regle 40 : N' -> N
Regle 41 : N -> F[NOM]
```

```
** Representation syntaxique :

              proposition
                   |
       +-----------^-----------+
       sn                       sv
       |                        |
 +----^-----+              +---^----+
  d         fnom           v        sn
  |          |             |        |
"les"  "disc-jockeys"  "accepter"
                                 +---^----+
                                 d       fnom
                                 |         |
                               "ces"  "cassette"

** Schemas ISL =
 -> forme([cat("accepter",v,[pe6,ind,imp,for,ord])],3,3) -
[complements(c0(obl,[sn(nnh,[])]),[],[],[],[],[]),
complements(c0(obl,[sn(nnh,[])]),c1(obl,[sn(nnh,[])]),[],[],[],[]),
complements(c0(obl,[sn(nnh,[])]),c1(obl,[sn(nhm,[])]),[],[],[],c5(ob
l,[sn(nhu,[avec])]))]

** Structure fonctionnelle =
Verbe : "accepter" ([c0(obl,[sn(nnh,[])]),[],[],[],[],[]])
C0 : [d[les],fnom[disc-jockeys]]
C1 : []
C2 : []
C3 : []
C4 : []
C5 : []

** Structure fonctionnelle =
Verbe : "accepter"
([c0(obl,[sn(nnh,[])]),c1(obl,[sn(nnh,[])]),[],[],[],[]])
C0 : [d[les],fnom[disc-jockeys]]
C1 : [d[ces],fnom[cassette]]
C2 : []
C3 : []
C4 : []
C5 : []

 -> Fin d'etude des schemas syntaxiques du verbe
```

L'agent Synt ayant réussi à atteindre son but local (il ne trouve d'ailleurs qu'une seule solution d'analyse en constituants qui est correcte), envoie un message de confirmation à l'agent Morph (cf. flèche n°2 de la Figure 15) pour valider la catégorie F de la forme inconnue "*disc-jockeys*".

```
**********************************************************
                         AGENT : morphologie

Lecture du message suivant emis par l'agent  syntaxe

message(answer,confirm,[ph_lv_amb([forme([cat("les",d,[grn,plu,def,f
ai])],1,1),forme([cat("disc-jockeys",f,[grn,nbn,nan])],2,2),
forme([cat("accepter",v,[pe6,ind,imp,for,ord])],3,3),forme([cat("ces
",d,[grn,plu,def,fai])],4,4),forme([cat("cassette",f,[fem,plu,nom,fo
r,com])],5,5),forme([cat(".",t,[vp1])],6,6)]),form_namb_v([forme([ca
t("accepter",v,[pe6,ind,imp,for,ord])],3,3)])],_18172)
```

```
===> Message correctement traite par l'agent

***************************************************************
* FIN D'UNE SOLUTION (Toutes les boites aux lettres sont vides) *
***************************************************************
```

Cette solution est la seule qui est produite par le système.

IV) Vers de nouvelles résolutions coopératives

Nous présentons des stratégies de résolution coopérative pour les ambiguïtés présentées au chapitre III (relevant des niveaux d'analyse morphologique et syntaxique) que nous n'avons pas encore résolues.

IV-1) Coopération entre Regul et Morph

Cette coopération permet de résoudre des ambiguïtés relevant du niveau d'analyse morphologique.

IV-1-1) Ambiguïtés d'origine typographique

- **Traitement des majuscules en début de phrase**

L'agent Regul envoie à l'agent Morph la forme qui comporte une majuscule. Ce dernier analyse les flexions et consulte son dictionnaire. Si la base n'est pas dans le dictionnaire, elle sera considérée comme un nom propre. Si elle est dans le dictionnaire, il peut s'agir de deux cas possibles : nom propre et nom commun (même si la forme a plus de chance d'être un nom commun, il ne faut pas éliminer des cas comme "*Paris est…*"). Il y a alors multiplicité des solutions. Morph doit renvoyer l'information à l'agent Regul.

- **Traitement d'autres ponctuations**

Pour le traitement du point, l'agent Regul coopère avec l'agent Morph afin de déterminer si le point est un délimitateur de phrase ou non. Pour cela, Regul lui envoie la forme précédant (ou comportant) le point afin que Morph utilise ses connaissances lui permettant de reconnaître s'il se trouve en présence d'une abréviation ("*cf.*"), d'un acronyme ("*S.N.C.F.*"),… L'agent Morph doit faire appel à ses stratégies de traitement de formes inconnues. Si l'agent ne reconnaît pas la forme, le point sera considéré comme une marque de fin de phrase.

IV-1-2) Ambiguïtés d'origine morphologique dérivationnelle

• Traitement du trait d'union
L'agent Regul interrogera Morph sur le découpage d'une forme comportant un (ou plusieurs) trait(s) d'union ("*peut-on*", "*vis-à-vis*"). Morph dispose d'une stratégie particulière de traitement du trait d'union. Il analyse les flexions et regarde dans le dictionnaire des mots composés. Si la base est présente, elle ne sera pas segmentée ("*vis-à-vis*"). Puis Morph vérifie si la forme est du type verbe suivi d'un préverbal ; si c'est le cas la forme est segmentée ("*peut*" et "*on*"). Ainsi, pour "*rendez-vous*" les deux solutions seront conservées par l'agent Regul.

Exemple :
Une forme "rendez-vous" : Nom VS Deux formes : "rendez" Verbe et "vous" Préverbal

IV-2) Coopérations avec Sema et/ou Pragm

Ces coopérations devraient permettre d'éliminer de nombreuses ambiguïtés. Nous présentons une problématique liée à la grammaire du syntagme, pour les ambiguïtés présentées au chapitre III (ambiguïtés relevant des niveaux d'analyses morphologiques et syntaxiques) et non encore résolues.

• D'autres ambiguïtés sur le rattachement de syntagmes
L'agent Synt peut ne pas arriver à lever de manière locale l'ambiguïté sur le rattachement d'un syntagme, en particulier lors de la présence d'un coordonnant ; il a alors besoin de connaissances d'autres agents (comme Coord ou Segm).
Exemples de rattachement de syntagmes prépositionnels ambigus :

1. "Le succès considérable et l'enthousiasme *du public…*"
   pour un lecteur humain, a priori : [SN et [SN SP]$_{SN}$]
2. "… l'apprentissage et la pratique *de la lecture…*"
   pour un lecteur humain, a priori : [[SN et SN]$_{SN}$ SP]

Dans le premier exemple, le syntagme prépositionnel "*du public*" doit être rattaché au syntagme nominal "*l'enthousiasme*" le précédant ; alors que dans le second exemple, le syntagme prépositionnel "*de la lecture*" se rapporte aux deux syntagmes nominaux le précédant ("*l'apprentissage*" et "*la pratique*"). Des connaissances sémantiques et pragmatiques sont nécessaires, mais sont-elles toujours suffisantes ?

IV-3) D'autres coopérations de Coord

Nous avons déjà présenté quelques exemples de détermination de la portée de la coordination. D'autres cas sont à prendre en considération :

• Cas de coordinations intra-phrastiques

Nous avons proposé la stratégie suivante :

1. Si l'on remarque une forte symétrie, sur un secteur court[49], de formes de même catégories (grâce à Morph), il s'agit certainement d'une coordination de bas niveau (coordination de formes de même catégorie).

   Exemples :

   "… destinées à habituer les enfants à lire des textes [*utiles et attrayants*],…"(Fadj et Fadj)

   "… risquait de paralyser [*purement et simplement*] leur fonctionnement." (W et W)

2. Si l'on remarque une forte symétrie sur un secteur court, de syntagmes (grâce à Synt), on se trouve en présence d'une coordination de syntagmes.

   Exemples :

   "L'ouverture, c'est [*un espoir et une crainte*]." (SN et SN)

   "Toutes ces modifications poussent évidemment [*à des évolutions et à des changements*], même s'il ne faut pas aller trop vite." (SP et SP)

Un des problèmes que l'agent Coord doit traiter ici est la détection de la fin du segment coordonné que nous n'avons pas pris en compte jusqu'à maintenant.

Cette stratégie doit être affinée pour reconnaître les coordinations de formes ou de syntagmes de catégories différentes, comme dans l'exemple suivant (déjà cité précédemment) :

   Exemple :

   "… seront ouvertes non seulement pendant le temps scolaire mais aussi pendant les inter-classes, [*les mercredis*] *et* [*durant les petites vacances*]…" (*…SN et SP*)

3. Si l'on n'a trouvé aucune symétrie (de formes ou syntagmes) autour du coordonnant, il s'agit certainement d'une coordination de haut niveau, c'est-à-dire une coordination de propositions ; l'agent Segm aura par ailleurs détecté la présence d'un autre verbe conjugué (à l'aide de Morph) pouvant relever d'une coordination inter-propositions.

   Exemple :

   "[La pièce est composée de deux parties trop distinctes] *et* [le propos devient manichéen]" (P1 *et* P2)

---

[49] En général, ces formes se situent autour du coordonnant.

Cette stratégie triviale doit être affinée, pour rendre compte notamment des cas suivants : les énumérations, l'ellipse dans la proposition suivant le coordonnant, et les cas d'ambiguïté effective en contexte. Elle doit prendre en considération d'autres critères comme par exemple, la présence de virgules dans la phrase comportant un coordonnant.

Cas d'énumérations :
Si une virgule est située juste avant la symétrie de formes ou de syntagmes entourant le coordonnant *et*, on peut se trouver en présence d'une énumération i.e. une coordination de plus de deux catégories ou syntagmes (ex. : SN, SN, *et* SN).

> "... à l'initiative de [MM Michel Giraud, député (RPR), président de l'association des maires de France]**,** [Jean Puech, sénateur (UDF), président de l'association des présidents des conseils généraux]**,** ***et*** [Jacques Blanc, député (UDF), président de la conférence des présidents des conseils régionaux]." (SN, SN, et SN)

Présence d'un segment elliptique dans la proposition suivant le coordonnant :
Si l'on trouve une virgule juste devant un indicateur tel que *et* et aucune symétrie, on se trouve en présence d'une coordination de haut niveau (P1, *et* P2), qui aura été signalée par la segmentation. Par ailleurs, l'ellipse du sujet dans deux propositions coordonnées est courante et nécessite l'action d'un agent Ellip permettant le rétablissement du segment elliptique (cf. chapitre 6).

> Exemple :
> "[Le Front populaire azéri décrète la journée de samedi journée de "deuil national" en Azerbaidjan] ***, et*** [{restitution du sujet elliptique} appelle les Azéris à la désobéissance civile]. (P1, et P2)

Cas d'ambiguïté effective :
Lorsque le nombre de verbes conjugués est ambigu, les agents du système doivent tester toutes les solutions possibles : un verbe avec une proposition, ou deux verbes avec deux propositions…. L'objectif est de pouvoir rendre compte de l'ambiguïté effective suivante "*Il détient le pouvoir et le contrôle*" devant être analysé comme :

- une proposition : "*il détient [le pouvoir et le contrôle]*" (coordination de syntagmes),
- deux propositions : "*[il détient le pouvoir] et [(il) le contrôle]*" (coordination inter-propositions avec ellipse du sujet).

• Cas de coordinations inter-phrastiques

Ce type de coordination doit être traité par Segm qui se trouve en présence d'un coordonnant en début de phrase. Ces coordinations peuvent introduire des ellipses ou des anaphores, auquel cas le fait de savoir qu'on est en présence d'une coordination inter-phrastiques permet d'aller chercher l'antécédent dans la phrase précédente. La détection d'un coordonnant en début de phrase entraîne le marquage de la phrase comme coordonnée à la précédente par CRD[50]. Mais, comme le montre l'exemple suivant, de nombreuses coopérations peuvent être nécessaires : rétablissement des segments elliptiques avec l'agent Ellip, reconnaissance et détermination de la portée de la négation avec l'agent Néga,…

---

[50] CRD signifie que la phrase est coordonnée.

> Exemple :
> "Bien sûr, j'ai d'abord le sentiment d'être Allemand, mais dans tous nos documents vous trouverez des références à l'Europe. ***Et*** pas seulement à l'Ouest."
> → "[et pas seulement à l'Ouest] CRD" qu'on pourrait interpréter de la manière suivante "et {vous ne trouverez} pas seulement {des références à l'Europe} [à→de] l'Ouest."

La coordination est un phénomène complexe dans la mesure où l'utilisation du principe de symétrie des constituants coordonnés, n'est bien souvent pas suffisant à la reconnaissance de la portée d'un coordonnant. Il reste un travail linguistique important à réaliser avant d'obtenir un traitement de la coordination satisfaisant.

Nous avons présenté différents résultats fournis par le système TALISMAN II ; afin de mieux comprendre le déroulement du système, nous développons, dans le chapitre suivant, certains aspects informatiques de l'implémentation.

PARTIE III :
MISE EN ŒUVRE

CHAPITRE V : IMPLANTATION

Le langage de PROgrammation LOGique (PROLOG) est souvent utilisé en traitement automatique des langues. En effet, les principes d'unification et de retour arrière (*backtracking*) de Prolog sont bien adaptés à un système non déterministe fournissant toutes les solutions comme pour le système d'analyse morpho-syntaxique que nous avons développé.

La première maquette de TALISMAN a été réalisée en Bim Prolog 3.0 sur station Sparc, elle comprenait les 5 agents suivants : Pret, Transf, Morph, Synt et Segm [Stéfanini, 93]. M.H. Stéfanini a effectué une restructuration de cette maquette sous Prolog II+ (en syntaxe Edinburgh) sur PC (sous Windows 95). Prolog II+ (version 3.2 de prologIA, syntaxe Edinburgh) est un langage approprié à notre application ; de plus, l'utilisation de Windows permet une portabilité aisée du système.

I) Description externe du système

I-1) Mise en œuvre de TALISMAN II

• Chargement de TALISMAN II
Après avoir chargé Prolog II+ (en syntaxe Edinburgh), l'utilisateur doit insérer le fichier "`proto.p2E`". L'insertion des fichiers nécessaires au bon déroulement du système est alors réalisée.

```
to_begin :-
  insere_module_dessin,
  inserer("C:\TALISMAN\K-Regul.p2E"), /* agent regul(arisation) */
  inserer("C:\TALISMAN\K-Transf.p2E"), /* agent transf(ormation) */
  inserer("C:\TALISMAN\K-Morph.p2E"), /* agent morph(ologie) */
  inserer("C:\TALISMAN\K-Segm.p2E"), /* agent segm(entation) */
  inserer("C:\TALISMAN\K-Synt.p2E"), /* agent synt(axe) */
  inserer("C:\TALISMAN\K-Coord.p2E"), /* agent coord(ination) */
  inserer("C:\TALISMAN\K-Comm.p2E"), /* protocoles de communication */
  inserer("C:\TALISMAN\K-Dico.p2E"), /* dictionnaire des formes */
  inserer("C:\TALISMAN\K-verb.p2E"), /* schemas des verbes */
  inserer("C:\TALISMAN\phrases.p2E"), /* phrases tests */
  line,
  creer_fenetres.
```

Le prédicat `insere_module_dessin` permet le chargement du module de dessin de Prolog II+ indispensable à l'affichage arborescent de l'analyse en constituants.

```
insere_module_dessin:-getenv("c:\talisman\",S),
    conc_string(S,"dessin.mo",S1),
    load(S1).
```

Le prédicat `creer_fenetres` permet l'obtention du système de fenêtrage que nous développons dans la section suivante.

- **Lancement de l'analyse d'une phrase**

Le lancement de l'analyse morpho-syntaxique d'une phrase est possible par le prédicat `go`. Les phrases sont prédéfinies et reconnaissables par un numéro. Ainsi la commande "`go(2).`" permet l'exécution de la phrase associée au numéro 2.

Les résultats produits par chacun des agents sont visibles dans la fenêtre qui leur est associée.

- **Arrêt de l'analyseur**

Chaque fin d'une solution d'analyse morpho-syntaxique de la phrase est indiquée par le message :

```
*****************************************************************
* FIN D'UNE SOLUTION (Toutes les boites aux lettres sont vides) *
*****************************************************************
```

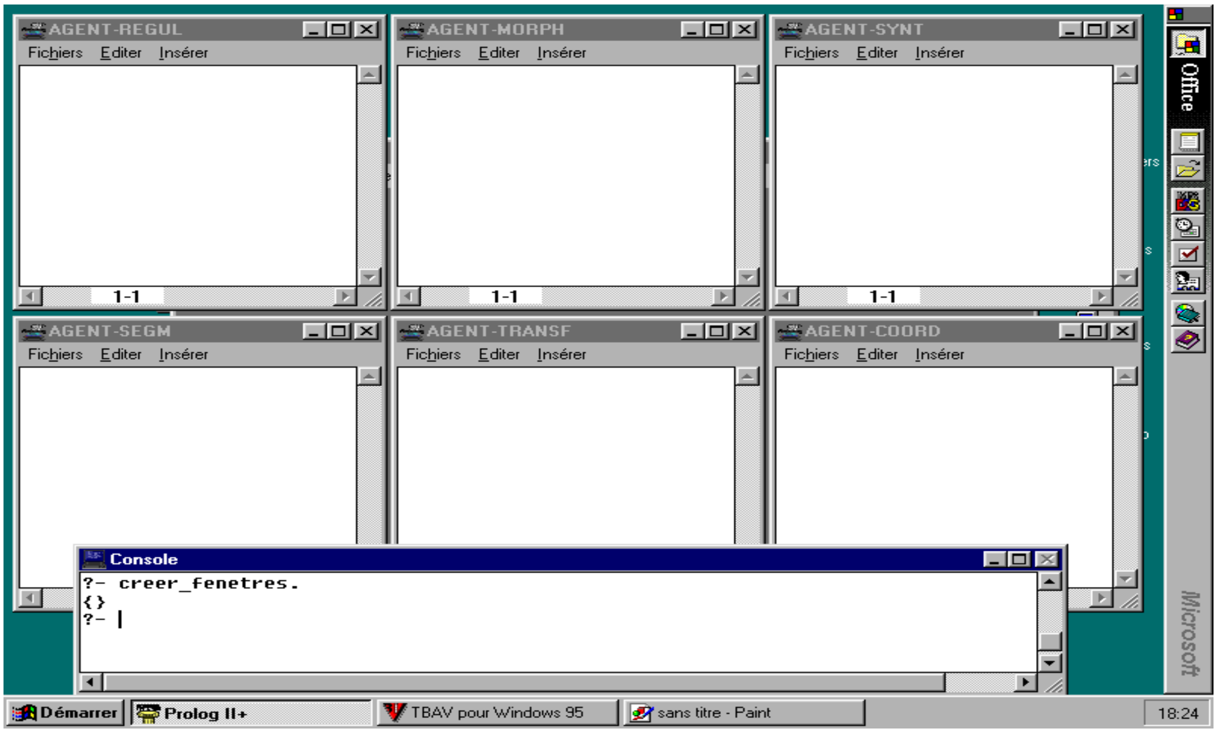
I-2) Système de fenêtrage

Chaque agent est symbolisé par une fenêtre ; ainsi, l'état d'avancement et les résultats de chaque agent sont lisibles dans la fenêtre qui lui est associée.

La création des fenêtre est réalisée par le prédicat `creer_fenetres` :

```
creer_fenetres :-
  new_window("AGENT-REGUL",:edit,1,10,50,250,250),
  new_window("AGENT-MORPH",:edit,1,260,50,500,250),
  new_window("AGENT-SYNT",:edit,1,510,50,750,250),
  new_window("AGENT-SEGM",:edit,1,10,300,250,500),
  new_window("AGENT-TRANSF",:edit,1,260,300,500,500),
  new_window("AGENT-COORD",:edit,1,510,300,750,500),
  set_window("console",1,50,470,700,570).
```

Cette clause produit le résultat suivant :


AGENT-REGUL
AGENT-MORPH
AGENT-SYNT
AGENT-SEGM
AGENT-TRANSF
AGENT-COORD
Fichiers Editer Insérer
1-1
Console
Démarrer
Prolog II+
TBAV pour Windows 95
sans titre - Paint
18:24
Office
Microsoft

```
?- creer_fenetres.
{}
?- 
```

Figure 16 : Système de fenêtrage

Les fenêtres définies sont de type `:edit` ; i.e. elles sont destinées à l'édition de texte (et d'arbres). Ces fenêtres peuvent être re-dimensionnées (réduction, agrandissement, fermeture) ; leur contenu peut être sauvegardé dans un fichier ou imprimé.

II) Description interne du système

Les agents sont découpés en différents fichiers contenant leurs connaissances, compétences et stratégie générale. Nous allons approfondir certains aspects de l'implémentation des agents (Morph, Segm, Synt et Coord). Ensuite, nous verrons comment la communication est mise en œuvre dans le système.

II-1) Les agents

La spécification des agents du système est développée dans [Stéfanini, 93]. Nous rappelons que le prédicat `agent` permet de définir les agents du système ; exemples :

```
agent(morphologie).
agent(segmentation).
agent(syntaxe).
```

A chaque agent est associé un but local ; par exemple :

```
but_local(morphologie, analyse_morphologique).
but_local(segmentation, segmentation_proposition).
```

Un agent dispose de connaissances :

```
connaissances(segmentation,
        [ liste_introducteurs_forts_proposition(_),
          liste_introducteurs_faibles_proposition(_) ]).
```

Pour atteindre son but local, un agent utilise ses compétences ; elles peuvent, par ailleurs être sollicitées par d'autres agents :

```
competences(segmentation,
        [ chercher_introducteur_prop(_,_),
          nombre_propositions(_,_),
          debut_segmentation(_,_) ]).
```

Un agent utilise différentes stratégies pour atteindre son but local. Voici un exemple de stratégie associée à l'agent Synt :

```
strategie(syntaxe,
        [ analyse_constituants(_,_),
          reperer_structure_isl(_,_),
          analyse_fonctionnelle(_,_,_) ]).
```

Chaque agent dispose d'une boîte aux lettres (ou `mail_box`) dans laquelle il enregistre des informations à l'aide de la procédure `enregistrer_memoire_locale` :

```
mail_box(morphologie, _liste_messages).
Enregistrer_memoire_locale(_agent,_information).
```

Pour chaque agent, nous présentons d'une manière générale les types d'entrée et de sortie ; puis, nous développons certaines de ses compétences.

II-1-1) Agent Morph

II-1-1-1) Présentation générale

L'agent Morph tente d'atteindre son **but local** (l'analyse morphologique d'une forme) après réception du message de l'agent Transf ayant effectué les modifications nécessaires au rétablissement de l'ordre canonique de la phrase. L'agent Morph tente donc d'effectuer l'analyse flexionnelle des formes ; il recherche dans son dictionnaire les entrées lexicales correspondantes ; si aucune entrée lexicale ne correspond, l'agent Morph active sa stratégie de traitement de forme inconnue. Une fois que toutes les formes de la phrases ont été étiquetées, l'agent Morph active son module de levée d'ambiguïtés afin d'éliminer un maximum de solutions parasites.

La structure de données de l'**entrée** de Morph est une liste du type :

```
ph_pret([forme("le","le",1,1),forme("propos","propos",2,2),forme("de
vient","devient",3,3),forme("maniche'en","maniche'en",4,4),forme("."
,".",5,5)])
```

En **sortie**, l'agent Morph fournit une liste contenant les catégories possibles (suivies des valeurs des variables associées) pour chaque forme.
Chaque forme est représentée dans le format suivant : une liste d'entrées lexicales suivies de la catégorie associée et des valeurs des variables.

```
forme(
[cat(    Entrée lexicale,
         Catégorie,
         [Valeur des variables associées],
         position initiale, position après régularisations),
cat(Entrée lexicale identique ou différente, Catégorie identique ou
différente ,…])
```

Ainsi, la phrase est représentée de la manière suivante :

```
[forme([cat("le",y,[mas,sng,pe3,i2a,fai]),cat("le",d,[mas,sng,def,fai]
)],1,1),
forme([cat("propos",f,[mas,nbn,nom,for,com])],2,2),
forme([cat("devenir",v,[pe3,ind,pst,for,ord])],3,3),
forme([cat("maniche'en",f,[mas,sng,nan,for,com])],4,4),
forme([cat(".",t,[vp1])],5,5)])
```

L'agent Morph effectue aussi d'autres tâches sur **demande des autres agents** du système : par exemple, il fournit à l'agent Synt la liste des formes ambiguës de catégorie verbe. Ou encore, Morph réactive sur demande son module de levée d'ambiguïtés.

II-1-1-2) Prédicats de désambiguïsation morphologique

Les règles de désambiguïsation, présentées au chapitre 3, sont programmées de la manière suivante :

```
/* regle 1a : y/d + f -> d + f */
regle_morphologique(_mode,
   [ /* liste initiale contenant la suite Y/D et F */
    forme([cat(_forme_canonique1,y,_var1),
           cat(_forme_canonique2,d,_var2)],_pa1,_pp1),
    forme([cat(_forme_canonique3,f,_var3)],_pa2,_pp2)|_l],
   [ /* liste resultat contenant la suite D et F */
    forme([cat(_forme_canonique2,d,_var2)],_pa1,_pp1),
    forme([cat(_forme_canonique3,f,_var3)],_pa2,_pp2)|_l]) :-
/* conditions d'application de la regle = */
      hors(ppr,_var3),/*le nominal n'est pas un participe present */
      _var2 = [_genre1,_nombre1|_],
      _var3 = [_genre2,_nombre2|_],
      accord_genre(_genre1,_genre2), /* verifie l'accord en genre */
      accord_nombre(_nombre1,_nombre2),/*verifie l'accord en nombre*/
      !,
      outml("Regle 1a : y/d + f -> d + f").
```

II-1-2) Agent Synt

II-1-2-1) Présentation générale

L'agent Synt tente d'atteindre son **but local** (l'analyse syntaxique en constituants et l'analyse fonctionnelle de propositions) après l'intervention de l'agent Segm qui découpe une phrase en propositions. L'agent travaille donc sur une suite de formes contenues dans une proposition ; les **entrées** de Synt peuvent se présenter comme suit :

```
/* la phrase est composée d'une seule proposition (la principale) */
prin([forme([cat("l'",d,[grn,sng,def,fai])],1,1),forme([cat("UC",f,[
grn,nbn,nom,prp])],2,2),forme([cat("faire",v,[pe3,ind,pst,for,ord]),
cat("fait",f,[mas,sng,nom,for,com]),cat("faire",f,[mas,sng,adj,ppa,f
or])],3,3),forme([cat("la",y,[fem,sng,pe3,i2a,fai]),cat("la",d,[fem,
sng,def,fai]),cat("la",f,[mas,sng,nom,for,com])],4,4),forme([cat("so
urde",f,[fem,sng,nan,for]),cat("sourdre",v,[pe3,ind,pst,for,ord])],5
,5),forme([cat("oreille",f,[fem,sng,nom,for,com])],6,6),forme([cat("
.",t,[vp1])],7,7)])

/* la phrase est composée de plusieurs propositions */
deb_prin([forme([cat("la",d,[fem,sng,def,fai])],1,1),forme([cat("pie
`ce",f,[fem,sng,nom,for,com])],2,2),forme([cat("avoir",v,[pe3,ind,ps
t,for,aux])],3,3),forme([cat("deux",d,[grn,plu,num])],4,4),forme([ca
t("partie",f,[fem,plu,nom,for,com])],5,5)])

deb_coord([forme([cat("le",d,[mas,sng,def,fai])],7,7),forme([cat("pr
opos",f,[mas,nbn,nom,for,com])],8,8),forme([cat("devenir",v,[pe3,ind
,pst,for,ord])],9,9),forme([cat("maniche'en",f,[mas,sng,nom,for,com]
)],10,10),forme([cat(".",t,[vp1])],11,11)])
```

• Construction des différentes trajectoires possibles

Lorsque la phrase présente des ambiguïtés "persistantes" (i.e. après une première levée d'ambiguïtés par l'agent Morph), toutes les suites possibles d'entrées lexicales (ou trajectoires) sont construites.

L'agent commence par examiner les verbes potentiels ; ils seront tour à tour choisis (de gauche à droite), et les autres possibilités de verbe seront alors éliminées. Après ce choix du verbe[51] de la proposition, toutes les trajectoires possibles sont construites grâce au prédicat `construit_listes_cat_amb`.

```
/* condition d'arret lorsque la proposition est une phrase et se
termine par une ponctuation forte */
construit_listes_cat_amb([forme([cat(_forme,t,[vp1])],_pos1,_pos2)],
[]):-!.

/* condition d'arret lorsque la proposition n'est pas une phrase
mais une proposition ou un syntagme qui ne se termine pas par une
ponctuation forte */
construit_listes_cat_amb([],[]):-!.

/* lorsque l'entrée lexicale de la forme est unique, elle est
ajoutee a la liste resultat et la forme suivante est examinee */
construit_listes_cat_amb(
[forme([cat(_forme,_cat,_var)|[]],_pos1,_pos2)|Q1],
[_cat,_forme,_var|Q2]):-
   construit_listes_cat_amb(Q1,Q2).

/* lorsque la forme releve de plusieurs entrees lexicales :
premier cas : construit la liste avec la premiere entree lexicale */
construit_listes_cat_amb(
[forme([cat(_forme,_cat,_var)|Cat_suiv],_pos1,_pos2)|Q1],
[_cat,_forme,_var|Q2]):-
   Cat_suiv \== [],
   write("\t-> Differents choix possibles pour la forme :"),
```

---

[51] La stratégie de choix du verbe doit pouvoir être améliorée en utilisant des critères linguistiques.

```
    write(_forme),line,
    construit_listes_cat_amb(Q1,Q2).

/* lorsque la forme releve de plusieurs entrees lexicales :
second cas : construit la liste avec l'entree lexicale suivante */
construit_listes_cat_amb(
[forme([cat(_forme,_cat,_var)|Cat_suiv],_pos1,_pos2)|Q1],Q2):-
    Cat_suiv \== [],
    construit_listes_cat_amb([forme(Cat_suiv,_pos1,_pos2)|Q1],Q2).
```

• Ses sorties

L'agent fournit en sortie une (ou plusieurs) représentation de la proposition en *constituants* sous la forme suivante :

```
proposition[[sn[[d["le"],fnom["propos"]]],sv[[v["devenir"],sn[fnom["
maniche'en"]]]]]]
```

L'agent Synt produit aussi, après l'analyse *fonctionnelle*, la structure de la phrase représentant les différents compléments (du C0 au C5) :

```
complements([d["l'"],fnom["UC"]],[d["la"],[fadj["sourde"],fnom["orei
lle"]]],[],[],[],[])
```

Par ailleurs, l'agent Synt peut être sollicité pour analyser une suite de formes (comme un syntagme).

II-2-2-2) Algorithme en profondeur

La programmation des règles de la grammaire en constituants a nécessité la mise en œuvre d'un "Méta-Programme" ; en effet, certaines de ces règles provoquent en Prolog des boucles infinies.

• Problématique des boucles infinies

Afin de permettre la définition des règles de la grammaire syntaxique, il nous faut définir un méta-programme. Ce dernier doit empêcher Prolog II+ de rester dans des boucles infinies ne permettent pas la résolution de problème. Nous avons choisi d'utiliser un méta-interpréteur qui empêche l'arbre de résolution de dépasser une certaine profondeur.

Les boucles infinies peuvent être de différents types (cf. [Sellami, 89]), nous ne nous intéressons, dans le cadre de cette étude, qu'à la *récursivité à gauche*. La récursivité à gauche est un phénomène qui se produit lorsque la première clause de la partie droite de la règle est identique à la tête (ou partie gauche) de la règle.

Exemple de règle de réécriture d'un nominal récursif à gauche :

```
N :- N + SP
```

• Méta-programmation

Nous dirons qu' "un métaprogramme est un programme qui utilise un autre programme (le programme objet) comme donnée." [Sellami, 89]

Un exemple général de méta-programme est un méta-interpréteur de base permettant l'activation des règles définies dans le programme. Le méta-interpréteur le plus simple qui puisse être écrit en Prolog est le suivant :

```
solve(A) :- A.
```

Nous interpréterons ce programme de la manière suivante : la relation "solve(but)" est vraie si "But" est vrai.

Voici un algorithme classique d'un méta-interpréteur de base (conservant la stratégie de résolution de Prolog i.e. en profondeur d'abord et de gauche à droite) [Sterling & Shapiro, 86] :

```
(1) solve([]).
(2) solve([X|Y]) :- solve(X), solve(Y).
(3) solve(X) :- clause(X,Y), solve(Y).
```

La clause (1) est la condition d'arrêt du méta-interpréteur : la liste vide.
La clause (2) définit la fonction de sélection de l'interpréteur qui impose la résolution du littéral le plus à gauche dans la clause. Ainsi pour résoudre la liste [X|Y], il faut résoudre d'abord X puis Y.
La clause (3) représente le cas général de résolution de problème. Pour résoudre le but X, il faut choisir une clause du programme dont la tête s'unifie avec ce but X ; ensuite, il faut résoudre le corps de la clause. Pour faire fonctionner ce méta-interpréteur, il faut utiliser le prédicat clause qui sert à représenter le programme.

```
Exemples :      la règle du programme : R1(X,Y) :- R2(X).
                      devient : clause(R1(X,Y), R2(X)).
                la règle : R1(X,Y) :- R2(X), R3(Y).
                      devient : clause(R1(X,Y), [R2(X), R3(Y)]).
```

• Algorithme en profondeur

Nous avons retenu pour sa simplicité et son adéquation à nos besoins, l'algorithme en profondeur de [Sellami, 89] :

```
(1) profondeur([],Prof).
(2) profondeur([But1|But2],Prof):-
                profondeur(But1,Prof),
                profondeur(But2,Prof).
(3) profondeur(not(But),Prof):-
                not(profondeur(But,Prof)).
(4) profondeur(But,_):-
                primitive(But),But.
(5) profondeur(But,Prof):-
                Prof < 10,
                clauses(But,Corps),
                val(add(Prof,1),Prof1),
                profondeur(Corps,Prof1).
```

La première clause est la condition d'arrêt : la liste vide.
La seconde clause définit le mode de résolution : de gauche à droite.
La troisième clause est utilisée pour rendre "transparent" le prédicat `not` : si ce prédicat se présente, le méta-interpréteur n'a pas besoin de rechercher la primitive correspondante, il utilise cette clause. Cette clause peut être éliminée, auquel cas il faut définir la primitive `not` : on dit alors que la primitive est "masquée".
La clause (4) sert à l'exécution des primitives (ou faits) "masquées" du programme : il faut alors définir une table de primitives contenant toutes les primitives utilisées dans le programme.
Exemples de primitives :

```
primitive(write(X)).
primitive(line).
```

La dernière clause représente le cas général de résolution (avec parcours des clauses) :

- La valeur de Prof (représentant la profondeur de l'arbre de résolution) doit être inférieur à 10 (dans l'algorithme présenté) ; sinon, il y aura un échec.
- Trouver une clause dont la tête s'unifie avec le "But".
- On peut alors incrémenter Prof.

- On cherche à résoudre le Corps de la clause retenue (dont la tête s'unifiait avec le But).

Nous avons implémenté le méta-interpréteur permettant de limiter la profondeur de l'arbre de résolution pour la mise en œuvre de l'analyse syntaxique en constituants. Nous avons utilisé, en général[52], une profondeur égale à 11, ce qui permettait d'obtenir des temps d'exécution assez courts. Mais, pour des propositions plus longues[53], cette profondeur doit être augmentée, la méta-programmation devient alors coûteuse en temps ; une possibilité est d'utiliser une méthode d'évaluation partielle.

II-2-2-3) Prédicats de la grammaire en constituants

• Appel de l'analyseur syntaxique

L'appel à l'analyseur syntaxique est réalisée par le prédicat `anal_prop` qui fait appel au méta-interpréteur `profondeur`.

```
anal_prop(S0):-
    profondeur(prop2(S0,[],Liste_regles,Arbre),0),
    affiche(S0,Liste_regles,Arbre,nb_sol).
          /* la clause "affiche" sert à l'affichage des solutions */
```

• Réécritures des règles de la grammaire

La grammaire de la proposition écrite en Prolog II+ sans le recours à un algorithme de profondeur est composée de règles se présentant de la manière suivante :

```
/* Règle : SN -> SN + SN pour "un reseau ethernet" */
sn(S0,S,Liste_regles,sn[Liste_synt]) :-
/* recherche d'un premier syntagme nominal avec le debut de la suite
des categories S0 */
    sn(S0,S1,Liste_regles1,Liste_synt1),
/*  recherche  d'un  second  syntagme  nominal  avec  la  suite  des
categories S1 (non utilisees par le premier SN) */
    sn(S1,S,Liste_regles2,Liste_synt2),
/*Liste_synt mémorise l'arbre syntaxique en constituants */
    Liste_synt = [Liste_synt1,Liste_synt2],
/*Liste_regles mémorise les règles de la grammaire utilisees */
    Liste_regles = ["Regle 34 : SN -> SN + SN",
                    Liste_regles1,Liste_regles2].
```

L'ensemble des règles de la grammaire ont été réécrites avec le prédicat `clauses`.

```
/* REGLE 34 : SN -> SN + SN */
clauses(sn(S0,S,Liste_regles,sn[Liste_synt]),
    [sn(S0,S1,Liste_regles1,Liste_synt1),
     sn(S1,S,Liste_regles2,Liste_synt2),
            /*Liste_synt mémorise l'arbre syntaxique */
            /*Liste_regles mémorise les règles de la grammaire */
     Liste_synt = [Liste_synt1,Liste_synt2],
     Liste_regles = ["Regle 34 : SN -> SN + SN",
                    Liste_regles1,Liste_regles2]]).
```

Nous avons prototypé les primitives à l'aide du prédicat `primitive` ; seuls les faits sont restés inchangés.

```
    primitive(d(_,_,_,_)).  /* primitive */
    d([d,Forme,Var|S],S,    /* fait */
            "Regle de reconnaissance d'un determinant",
```

---

[52] Nous avons utilisé une profondeur égale à 11 pour les propositions "simples" figurant en annexe.

[53] Les propositions "simples" testées sur TALISMAN II n'excèdent pas une dizaine de formes. Pour des propositions comportant un nombre plus important de formes, la profondeur doit être augmentée.

```
                  d[Forme]).
```

• Affichage arborescent

Utilisation du prédicat prédéfini `draw_tree` ; ce prédicat est utilisé avec la structure de données suivante :

```
draw_tree(proposition((sn((d(le),fnom(propos))),sv((v(devenir),sn(fn
om(maniche'en))))))).
```

II-2-2-4) Prédicats de la grammaire fonctionnelle

• Présentation générale

L'analyse fonctionnelle d'une proposition est réalisée à partir des résultats fournis par l'analyseur syntaxique en constituants ; elle utilise aussi le verbe choisi comme verbe de la proposition. L'analyseur tente d'associer à chaque complément du verbe, un syntagme (nominal ou prépositionnel) de la proposition. En présence d'un verbe à construction multiple, tous les schémas syntaxiques sont testés (dans l'ordre où ils ont été définis).

• Appel de l'analyse fonctionnelle

L'appel de l'analyseur fonctionnel est décomposé en deux procédures principales. Dans un premier temps, l'analyseur recherche les différents schémas possibles (variable Isl) du verbe (variable Verbe) ; cela est réalisé par le prédicat `reperer_structure_isl (Verbe,Isl)`.

Dans un second temps, l'analyseur tente d'associer à chaque complément du schéma du verbe, un syntagme de la phrase : selon le type du complément régi (et sa position), l'analyseur recherche tantôt un syntagme nominal (noté sn dans la structure de la variable Arbre qu'il a reçue en entrée), tantôt un préverbal (noté y) et tantôt un syntagme prépositionnel (noté sp). Cette tâche est réalisée par le prédicat `associer_schema(Isl, Arbre, Verbe, Complements)`.

```
/* condition d'arret lorsque la liste des schemas possibles du verbe
est vide */
associer_schema([],Arbre,Complements,Verbe) :-
   outml(" -> Fin d'etude des schemas syntaxiques du verbe").

associer_schema(Schemas,Arbre,[Complements|Comp],Verbe) :-
/* la liste des schemas du verbe est represente par la variable
Schemas */
   Schemas = [complements(_c0,_c1,_c2,_c3,_c4,_c5)|Autres_schemas],
/* tentative pour associer le premier schema du verbe aux
complements trouves dans la structure en constituants de la variable
Arbre */
   associer([_c0,_c1,_c2,_c3,_c4,_c5],Arbre,Complements,Verbe),
/* etude des autres achemas du verbe */
   associer_schema(Autres_schemas,Arbre,Comp,Verbe).
```

• Exemple de prédicats de la grammaire fonctionnelle

A titre d'exemple, nous présentons un prédicat permettant, entre autres, la reconnaissance d'un C1 (complément d'objet direct) soit préverbal (appartenant à la liste finie "*le*", "*la*" et "*les*"), soit de type syntagme nominal (reconnu par la présence du terme "*sn*").

```
/* c1 = [] donc le verbe ne regit pas de C1 */
associer2(c1,_l,_l,[],[]) :-
    !.
```

```
/* dans le cas ou le verbe regit un C1 : recherche d'un preverbal de
categorie Y pouvant etre "le", "les" ou "la" */
associer2(c1,Arbre_prop,Arbre_suite,_c1,Synt_c1) :-
    Arbre_prop = [Synt1|Synt2],
    (Forme = "le";Forme="les";Forme="la"),
    associe_y(Synt1,Forme), /* c1 de type y */
    Arbre_suite = Synt2,
    Synt_c1 = Forme,
    !.

/* dans le cas ou le verbe regit un C1 : recherche d'un syntagme
nominal introduit par "sn" */
associer2(c1,Arbre_prop,Arbre_suite,_c1,Synt_c1) :-
    Arbre_prop = [Synt1|Synt2],
    associe_sn(Synt1,Synt_c1), /* c1 de type sn */
    Arbre_suite = Synt2,
    !.
```

Le prédicat `associer2` utilise d'autres prédicats permettant la reconnaissance d'un syntagme nominal, d'un syntagme prépositionnel (introduit par une préposition appartenant à une liste finie), ou d'un préverbal (appartenant aussi à une liste finie) :

```
/* reconnaissance d'un syntagme nominal introduit par "sn[Synt]" */
associe_sn(Syntagme,Synt):-
    Syntagme = sn[Synt].

/* reconnaissance d'un syntagme prepositionnel indique par
"sp[Synt_sp]" et introduit par une preposition appartenant a une
liste representee par la variable Forme */
associe_sp(Syntagme,Forme):-
    Syntagme = sp[Synt_sp],
    Synt_sp = [Deb|Fin],
    Deb = p[Forme].

/* reconnaissance d'un preverbal introduit par "y[Forme]" ou Forme
appartient a une liste finie */
associe_y(Preverbal,Forme) :-
    Preverbal = y[Forme].
```

• Remarque : compléments "d'ordre modifié" (en cas de présence d'un préverbal)

La proposition 17 "*La séparation leur interdit la syndication*" montre comment est reconnue une inversion de la suite des compléments : le complément d'objet indirect C2 (représenté par le préverbal "*leur*") est avant le complément d'objet direct C1 (représenté par le syntagme nominal "*la syndication*").

Cette phrase est analysée correctement par le système : le syntagme nominal "*la séparation*" est reconnu comme étant le CO, le préverbal "*leur*" comme étant le C2 et le syntagme nominal "*la syndication*" comme étant le C1.

Le résultat suivant est affiché par le système :

```
** Structure fonctionnelle =
Verbe : "interdire" ([c0(obl,[sn(nnh,[])]), c1(obl,[sn(nhm, [])]),
c2(vd,[sn(hum,["a`"])]),[],[],[]])
C0 : [d[la],fnom[se'paration]]
C1 : [d[la],fnom[syndication]]
C2 : leur
C3 : []
C4 : []
C5 : []
```

Le résultat de l'analyse fonctionnelle est mémorisé par l'agent Synt sous le format suivant (les arguments correspondent aux C0, C1, C2, C3, C4 et C5) :

```
complements(
    [d["la"],fnom["se'paration"]], /* C0 */
    [d["la"],fnom["syndication"]], /* C1 */
    "leur", /* C2 */
    [],/* C3 */
    [],/* C4 */
    [])/* C5 */
```

II-2-2-5) Prédicats de désambiguïsation syntaxique

Ces règles sont utilisées lorsque la phrase présente des ambiguïtés "persistantes" (i.e. lorsque des ambiguïtés n'ont pu être éliminées par la première levée d'ambiguïtés effectuée par l'agent Morph). Toutes les suites possibles d'entrées lexicales (ou trajectoires) sont construites. Ces trajectoires font toutes l'objet de vérifications ; l'objectif est d'en éliminer un maximum en respectant des critères linguistiques.
Le prédicat `verif_accords` permet de vérifier les accords en genre et en nombre ou en personne selon les cas. La clause suivante permet de vérifier que le genre et le nombre d'un déterminant suivi d'un nom ou d'un adjectif sont identiques.

```
verif_accords([d,Forme1,Var1,f,Forme2,Var2|L]):-
    not(accord_genre_nombre(Var1,Var2)),
    !,
    outml(" -> Solution rejetee syntaxiquement (non accord de d + f)"),
    fail.
```

Le prédicat `verif_longue_distance` permet de réaliser des vérifications sur de longues distances. La clause suivante permet de vérifier qu'après un déterminant, il existe au moins un nominal.

```
verif_longue_distance([d,Forme,Var|Q]):-
        /* vérifie qu'apres un d on a un f */
    not(dans(f,Q)),
    !,
    outml(" -> Solution rejetee syntaxiquement (d sans f apres)"),
    fail.
```

Ces exemples de règles de désambiguïsation syntaxique montrent qu'elles sont faciles à développer ; une étude d'un corpus plus large devrait permettre d'en définir bien d'autres.

II-1-3) Agent Segm

L'agent Segm doit accomplir trois tâches principales pour atteindre son **but local** : reconnaître les introducteurs de propositions, calculer le nombre de propositions et segmenter la phrase en propositions (en délimitant les débuts de propositions).
Il utilise en **entrée** la phrase analysée par l'agent Morph ; et il recherche les introducteurs potentiels de proposition. Ensuite, une stratégie doit être mise au point afin de permettre à l'agent de prédire le nombre de propositions en fonction du nombre d'introducteur de proposition et du nombre de verbes conjugués de la phrase.
Nous avons défini la stratégie suivante (exposée au chapitre IV) pour le calcul du nombre de propositions réalisé par le prédicat `nombre_propositions` :

```
/* PREDICAT DE CALCUL DU NOMBRE DE PROPOSITIONS */
nombre_propositions(_liste_intro_prop,_nb_prop) :-
/* recherche de la liste des verbes non ambigus */
    chercher_memoire_locale(segmentation,form_namb_v(_l2)),
/*  calcul  du  nombre  de  propositions  en  fonction  du  nombre
d'introducteurs de propositions */
    nombre_intro_plus1(_liste_intro_prop,_nb_intro),
/* calcul du nombre de propositions en fonction du nombre de verbes */
    compter(_l2,0,_nombre_verbe),
/* prediction du nombre de propositions par comparaison */
    calcul(_nb_intro,_nombre_verbe,_nb_prop).

/* REGLES DE PREDICTION DU NOMBRE DE PROPOSITIONS EN FONCTION DU
NOMBRE D'INTRODUCTEURS DE PROPOSITIONS ET DU NOMBRE DE VERBES
CONJUGUES */
/* le nombre de verbes = nombre d'introducteurs +1 donc le nombre de
propositions est egal au nombre de verbes conjugues */
calcul(_n,_n,_n):-!.

/* aucun verbe non ambigu : le nombre de propositions est suppose
egal au nombre d'introducteurs */
calcul(_nb_intro,0,_nb_intro):-!.

/* le nombre de verbes est different du nombre d'introducteurs, le
nombre de propositions est suppose egal au nombre de verbes */
calcul(_nb_intro,_nombre_verbe,_nombre_verbe).
```

L'agent Segm effectue alors une segmentation de la proposition en fonction des introducteurs potentiels de proposition détectés ; cette segmentation est réalisée par le prédicat `debut_segmentation_prop`. La segmentation réalisée est simple : une forme de catégorie C introduit une proposition coordonnée (notée `prop_coord`), une forme de catégorie Q introduit une proposition subordonnée (notée `prop_sub`).

L'agent Segm fournit en sortie la liste des introducteurs potentiels, le nombre probable de propositions, la segmentation de la phrase en propositions qu'il a réalisée.

II-1-4) Agent Coord

L'agent Coord essaie d'atteindre son **but local** (i.e. la réalisation d'une coordination inter ou intra-proposition) lorsque la présence d'un coordonnant dans la phrase a été détectée. L'agent se base sur les résultats de Segm (notamment le nombre de propositions) pour décider s'il doit traiter une coordination inter ou intra-proposition.

- **Cas d'une coordination inter-proposition**

Lorsque le nombre de propositions prédit par l'agent Segm est supérieur à 1, l'agent Coord propose une coordination entre deux propositions. Il demande alors à l'agent Synt de l'informer de la structure en constituants des deux propositions. L'agent Synt va effectuer l'analyse en constituants et fonctionnelle de ces deux propositions. Puis l'agent Coord se charge de réaliser la coordination de deux propositions ; pour cela, l'agent utilise le prédicat `accroche_phrases` lui permettant de réaliser l'arborescence des propositions coordonnées.

```
accroche_phrases (Coordonnant, Coordonnee, Principale, Phrase):-
 Coordonnant = forme([cat(_forme_canonique,_cat,_var)],_pa,_pp),
/* reconnaissance des deux propositions */
 Principale = proposition[Princ],
 Prop_princ = principale[Princ],
 Coordonnee = proposition[Coor],
 Prop_coord = principale[Coor],
/* juxtaposition des deux propositions */
 Phrase1 = [Prop_princ,Prop_coord],
 string_term(_forme_canonique,C),
/*  ajout  de  l'information  de  la  coordination  entre  les  2
propositions qui forment la phrase */
 Phrase2 = C[Phrase1],
 Phrase3 = c[Phrase2],
 Phrase = phrase[Phrase3].
```

• Cas d'une coordination intra-proposition

Lorsque l'agent Segm n'a reconnu qu'une seule proposition, l'agent Coord tente de réaliser une coordination intra-proposition. L'agent Coord commence par demander à l'agent Synt de l'informer sur l'analyse en constituants, de la proposition qui précède le coordonnant et, du syntagme (l'agent Synt reconnaît le type de syntagme et construit sa structure arborescente) qui le suit.

Le prédicat `accroche_syntagme` est particulièrement important dans la mesure où il décide du rattachement des syntagmes précédant et suivant le coordonnant. La stratégie de rattachement de syntagme présentée ici est la suivante : parcours de la proposition de droite à gauche (pour cela, le prédicat `miroir` a été utilisé) et recherche du même type de syntagme (que celui suivant le coordonnant). Cette stratégie peut être aisément modifiée.

```
/* condition d'echec lorsque toute la proposition a ete parcourue et
qu'aucun syntame du meme type n'a ete trouve */
accroche_syntagme([],[Synt],Coordonant,Prop):-
    !,
    outml("Le syntagme n'a pas ete trouve").

/* examine le syntagme juste avant le coordonnant */
accroche_syntagme(Princ_res,[Synt],Coordonant,Prop):-
 Synt = Type_synt[S],
 Princ_res = [S0[S1]|Suite],
 accroche(S1,[Synt],Coordonant,Synt_res),
 Prop = [S0[Synt_res]|Suite],!.

/*  compare  la  nature  des  deux  syntagmes  et  realise  la  coordination
s'ils sont de meme nature */
accroche(Princ_res,[Synt],C,Synt_res):-
 Synt = Type_synt[S],
 Princ_res = [Type_synt[S1]|Suite],
 concat([Type_synt[S1]],[Synt],Synt_coord),
 Synt_res2 = C[Synt_coord],
 Synt_res3 = c[Synt_res2],
 Synt_res4 = [Synt_res3|Suite],
 Synt_res = Type_synt[Synt_res4],
 !.

/* examine la suite de la proposition pour trouver un autre syntagme
de meme nature que celui qui suit le coordonnant */
accroche([Deb|Suite],[Synt],Coordonant,[Deb|Prop]):-
accroche(Suite,[Synt],Coordonant,Prop).
```

Lorsque l'agent Coord a réussi à coordonner les syntagmes de la proposition, il en informe l'agent Synt qui effectue alors l'analyse fonctionnelle de la proposition.

II-2) La communication avec un contrôle décentralisé

Les agents de TALISMAN II bénéficient d'un contrôle décentralisé : ils sont autonomes, ils ont tous le même poids et les prises de décisions sont locales.
Les envois de messages sont réalisés en respectant les protocoles de communication (définis au chapitre 2). Les agents ont des connaissances sur les autres agents du système (leurs accointances) et connaissent le ou les agents destinataires ; ils sont de plus, capables de déterminer si l'information envoyée est sûre ou si elle ne constitue qu'une hypothèse. Par ailleurs, les agents du système ont des connaissances sur les compétences des autres agents du système.

II-2-1) Le format des messages

Nous avons présenté au chapitre 2, le format des messages dans TALISMAN II :

(Routage, Acte, Contenu)

Ce qui rejoint le format Prolog suivant utilisé par l'agent morphologie pour communiquer aux agent Segm et Coord, la phrase analysée morphologiquement et les verbes non ambigus qu'elle contient :

```
message( morphologie,      /* expediteur */
         [segmentation, syntaxe, coordination], /* destinataires */
         inform, assert,         /* Acte = type + force */
         [ph_lv_amb(_liste_cat2), /* contenu[54] */
         form_namb_v(_non_ambigu)]).
```

II-2-2) Les protocoles de communication

Le chapitre 2 nous a permis de modéliser les protocoles de communication nécessaires à la communication directe entre les agents du système. Ils ont été implémentés par les faits `forces_suivantes` dont :

- le premier argument correspond au type de l'acte de communication utilisé lors de l'envoi du message,
- le second représente la force (ou performative) associée au type de l'acte déterminé ci-dessus,
- le dernier argument spécifie les réponses possibles (en tant qu'une combinaison d'une type d'acte et d'une force).

```
/* Envoi d'une information ne nécessitant pas de réponse */
```

---

[54] Le contenu correspond ici à la phrase après une première levée d'ambiguïtés et à la liste des formes non ambiguës de catégorie verbe.

```
forces_suivantes(inform,assert,assert([])).

/* Envoi d'une demande d'action nécessitant une réponse */
forces_suivantes(request,[],answer([assert,no_opinion])).

/* Envoi d'une demande de coopération sur une proposition de l'agent
nécessitant une réponse */
forces_suivantes(request,propose,answer([modify,confirm,desagree,no_
opinion])).

/* Envoi d'une réponse n'engendrant pas de nouvelle réponse sauf
dans le cas d'une modification (mofify) */
forces_suivantes(answer,assert,assert([])).
forces_suivantes(answer,modify,modify([propose])).
forces_suivantes(answer,confirm,confirm([])).
forces_suivantes(answer,desagree,desagree([])).
forces_suivantes(answer,no_opinion,no_opinion([])).
```

II-2-3) Envoi de message

Comme nous l'avons dit, les agents connaissent les destinataires d'un message. Nous distinguerons trois cas d'envoi de messages.

- **L'agent a atteint son but local**

Un agent qui réussit à atteindre son but local doit informer les autres agents (ses accointances) susceptibles d'être intéressés par ses résultats.
Dans l'exemple suivant, l'agent Transf a réussi à rétablir l'ordre canonique de la phrase ; il en informe les agents Morph et Synt.

```
message( transformation,
         [morphologie,syntaxe],
         inform,assert,
         [transf(_canon,_modalite,_marque)]).
```

Après avoir atteint son but local, l'agent Morph envoie ses résultats aux agents Segm, Synt et Coord.

```
message( morphologie,
         [segmentation, syntaxe, coordination],
         inform, assert,
         [ph_lv_amb(_liste_cat2), form_namb_v(_non_ambigu)]).
```

- **La demande d'action (ou d'information)**

La demande d'action se produit lorsqu'un agent a recours à une compétence d'un autre agent. Dans ce cas, l'agent lui envoie un message de type `request []` ; dans son message l'agent expéditeur indique l'information qu'il cherche (nécessitant parfois un traitement à effectuer).
Dans l'exemple suivant, l'agent Synt demande à l'agent Morph de lui donner la liste des verbes potentiels (i.e. les formes ambiguës comportant la catégorie verbe V) :

```
message( syntaxe,
         [morphologie],
         request,[],
         donner_formes_ambigues_contenant_cat(v,_l1,_ambigu).
```

Dans cet autre exemple (ci-dessous), l'agent Coord demande à l'agent Synt de l'informer des résultats de l'analyse en constituants de deux propositions (ces deux propositions étant deux propositions coordonnées).

```
  message(  coordination,
            [syntaxe]
            request,[],
            [analyse_constituants(Prop1,Arbre1),
            analyse_constituants(Prop2,Arbre2)]).
```

• La demande de coopération

Lorsque l'agent dispose d'une information dont il n'est pas sûr, il envoie un message du type `request propose` à un agent susceptible d'évaluer la solution proposée.

Le message suivant est envoyé par l'agent Morph lorsqu'il a détecté une forme inconnue de son dictionnaire. Il utilise ses stratégies pour émettre des hypothèses quant à l'analyse morphologique possible de cette forme. Il envoie la demande de coopération suivante :

```
  message(  morphologie,
            [syntaxe],
            request, propose,
            [ph_lv_amb(_liste_cat2),form_namb_v(_non_ambigu)]).
```

II-2-4) Réception de message

L'agent Regul a un statut particulier dans la mesure où il est le premier agent à travailler sur la phrase. Mais, d'une manière générale, tous les agents ne deviennent "actifs" (i.e. ils n'activent leur but local ou leur compétence) qu'à la réception de messages.

L'activation des agents du système peut être de différente nature selon le type de message envoyé :

• Réception d'un message de type "inform assert"

Lorsque le message est du type `inform assert` l'agent commence par enregistrer dans sa mémoire locale le contenu de ce message. Ensuite, plusieurs cas peuvent se présenter. Dans certains cas, le contenu de ce message permet à l'agent de tenter d'atteindre son but local. Dans d'autres cas, le contenu du message est utile à l'agent mais il ne peut pas exploiter tout de suite les informations qu'il contient.

• Réception d'un message de type "request []"

Ce type de message indique à l'agent que ses compétences sont sollicitées. L'agent tente alors d'effectuer la tâche qui lui est demandée, s'il ne l'a pas déjà fait.

Si l'agent est capable de donner l'information, il renvoie à l'expéditeur un message du type `answer assert`, contenant les résultats de son traitement.

Ainsi, lorsque l'agent Morph reçoit le message de Synt (qui lui demande la liste des verbes potentiels de la phrase), l'agent Morph s'exécute ; et s'il réussit dans son traitement, l'agent Morph envoie le message suivant de réponse à l'agent Synt :

```
  message(  morphologie,
            [syntaxe],
            answer, assert,
            donner_formes_ambigues_contenant_cat(v,
  /* liste des formes de la phrase */
  [forme([cat("l'",d,[grn,sng,def,fai])],1,1),forme([cat("UC",f,[grn,n
  bn,nom,prp])],2,2),forme([cat("faire",v,[pe3,ind,pst,for,ord]),cat("
```

```
fait",f,[mas,sng,nom,for,com]),cat("faire",f,[mas,sng,adj,ppa,for])]
,3,3),forme([cat("la",y,[fem,sng,pe3,i2a,fai]),cat("la",d,[fem,sng,d
ef,fai]),cat("la",f,[mas,sng,nom,for,com])],4,4),forme([cat("sourde"
,f,[fem,sng,nan,for]),cat("sourdre",v,[pe3,ind,pst,for,ord])],5,5),f
orme([cat("oreille",f,[fem,sng,nom,for,com])],6,6),forme([cat(".",t,
[vp1])],7,7)],

/* liste resultat contenant les verbes potentiels */
[forme([cat("faire",v,[pe3,ind,pst,for,ord]),cat("fait",f,[mas,sng,n
om,for,com]),cat("faire",f,[mas,sng,adj,ppa,for])],3,3),forme([cat("
sourde",f,[fem,sng,nan,for]),cat("sourdre",v,[pe3,ind,pst,for,ord])]
,5,5)]).
```

Si l'agent a échoué et n'a pas pu trouver de solutions à la demande qui lui été faite, il renvoie un message du type `answer no_opinion`.

• Réception d'un message de type "request propose"

A la réception d'un message de ce type, l'agent tente de valider ou d'infirmer les hypothèses émises par l'expéditeur en activant ses compétences.

Lorsque l'agent Morph a détecté la forme inconnue "*disc-jockeys*", il a proposé à Synt l'analyse de cette forme par un nominal (de genre et de nombre neutre). L'agent Synt a réussi à analyser cette phrase (avec cette catégorisation) ; il envoie donc un message de confirmation à l'agent Morph.

```
message( syntaxe,
         [morphologie],
         answer,confirm,
         [ph_lv_amb([forme([cat("les",d,[grn,plu,def,fai])],1,1),
forme([cat("disc-jockeys",f,[grn,nbn,nan])],2,2),
forme([cat("accepter",v,[pe6,ind,imp,for,ord])],3,3),forme([cat("ces
",d,[grn,plu,def,fai])],4,4),forme([cat("cassette",f,[fem,plu,nom,fo
r,com])],5,5),forme([cat(".",t,[vp1])],6,6)]),form_namb_v([forme([ca
t("accepter",v,[pe6,ind,imp,for,ord])],3,3)])]).
```

Si l'agent Synt ne disposait d'aucune compétence permettant d'émettre un avis sur cette hypothèse, il aurait renvoyé un message du type `answer no_opinion`. Par contre, si l'analyse syntaxique de cette phrase avait abouti à un échec de l'analyseur, l'agent Synt aurait répondu à Morph un message du type `answer desagree`. Le dernier type de réponse possible `answer modify` suppose que l'agent dispose de compétences lui permettant de proposer une autre hypothèse.

III) Bilan

Nous présentons un exemple d'exécution du système TALISMAN II, avant d'étudier les résultats produits par le système pour une trentaine de propositions.

III-1) Exécution du système

La phrase que nous allons étudier est la suivante :
"*Est-ce que ce fabusien trop pris travaillait au sénat ou à l'assemblée ?*"
Cette phrase présente les caractéristiques suivantes :

- phrase interrogative du type question totale (notée qto),
- présence d'un coordonnant "*ou*",
- présence d'un mot inconnu (le néologisme "*fabusien*"),
- présence de l'amalgame orthographique "*au*" qui sera transformé en "*à*" + "*le*",
- cette phrase comporte deux verbes potentiels : le premier "*pris*" est ambigu car il relève des catégories nominal ou verbe, le second "*travaillait*" est un verbe non ambigu.
- cette phrase comporte d'autres ambiguïtés ("*ce*" et "*l*'" sont un déterminant ou un préverbal et "*assemblée*" est un nom ou un adjectif) que le système devra gérer.

Le schéma suivant résume les échanges de message indispensables à l'obtention de l'analyse de cette phrase :

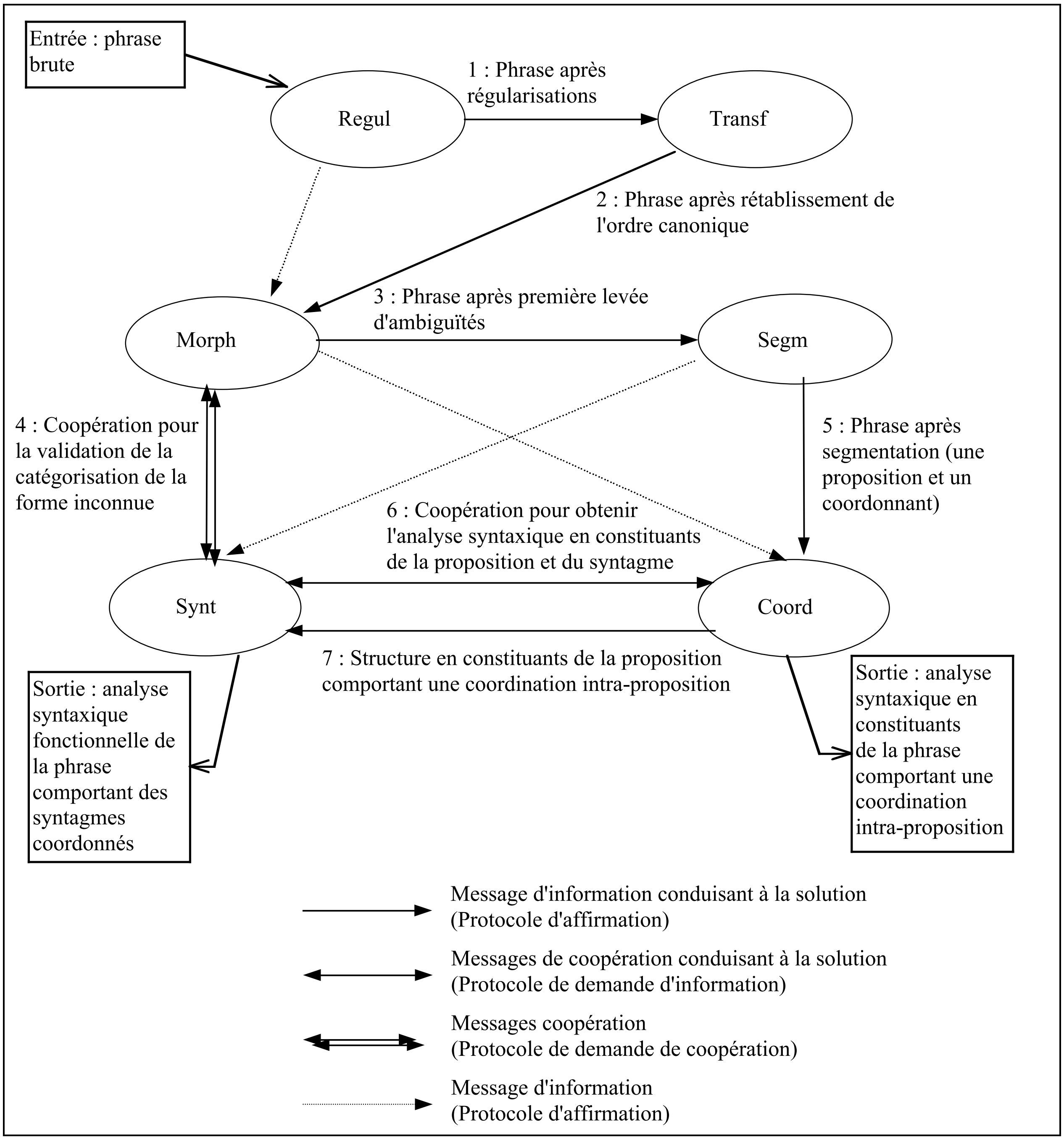
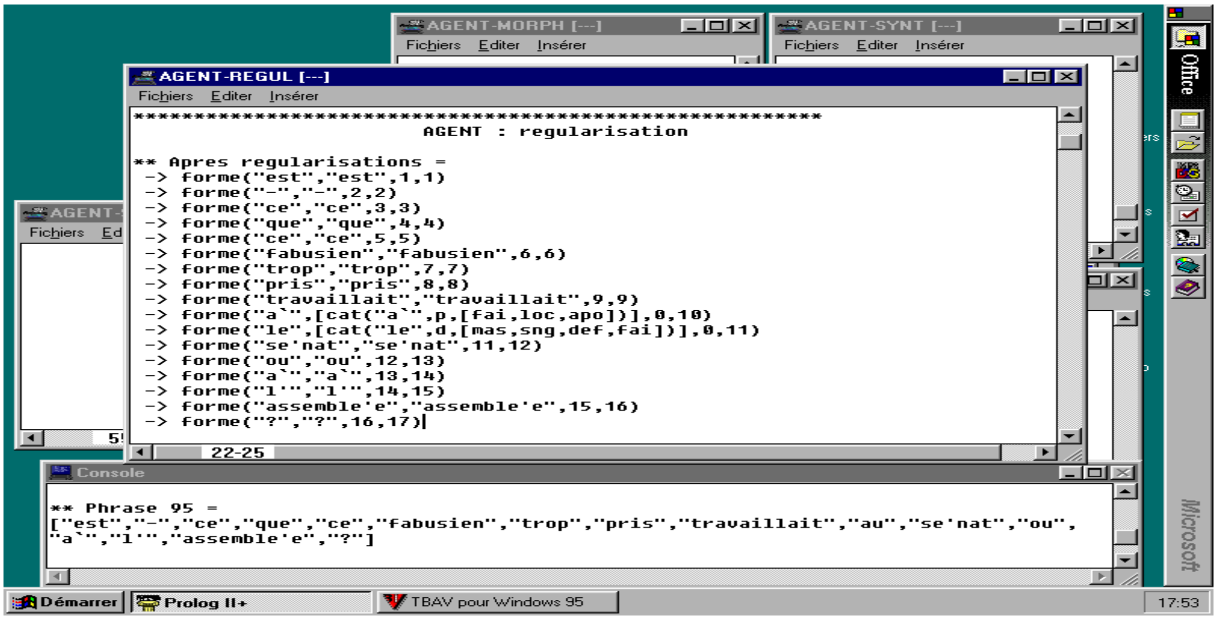

Entrée : phrase brute
Regul
1 : Phrase après régularisations
Transf
2 : Phrase après rétablissement de l'ordre canonique
Morph
3 : Phrase après première levée d'ambiguïtés
Segm
4 : Coopération pour la validation de la catégorisation de la forme inconnue
5 : Phrase après segmentation (une proposition et un coordonnant)
6 : Coopération pour obtenir l'analyse syntaxique en constituants de la proposition et du syntagme
Synt
Coord
7 : Structure en constituants de la proposition comportant une coordination intra-proposition
Sortie : analyse syntaxique fonctionnelle de la phrase comportant des syntagmes coordonnés
Sortie : analyse syntaxique en constituants de la phrase comportant une coordination intra-proposition
Message d'information conduisant à la solution (Protocole d'affirmation)
Messages de coopération conduisant à la solution (Protocole de demande d'information)
Messages coopération (Protocole de demande de coopération)
Message d'information (Protocole d'affirmation)

Figure 17 : Analyse d'une phrase complexe - Architecture

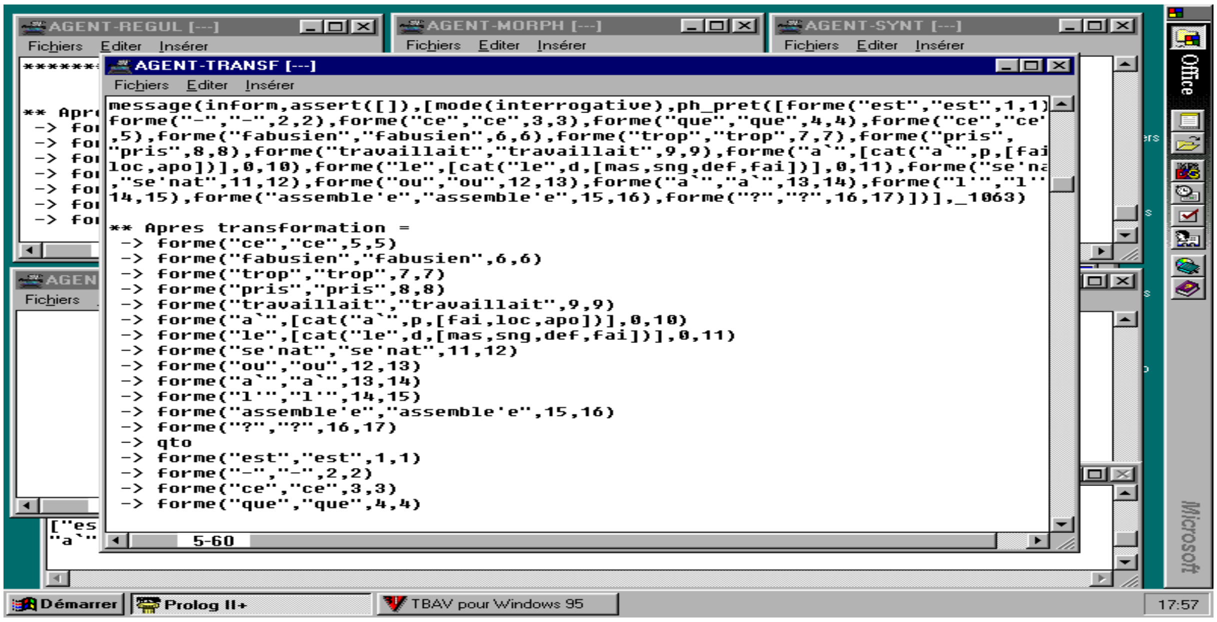

AGENT-MORPH [---]
Fichiers Editer Insérer
AGENT-SYNT [---]
Fichiers Editer Insérer
AGENT-REGUL [---]
Fichiers Editer Insérer
AGENT-
Fichiers Ed
22-25
Console
Démarrer
Prolog II+
TBAV pour Windows 95
17:53
Office
Microsoft

```
*************************************************************
                      AGENT : regularisation

** Apres regularisations =
 -> forme("est","est",1,1)
 -> forme("-","-",2,2)
 -> forme("ce","ce",3,3)
 -> forme("que","que",4,4)
 -> forme("ce","ce",5,5)
 -> forme("fabusien","fabusien",6,6)
 -> forme("trop","trop",7,7)
 -> forme("pris","pris",8,8)
 -> forme("travaillait","travaillait",9,9)
 -> forme("a`",[cat("a`",p,[fai,loc,apo])],0,10)
 -> forme("le",[cat("le",d,[mas,sng,def,fai])],0,11)
 -> forme("se'nat","se'nat",11,12)
 -> forme("ou","ou",12,13)
 -> forme("a`","a`",13,14)
 -> forme("l'","l'",14,15)
 -> forme("assemble'e","assemble'e",15,16)
 -> forme("?","?",16,17)
```

```
** Phrase 95 =
["est","-","ce","que","ce","fabusien","trop","pris","travaillait","au","se'nat","ou",
"a`","l'","assemble'e","?"]
```

Figure 18 : Analyse d'une phrase complexe - Agent Regul

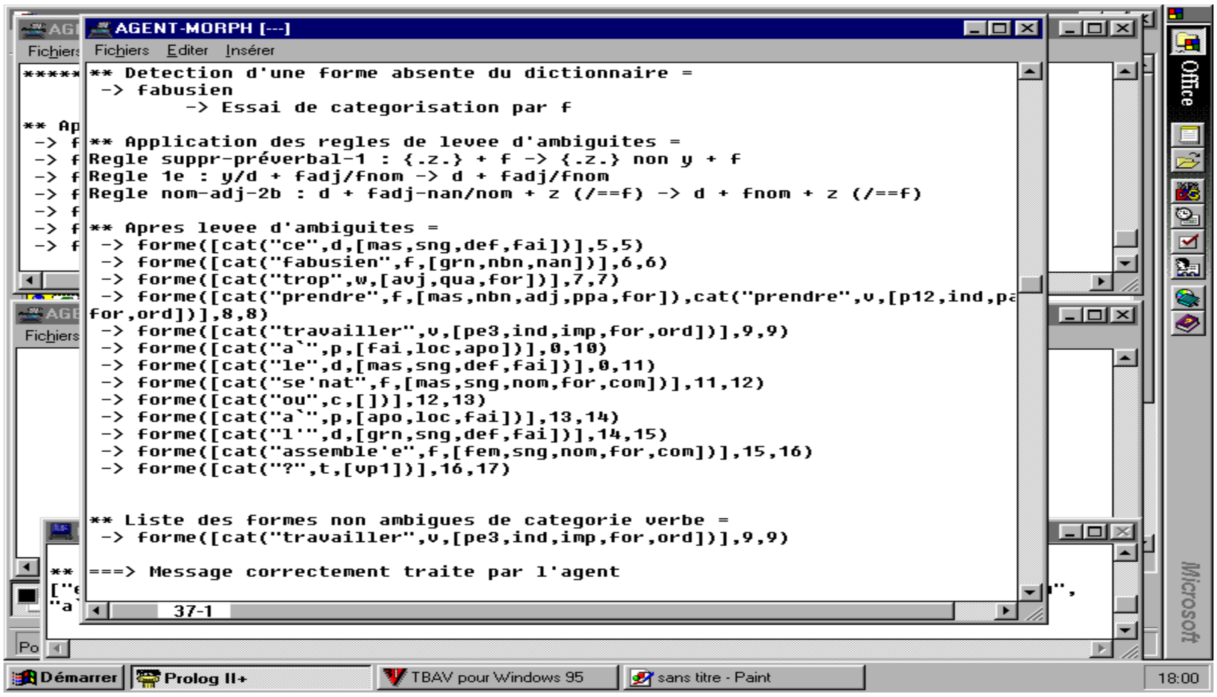
La Figure 18 montre les formes de cette phrase après régularisations : l'amalgame orthographique "*au*" a été transformé en "*a*`" (du type préposition) suivi de "*le*" (catégorisé comme un déterminant).


AGENT-REGUL [---]
Fichiers Editer Insérer
AGENT-MORPH [---]
Fichiers Editer Insérer
AGENT-SYNT [---]
Fichiers Editer Insérer
AGENT-TRANSF [---]
Fichiers Editer Insérer
AGEN
Fichiers
5-60
Démarrer
Prolog II+
TBAV pour Windows 95
17:57
Office
Microsoft

```
message(inform,assert([]),[mode(interrogative),ph_pret([forme("est","est",1,1)
forme("-","-",2,2),forme("ce","ce",3,3),forme("que","que",4,4),forme("ce","ce"
,5),forme("fabusien","fabusien",6,6),forme("trop","trop",7,7),forme("pris",
"pris",8,8),forme("travaillait","travaillait",9,9),forme("a`",[cat("a`",p,[fai
loc,apo])],0,10),forme("le",[cat("le",d,[mas,sng,def,fai])],0,11),forme("se'na
,"se'nat",11,12),forme("ou","ou",12,13),forme("a`","a`",13,14),forme("l'","l'"
14,15),forme("assemble'e","assemble'e",15,16),forme("?","?",16,17)])],_1063)

** Apres transformation =
 -> forme("ce","ce",5,5)
 -> forme("fabusien","fabusien",6,6)
 -> forme("trop","trop",7,7)
 -> forme("pris","pris",8,8)
 -> forme("travaillait","travaillait",9,9)
 -> forme("a`",[cat("a`",p,[fai,loc,apo])],0,10)
 -> forme("le",[cat("le",d,[mas,sng,def,fai])],0,11)
 -> forme("se'nat","se'nat",11,12)
 -> forme("ou","ou",12,13)
 -> forme("a`","a`",13,14)
 -> forme("l'","l'",14,15)
 -> forme("assemble'e","assemble'e",15,16)
 -> forme("?","?",16,17)
 -> qto
 -> forme("est","est",1,1)
 -> forme("-","-",2,2)
 -> forme("ce","ce",3,3)
 -> forme("que","que",4,4)
```

Figure 19 : Analyse d'une phrase complexe - Agent Transf

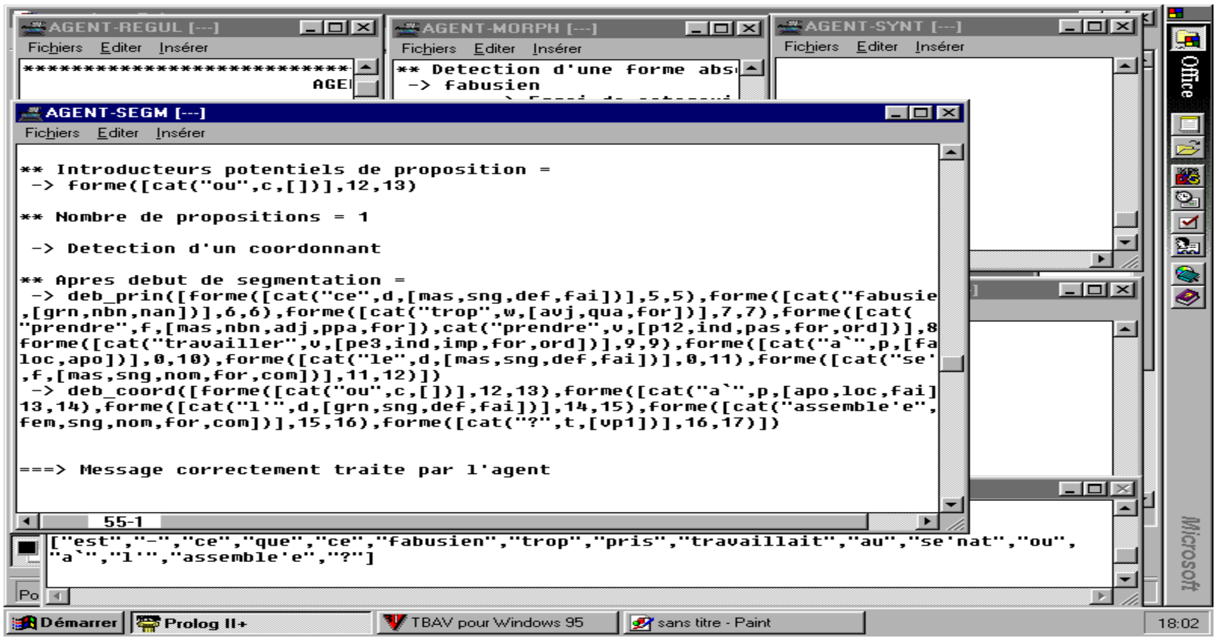
La Figure 19 montre la réception par l'agent Transf du message de l'agent Regul (cf. flèche n°1 de la Figure 17) comportant la phrase après régularisations. L'agent Transf reconnaît une question totale et effectue les traitements adéquats : suppression de "*est-ce que*", reconnaissance du mode interrogatif de la phrase et annotation cette dernière par "qto" (pour question totale).


AGENT-MORPH [---]
Fichiers Editer Insérer
37-1
Démarrer
Prolog II+
TBAV pour Windows 95
sans titre - Paint
18:00
Office
Microsoft

```
** Detection d'une forme absente du dictionnaire =
 -> fabusien
         -> Essai de categorisation par f

** Application des regles de levee d'ambiguites =
Regle suppr-préverbal-1 : {.z.} + f -> {.z.} non y + f
Regle 1e : y/d + fadj/fnom -> d + fadj/fnom
Regle nom-adj-2b : d + fadj-nan/nom + z (/==f) -> d + fnom + z (/==f)

** Apres levee d'ambiguites =
 -> forme([cat("ce",d,[mas,sng,def,fai])],5,5)
 -> forme([cat("fabusien",f,[grn,nbn,nan])],6,6)
 -> forme([cat("trop",w,[avj,qua,for])],7,7)
 -> forme([cat("prendre",f,[mas,nbn,adj,ppa,for]),cat("prendre",v,[p12,ind,pa
for,ord])],8,8)
 -> forme([cat("travailler",v,[pe3,ind,imp,for,ord])],9,9)
 -> forme([cat("a`",p,[fai,loc,apo])],0,10)
 -> forme([cat("le",d,[mas,sng,def,fai])],0,11)
 -> forme([cat("se'nat",f,[mas,sng,nom,for,com])],11,12)
 -> forme([cat("ou",c,[])],12,13)
 -> forme([cat("a`",p,[apo,loc,fai])],13,14)
 -> forme([cat("l'",d,[grn,sng,def,fai])],14,15)
 -> forme([cat("assemble'e",f,[fem,sng,nom,for,com])],15,16)
 -> forme([cat("?",t,[vp1])],16,17)


** Liste des formes non ambigues de categorie verbe =
 -> forme([cat("travailler",v,[pe3,ind,imp,for,ord])],9,9)

===> Message correctement traite par l'agent
```

Figure 20 : Analyse d'une phrase complexe - Agent Morph

L'agent Morph réalise l'analyse flexionnelle des formes de la phrase "transformée" (reçue par le message de Transf : cf. flèche n°2 de la Figure 17) et effectue une première levée d'ambiguïtés (cf. Figure 20). Seule la forme "*pris*" relève encore de deux catégories (nom ou verbe).

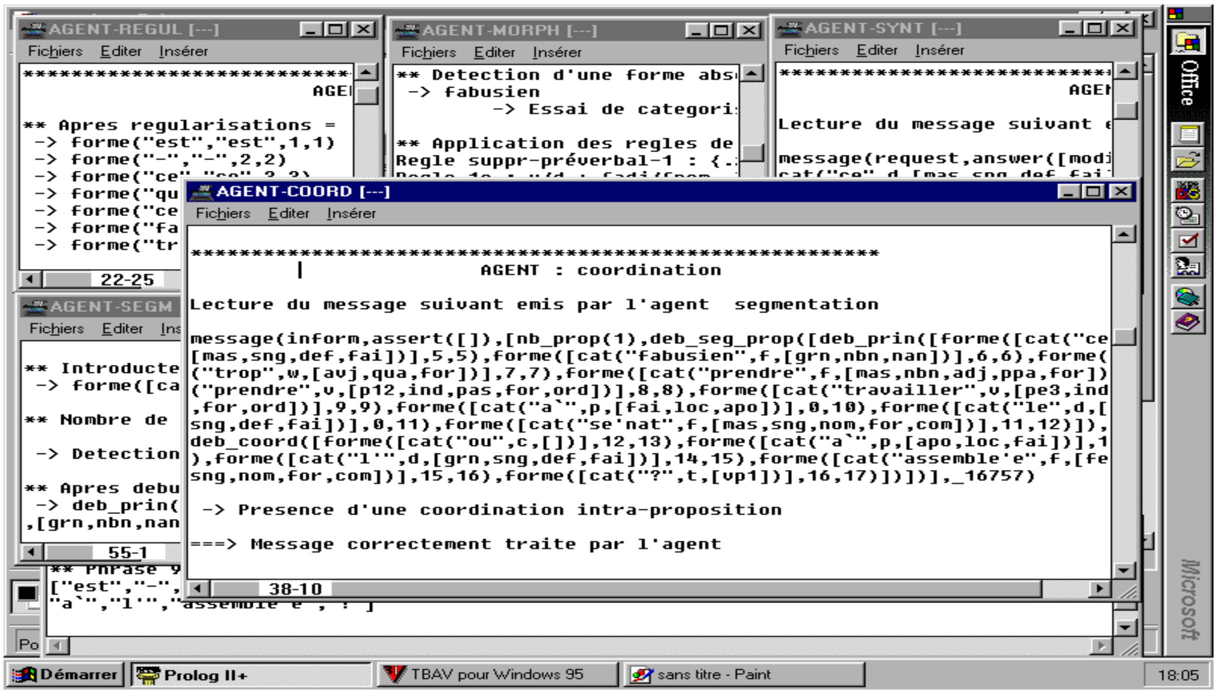

AGENT-REGUL [---]
Fichiers Editer Insérer
AGENT-MORPH [---]
Fichiers Editer Insérer
AGENT-SYNT [---]
Fichiers Editer Insérer
AGENT-SEGM [---]
Fichiers Editer Insérer
55-1
Démarrer
Prolog II+
TBAV pour Windows 95
sans titre - Paint
18:02
Office
Microsoft

```
** Introducteurs potentiels de proposition =
 -> forme([cat("ou",c,[])],12,13)

** Nombre de propositions = 1

 -> Detection d'un coordonnant

** Apres debut de segmentation =
 -> deb_prin([forme([cat("ce",d,[mas,sng,def,fai])],5,5),forme([cat("fabusie
,[grn,nbn,nan])],6,6),forme([cat("trop",w,[avj,qua,for])],7,7),forme([cat(
"prendre",f,[mas,nbn,adj,ppa,for]),cat("prendre",v,[p12,ind,pas,for,ord])],8
forme([cat("travailler",v,[pe3,ind,imp,for,ord])],9,9),forme([cat("a`",p,[fa
loc,apo])],0,10),forme([cat("le",d,[mas,sng,def,fai])],0,11),forme([cat("se'
,f,[mas,sng,nom,for,com])],11,12)])
 -> deb_coord([forme([cat("ou",c,[])],12,13),forme([cat("a`",p,[apo,loc,fai]
13,14),forme([cat("l'",d,[grn,sng,def,fai])],14,15),forme([cat("assemble'e",
fem,sng,nom,for,com])],15,16),forme([cat("?",t,[vp1])],16,17)])


===> Message correctement traite par l'agent
```

```
["est","-","ce","que","ce","fabusien","trop","pris","travaillait","au","se'nat","ou",
"a`","l'","assemble'e","?"]
```

Figure 21 : Analyse d'une phrase complexe - Agent Segm

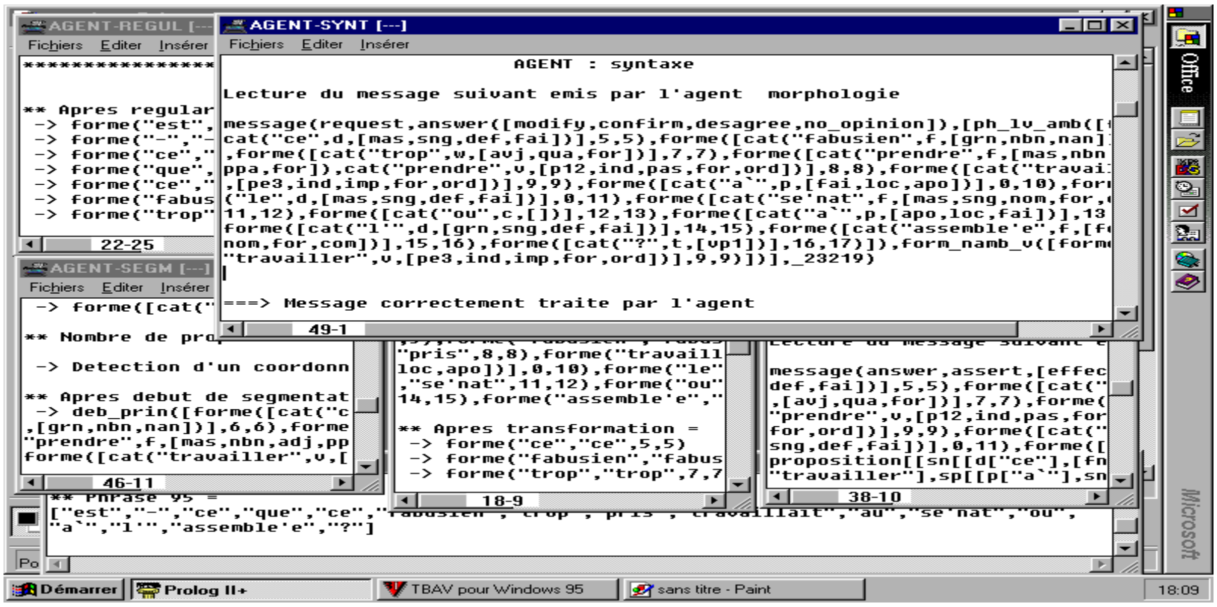
L'agent Segm (après réception du message de Morph, cf. flèche n°3 de la Figure 17) tente alors d'atteindre son but local : prédire le nombre de propositions et proposer une première segmentation de la phrase (cf. Figure 21).



```
AGENT-COORD [---]
*************************************************************
                        AGENT : coordination

Lecture du message suivant emis par l'agent  segmentation

message(inform,assert([]),[nb_prop(1),deb_seg_prop([deb_prin([forme([cat("ce
[mas,sng,def,fai])],5,5),forme([cat("fabusien",f,[grn,nbn,nan])],6,6),forme(
("trop",w,[avj,qua,for])],7,7),forme([cat("prendre",f,[mas,nbn,adj,ppa,for])
("prendre",v,[p12,ind,pas,for,ord])],8,8),forme([cat("travailler",v,[pe3,ind
,for,ord])],9,9),forme([cat("a`",p,[fai,loc,apo])],0,10),forme([cat("le",d,[
sng,def,fai])],0,11),forme([cat("se'nat",f,[mas,sng,nom,for,com])],11,12)]),
deb_coord([forme([cat("ou",c,[])],12,13),forme([cat("a`",p,[apo,loc,fai])],1
),forme([cat("l'",d,[grn,sng,def,fai])],14,15),forme([cat("assemble'e",f,[fe
sng,nom,for,com])],15,16),forme([cat("?",t,[vp1])],16,17)])])],_16757)

 -> Presence d'une coordination intra-proposition

===> Message correctement traite par l'agent
```

Figure 22 : Analyse d'une phrase complexe - Agent Coord

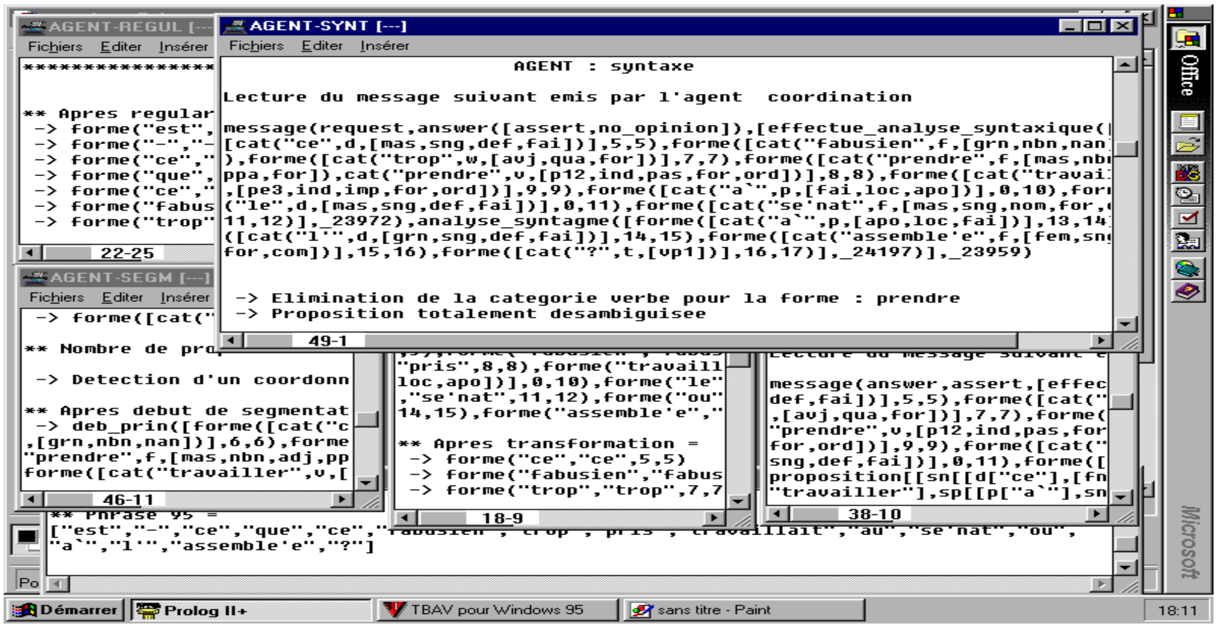
L'agent Segm informe l'agent Coord de ses résultats (cf. flèche n°5 de la Figure 17). L'agent Coord conclut qu'il se trouve en présence d'une coordination intra-proposition (cf. Figure 22).



```
AGENT-SYNT [---]
                        AGENT : syntaxe

Lecture du message suivant emis par l'agent  morphologie

message(request,answer([modify,confirm,desagree,no_opinion]),[ph_lv_amb([
cat("ce",d,[mas,sng,def,fai])],5,5),forme([cat("fabusien",f,[grn,nbn,nan]
,forme([cat("trop",w,[avj,qua,for])],7,7),forme([cat("prendre",f,[mas,nbn
ppa,for]),cat("prendre",v,[p12,ind,pas,for,ord])],8,8),forme([cat("travai
,[pe3,ind,imp,for,ord])],9,9),forme([cat("a`",p,[fai,loc,apo])],0,10),for
("le",d,[mas,sng,def,fai])],0,11),forme([cat("se'nat",f,[mas,sng,nom,for,
11,12),forme([cat("ou",c,[])],12,13),forme([cat("a`",p,[apo,loc,fai])],13
forme([cat("l'",d,[grn,sng,def,fai])],14,15),forme([cat("assemble'e",f,[f
nom,for,com])],15,16),forme([cat("?",t,[vp1])],16,17)]),form_namb_v([form
"travailler",v,[pe3,ind,imp,for,ord])],9,9)])],_23219)


===> Message correctement traite par l'agent
```

Figure 23 : Analyse d'une phrase complexe - Agent Synt

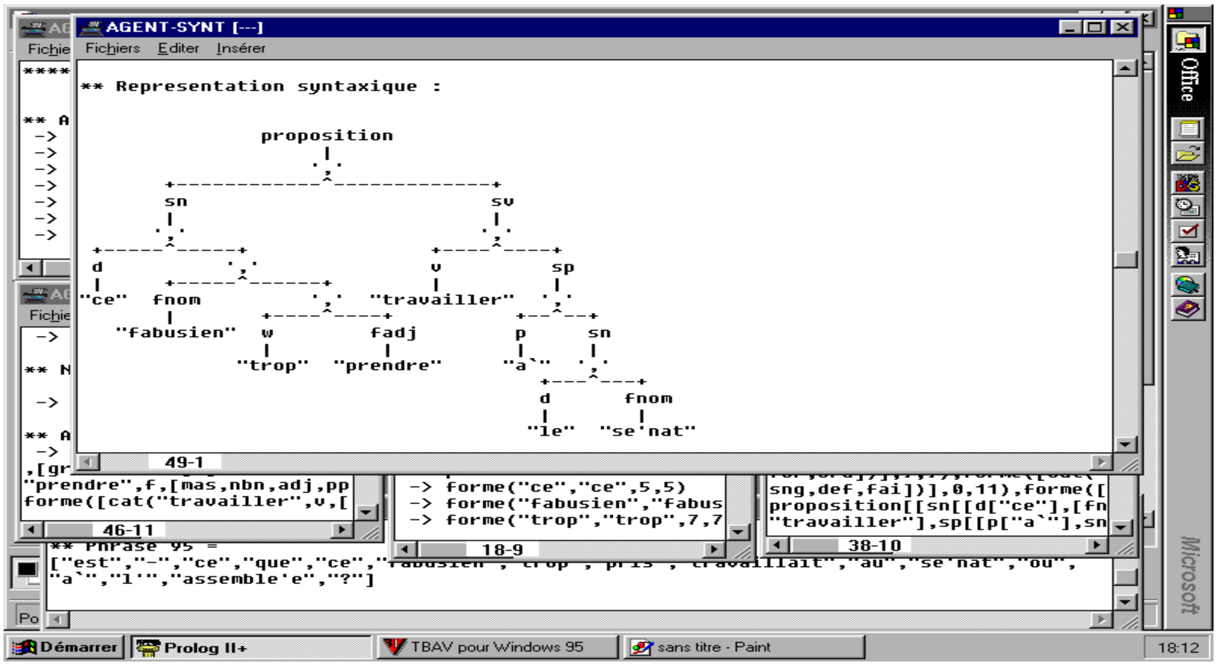
La Figure 23 montre la réception du message de demande de coopération (cf. flèche n°4 de la Figure 17) de l'agent Morph (à cause de la présence d'une forme inconnue du dictionnaire). L'agent Synt devra répondre par un message du type "answer (modify, confirm, desagree ou no_opinion) en fonction des résultats de l'analyse syntaxique qu'il effectuera.


AGENT-REGUL [---]
AGENT-SYNT [---]
AGENT-SEGM [---]
Fichiers Editer Insérer
Démarrer
Prolog II+
TBAV pour Windows 95
sans titre - Paint
18:11
Office
Microsoft

```
                         AGENT : syntaxe

Lecture du message suivant emis par l'agent  coordination

message(request,answer([assert,no_opinion]),[effectue_analyse_syntaxique(
[cat("ce",d,[mas,sng,def,fai])],5,5),forme([cat("fabusien",f,[grn,nbn,nan
),forme([cat("trop",w,[avj,qua,for])],7,7),forme([cat("prendre",f,[mas,nb
ppa,for]),cat("prendre",v,[p12,ind,pas,for,ord])],8,8),forme([cat("travai
,[pe3,ind,imp,for,ord])],9,9),forme([cat("a`",p,[fai,loc,apo])],0,10),for
("le",d,[mas,sng,def,fai])],0,11),forme([cat("se'nat",f,[mas,sng,nom,for,
11,12)],_23972),analyse_syntagme([forme([cat("a`",p,[apo,loc,fai])],13,14
([cat("l'",d,[grn,sng,def,fai])],14,15),forme([cat("assemble'e",f,[fem,sn
for,com])],15,16),forme([cat("?",t,[vp1])],16,17)],_24197)],_23959)


 -> Elimination de la categorie verbe pour la forme : prendre
 -> Proposition totalement desambiguisee
```

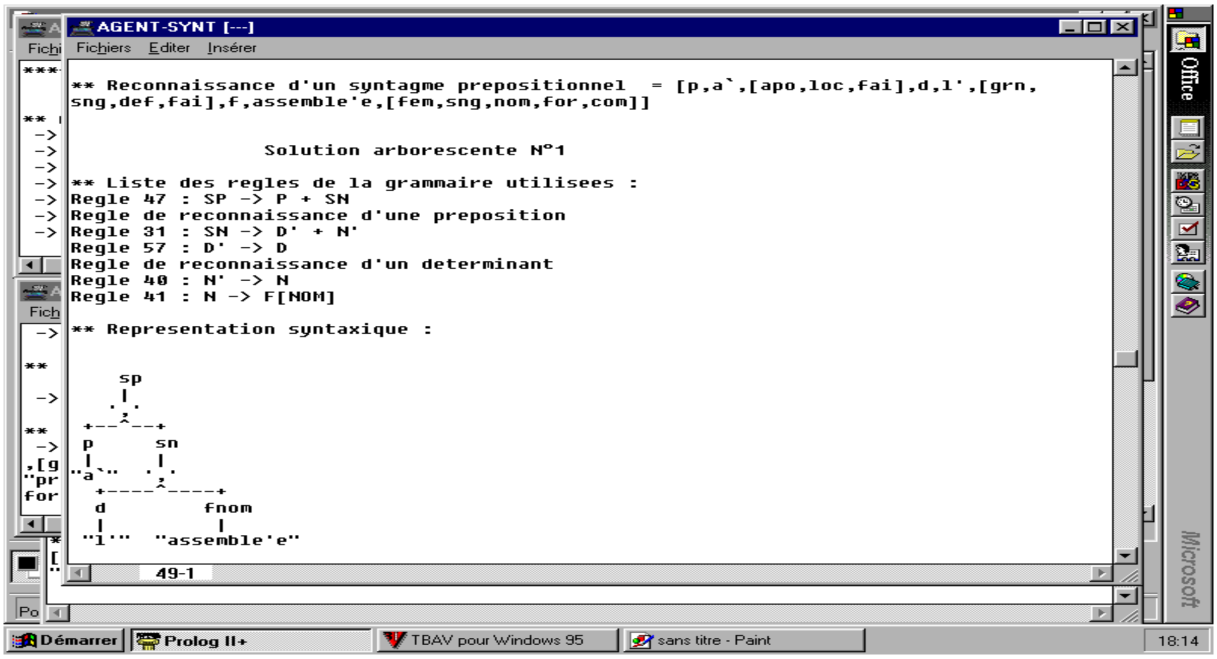
Figure 24 : Analyse d'une phrase complexe - Agent Synt

L'agent Synt est sollicité par l'agent Coord (cf. flèche n°6 de la Figure 17) pour l'informer de l'analyse en constituants du début de la proposition et d'un syntagme. La Figure 24 montre que l'agent Synt choisi comme verbe le verbe non ambigu "*travailler*" et élimine les autres verbes potentiels ; la proposition est alors totalement désambiguïsée. L'agent Synt tente alors de retrouver l'analyse en constituants de cette proposition. Les résultats sont visibles Figure 25.


AGENT-SYNT [---]
Fichiers Editer Insérer
Démarrer
Prolog II+
TBAV pour Windows 95
sans titre - Paint
18:12
Office
Microsoft

```
** Representation syntaxique :

               proposition
                    |
                   .,.
       +------------^-------------+
       sn                         sv
       |                          |
      .,.                        .,.
 +-----^-----+               +----^----+
 d          ','              v         sp
 |     +-----^------+        |         |
"ce"  fnom         ','  "travailler"  ','
       |       +----^----+          +--^--+
   "fabusien"  w        fadj        p     sn
               |         |          |     |
             "trop"  "prendre"     "a`"  ','
                                      +---^---+
                                      d      fnom
                                      |       |
                                     "le"  "se'nat"
```

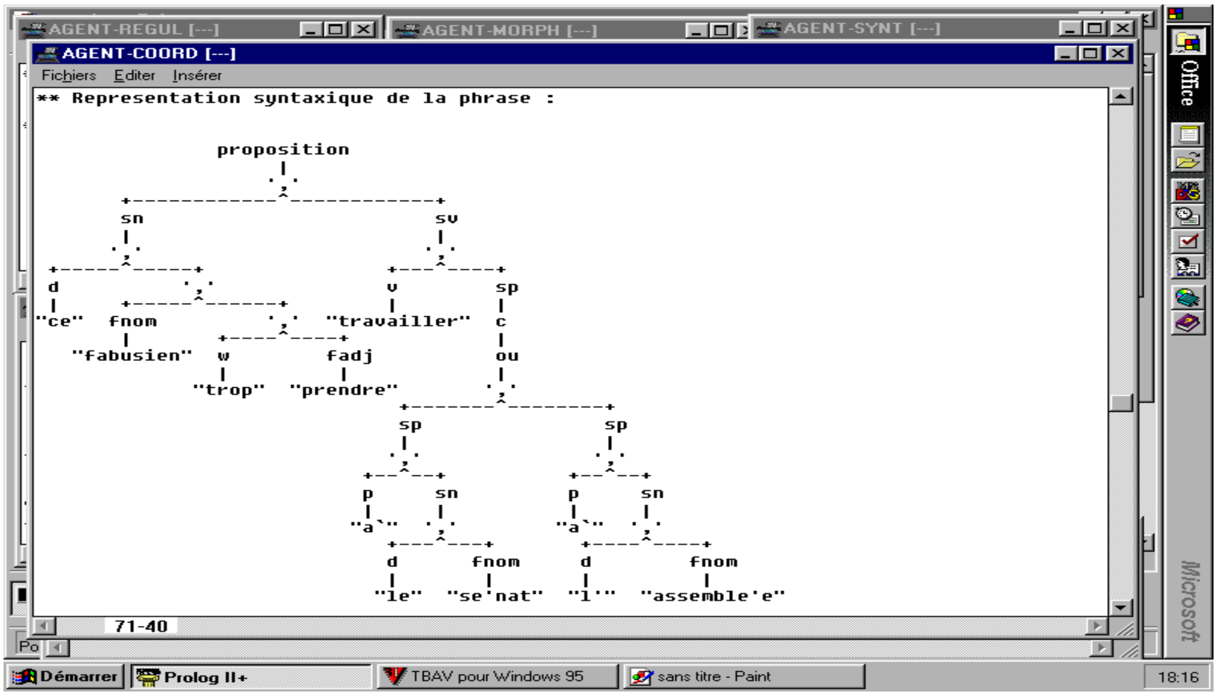
Figure 25 : Analyse d'une phrase complexe - Agent Synt


AGENT-SYNT [---]
Fichiers Editer Insérer
49-1
Démarrer
Prolog II+
TBAV pour Windows 95
sans titre - Paint
18:14
Office
Microsoft

```
** Reconnaissance d'un syntagme prepositionnel  = [p,a`,[apo,loc,fai],d,l',[grn,
sng,def,fai],f,assemble'e,[fem,sng,nom,for,com]]

                Solution arborescente N°1

** Liste des regles de la grammaire utilisees :
Regle 47 : SP -> P + SN
Regle de reconnaissance d'une preposition
Regle 31 : SN -> D' + N'
Regle 57 : D' -> D
Regle de reconnaissance d'un determinant
Regle 40 : N' -> N
Regle 41 : N -> F[NOM]

** Representation syntaxique :


    sp
    |
   . .
    ,
 +--^--+
 p     sn
 |     |
"a`"  . .
       ,
 +-----^-----+
  d        fnom
  |         |
 "l'"  "assemble'e"
```

Figure 26 : Analyse d'une phrase complexe - Agent Synt

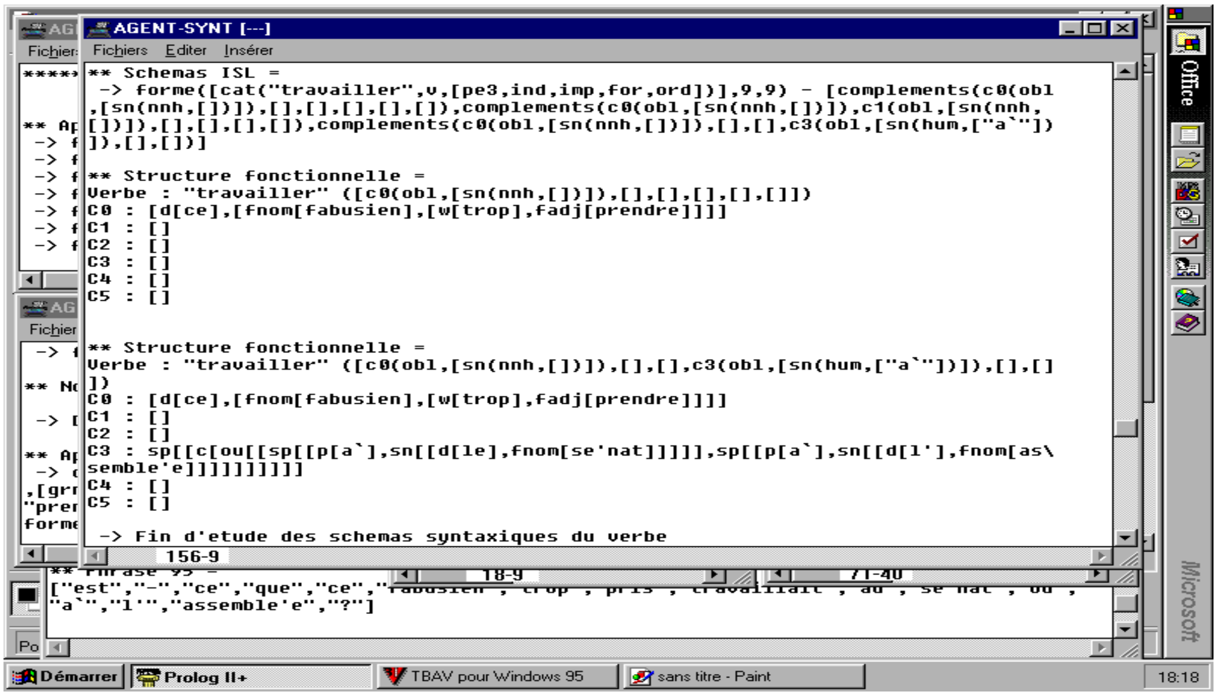
La Figure 26 montre l'analyse du syntagme par l'agent Synt : il reconnaît un syntagme prépositionnel et effectue son analyse en constituants.


AGENT-REGUL [---]
AGENT-MORPH [---]
AGENT-SYNT [---]
AGENT-COORD [---]
Fichiers Editer Insérer
71-40
Démarrer
Prolog II+
TBAV pour Windows 95
sans titre - Paint
18:16
Office
Microsoft

```
** Representation syntaxique de la phrase :


               proposition
                    |
                   . .
                    ,
       +------------^------------+
       sn                        sv
       |                         |
      . .                       . .
       ,                         ,
 +-----^-----+               +---^----+
 d          ','              v        sp
 |     +-----^------+        |        |
"ce"  fnom         ','  "travailler"  c
       |       +----^----+            |
   "fabusien"  w        fadj          ou
               |         |            |
             "trop"  "prendre"       ','
                              +-------^--------+
                              sp               sp
                              |                |
                             . .              . .
                              ,                ,
                           +--^--+          +--^--+
                           p     sn         p     sn
                           |     |          |     |
                          "a`"  . .        "a`"  . .
                                 ,                ,
                             +---^---+       +----^----+
                             d      fnom     d        fnom
                             |       |       |         |
                            "le"  "se'nat"  "l'"  "assemble'e"
```

Figure 27 : Analyse d'une phrase complexe - Agent Coord

L'agent Coord reçoit les résultats de l'agent Synt (cf. flèche n°6 de la Figure 17). Il réalise alors la coordination des deux syntagmes prépositionnels (cf. Figure 27).


AGENT-SYNT [---]
Fichiers Editer Insérer
156-9
18-9
71-40
Démarrer
Prolog II+
TBAV pour Windows 95
sans titre - Paint
18:18
Office
Microsoft

```
** Schemas ISL =
 -> forme([cat("travailler",v,[pe3,ind,imp,for,ord])],9,9) - [complements(c0(obl
,[sn(nnh,[])]),[],[],[],[],[]),complements(c0(obl,[sn(nnh,[])]),c1(obl,[sn(nnh,
[])]),[],[],[],[]),complements(c0(obl,[sn(nnh,[])]),[],[],c3(obl,[sn(hum,["a`"])
]),[],[])]

** Structure fonctionnelle =
Verbe : "travailler" ([c0(obl,[sn(nnh,[])]),[],[],[],[],[]])
C0 : [d[ce],[fnom[fabusien],[w[trop],fadj[prendre]]]]
C1 : []
C2 : []
C3 : []
C4 : []
C5 : []


** Structure fonctionnelle =
Verbe : "travailler" ([c0(obl,[sn(nnh,[])]),[],[],c3(obl,[sn(hum,["a`"])]),[],[]
])
C0 : [d[ce],[fnom[fabusien],[w[trop],fadj[prendre]]]]
C1 : []
C2 : []
C3 : sp[[c[ou[[sp[[p[a`],sn[[d[le],fnom[se'nat]]]]],sp[[p[a`],sn[[d[l'],fnom[as\
semble'e]]]]]]]]]
C4 : []
C5 : []

 -> Fin d'etude des schemas syntaxiques du verbe
```

Figure 28 : Analyse d'une phrase complexe - Agent Synt

L'agent Coord informe l'agent Synt de l'analyse en constituants de cette proposition comportant une coordination intra-proposition (cf. flèche n°7 de la Figure 17). L'agent Synt réalise alors l'analyse fonctionnelle de cette phrase (cf. Figure 28).

Figure 29 : Analyse d'une phrase complexe - Agent Morph

L'agent Synt qui a réussi à analyser syntaxiquement cette phrase, envoie un message de confirmation à l'agent Morph (cf. flèche n°4 de la Figure 17) : sa prédiction sur la catégorie de la forme inconnue doit être correcte.

III-2) Etude des résultats

Nous présentons les résultats de l'analyse morpho-syntaxique de 26 propositions issues de notre corpus ; ces propositions sont des propositions assertives "simples" (sans coordonnants, mots inconnus du dictionnaire,…) présentées en annexe.

Numéro propositions	Cas des verbes	Nouvelle levée d'ambiguïtés	Nombre de trajectoires[55]	Nombre d'arbres syntaxiques en constituants	Nombre d'analyses fonctionnelles
1	1 seul verbe non ambigu	non	1	4	4
2	1 verbe non ambigu 1 verbe ambigu	1 verbe ambigu éliminé par Synt	1	13	6
3	1 seul verbe non ambigu	non	1	1	1
4	1 verbe non ambigu 2 verbes ambigus	2 verbes ambigus éliminés par Synt	8 dont 2 trajectoires valides[56] et 6 éliminés par règles de désambiguïsation de Synt	6	8
5	1 seul verbe non ambigu	non	1	1	1
6	1 seul verbe non ambigu	non	2 dont 1 trajectoire valide et 1 éliminé par règles de désambiguïsation de Synt	4	1
7	1 seul verbe non ambigu	non	1	1	1
8	2 verbes ambigus	Nouvelles désambiguïsations par Morph (1 puis 2)	2 dont 1 trajectoire valide et 1 échec analyseur en constituants de Synt	3	6
9	1 seul verbe non ambigu	non	1	1	1
10	1 seul verbe non ambigu	non	1	8	16
11	1 seul verbe non ambigu	non	1	1	1
12	1 seul verbe non ambigu	non	1	3	2
13	1 seul verbe non ambigu	non	1	2	2
14	1 seul verbe non ambigu	non	1	1	1
15	1 seul verbe non ambigu	non	1	2	2
16	1 seul verbe non ambigu	non	1	3	2
17	1 seul verbe ambigu	Autres catégories pour le verbe éliminées par Synt	1	1	2
18	1 verbe non ambigu 1 verbe ambigu	1 verbe ambigu éliminé par Synt	1	4	4
19	1 seul verbe non ambigu	non	1	4	6
20	1 seul verbe non ambigu	non	6 dont 5 trajectoires valides et 1 échec analyseur en constituants de Synt	28	14
21	1 seul verbe non ambigu	non	1	2	2
22	1 seul verbe non ambigu	non	1	24	34
23	2 verbes ambigus	Nouvelles désambiguïsations par Morph (1 puis 0)	7 dont 2 trajectoires valides et 5 éliminés par règles de désambiguïsation de Synt	9	9
24	1 seul verbe ambigu	Autres catégories pour le verbe éliminées par Synt	1	2	2

---

[55] Lorsqu'une forme correspond à plusieurs entrées lexicales, elles sont toutes testées. Une trajectoire est une suite possible d'entrées lexicales correspondant à chaque forme présente de la phrase.

[56] Nous définissons une " trajectoire valide" comme une trajectoire (non éliminée par les règles syntaxiques) dont l'analyse syntaxique en constituants a abouti.

Numéro propositions	Cas des verbes	Nouvelle levée d'ambiguïtés	Nombre de trajectoires[55]	Nombre d'arbres syntaxiques en constituants	Nombre d'analyses fonctionnelles
25	2 verbes ambigus	Nouvelles désambiguïsations par Morph (0 puis 1)	4 dont 2 trajectoires valides et 1 éliminé par règles de désambiguïsation et 1 échec analyseur en constituants de Synt	18	12
26	1 seul verbe non ambigu	non	2 trajectoires valides	10	8

Tableau 11 : Résultats de l'analyse syntaxique de propositions "simples"

Quelques remarques :

Parmi les 26 propositions, 23 ne posent pas de problèmes quant au *choix du verbe* :

- 18 ne contiennent qu'un seul verbe non ambigu,
- 2 propositions ne contiennent qu'un seul verbe ambigu,
- 3 propositions comprennent un verbe non ambigu et d'autres verbes ambigus (qui sont éliminés),

Les 3 autres propositions contiennent deux verbes ambigus ; ces deux verbes potentiels sont testés ce qui complique la suite de l'analyse et augmente le nombre de solutions fournies par le système.

En ce qui concerne le *nombre de trajectoires* possibles donnant lieu à une analyse syntaxique (i.e. après nouvelle désambiguïsation par Morph et après élimination par les règles syntaxiques), les résultats sont les suivants :

- pour 21 propositions, une seule trajectoire est analysée syntaxiquement,
- 4 propositions engendrent deux trajectoires possibles dont l'analyse syntaxique aboutit,
- seule une proposition (comportant de nombreuses solutions multiples) engendre 5 trajectoires possibles fournissant des solutions syntaxiques différentes.

L'analyse des résultats fournis montre que nous n'en sommes qu'à une première étape dans l'élaboration de stratégies d'élimination de solutions parasites syntaxiques : parmi ces 26 propositions (ne comportant pas d'ambiguïtés effectives), seulement 7 propositions engendrent une analyse fonctionnelle unique.

Dans ce chapitre, nous avons montré les principales stratégies d'analyse des agents que nous avons implantées dans TALISMAN II, ainsi que les moyens de communication dont disposent les agents. Dans le chapitre suivant, nous évaluons nos travaux de recherche en proposant diverses perspectives.

CHAPITRE VI : PERSPECTIVES

L'objectif de ce chapitre est de mettre en relation nos travaux de recherche avec d'autres domaines de recherche. Dans un premier temps, nous validons l'intérêt de notre travail par son application à l'indexation automatique. Ensuite, nous exposons plusieurs perspectives de recherche auxquelles nous nous sommes intéressé, afin de montrer la flexibilité de notre système et son utilité à d'autres aspects du traitement automatique des langues. Enfin, nous faisons le bilan de notre travail : nous tentons de dégager ses apports et ses limites.

I) Application à l'indexation automatique

L'indexation est une opération qui consiste à analyser le contenu d'un document afin de dégager une liste de termes (nommés "descripteurs") permettant de caractériser et de représenter les notions importantes contenues dans le document. L'indexation peut concerner certaines parties du texte comme le titre ou le résumé, ou elle peut porter sur le texte dans son intégralité.

L'indexation peut être une opération intellectuelle (ou manuelle), auquel cas elle est coûteuse en temps et en personnel qualifié ; de plus, le résultat (i.e. l'index) est alors dépendant des connaissances de l'indexeur sur le domaine dont relève le document. L'indexation intellectuelle est utilisée dans le cadre des bases de données bibliographiques (contenant les références des documents) ; par contre, elle s'avère difficile à mettre en œuvre pour les bases de documents en texte intégral (car elle est coûteuse en temps et en personnel qualifié). L'intérêt pour l'indexation automatique, en particulier lorsqu'il s'agit de gros documents, s'est accru : elle doit permettre une meilleure adéquation entre le texte et sa représentation, et garantir une neutralité de cette représentation.

I-1) L'indexation automatique

Nous présentons ici succinctement la problématique de l'indexation automatique, et plus particulièrement, les travaux de recherche de notre équipe sur lesquels nous nous sommes fondé.

I-1-1) Présentation générale

• Généralités

Avec l'indexation automatique classique, tous les mots du texte, sauf les "mots vides" sont conservés comme descripteurs du contenu du texte pour être utilisés à l'interrogation. Un dictionnaire des "mots vides" est établi préalablement ; il contient notamment des formes grammaticales comme les prépositions, les articles, les pronoms.

Cette indexation, sur des chaînes de caractères, de textes pleins est particulièrement exhaustive et concerne toutes les notions abordées dans le document. L'indexation automatique de textes pleins remédie aux lacunes de l'indexation manuelle, mais elle crée d'autres problèmes, en particulier, le pendant à cette exhaustivité est que l'index comporte une multitude d'entrées.

Ainsi, par exemple, les différentes formes conjuguées d'un verbe (ou formes fléchies d'une forme nominale) ne sont pas regroupées sous une seule entrée mais en constituent plusieurs dans l'index ("*indexer*", "*indexez*", "*indexe*",…).

• Utilité de l'analyse automatique

La nécessité de fonder l'indexation automatique de textes sur des traitements linguistiques devient évidente. Grâce à l'analyse du texte, les formes fléchies ou conjuguées d'une même base doivent être regroupées ne formant qu'une entrée de l'index. De même, tous les mots synonymes, présents dans le texte et qui représentent la même notion ("*manuscrit*", "*livre*",…) doivent aussi être reconnus. Il devrait aussi être possible de regrouper dans l'index, les différentes formes nominales, adjectivales et verbales recouvrant la même notion ("*indexer*", "*indexeur*", "*indexation*"). Notons, aussi que les termes de l'index ne doivent pas forcément être des unitermes mais plutôt des syntagmes nominaux[57] ("*compte rendu*", "*système d'exploitation*",…).

Si l'indexation automatique doit rendre compte de tous ces phénomènes possibles, il est clair qu'elle doit utiliser une analyse morpho-syntaxique du texte, voire même une analyse sémantico-pragmatique (pour rendre compte, par exemple, des problèmes de polysémie : "*livre*").

I-1-2) Présentation des travaux de l'équipe

L'indexation automatique de gros documents est une des préoccupations de notre équipe de recherche. De nombreux travaux ont été réalisés, tous conformes à des objectifs précis [Bertier & al., 97][58]. Nous nous intéressons à de gros documents techniques principalement textuels. Ce type de document décrit non seulement des dispositifs complexes, mais contient également des procédures d'emploi ou de maintenance. En conséquence, la recherche d'information doit s'effectuer sur des objets ou concepts mais aussi sur des méthodes. L'index réalisé doit donc comporter : des syntagmes nominaux (représentant les objets) et des verbes (représentatifs des actions) [Paganelli, 97]. Enfin, du fait du volume important des documents considérés, nous supposons que les documents sont structurés conformément à une norme : chaque document possède une structuration logique (c'est-à-dire une présentation en chapitres, parties,…).

---

[57] Ce qui rejoint la tendance actuelle des systèmes d'indexation automatique et de recherche d'information, comme les systèmes SPIRIT, CLARIT,… (cf. [Bertier & al., 97]).

[58] Cette référence correspond au dernier travail de recherche en indexation automatique effectué par l'équipe CRISTAL-GRESEC. Il s'agit du contrat N° 94 K 6427 d'aide à la recherche intitulé "Indexation de gros documents" achevé en 1997.

I-2) Vers une intégration d'un agent INDEX dans TALISMAN II

Nous désirons rester en adéquation avec les travaux de notre équipe[59]. Nous nous plaçons donc dans le cadre d'une indexation automatique de gros documents techniques structurés. Parmi les différents choix possibles pour les candidats descripteurs, nous retiendrons : les syntagmes nominaux ainsi que les verbes.
Par ailleurs, nous avons choisi de faire, du module d'indexation, un agent à part entière ; nous développerons ses interactions avec les autres agents du système.

I-2-1) Description interne de l'agent INDEX

Nous proposons ici une description schématique du contenu de l'agent permettant l'indexation automatique : l'agent Index.

But : construire, à partir d'un texte structuré, un index des descripteurs contenant les entrées lexicales correspondant aux syntagmes nominaux et aux verbes pertinents.

Connaissances :

- Des règles permettant le regroupement de syntagmes nominaux "proches" (comme par exemple : "logiciel de développement" et "progiciel de développement").[60]
- Des formules statistiques de détermination de la pertinence d'un candidat descripteur.

Compétences :

- Création d'une liste (par défaut vide) de candidats descripteurs.
- Ajout d'un nouveau candidat descripteur (qui peut être soit un syntagme nominal, soit un verbe) et de son adresse (correspondant à la localisation du candidat descripteur dans le document ; par exemple : le numéro du chapitre, de la partie, de la phrase).
- Ajout d'une nouvelle occurrence d'un candidat descripteur déjà dans l'index, ainsi que de son adresse. Cela se produit d'une part, si le candidat descripteur de type verbe relève d'une même entrée lexicale (i.e. possédant la même forme infinitive) qu'un candidat descripteur déjà présent dans la liste. Et, d'autre part, l'ajout est réalisé pour un candidat descripteur de type syntagme nominal, identique ou dérivé (par une méthode automatique de calcul de distance) d'un de ceux déjà recensés dans la liste des candidats descripteurs.
- Filtrage des candidats descripteurs afin de ne conserver que ceux qui sont pertinents. Ce filtrage est effectué à l'aide de traitements statistiques de comptage d'occurrences des candidats descripteurs.

Stratégie :

---

[59] En plus, ce sont ceux que nous connaissons le mieux.

[60] Une étude est en cours dans notre équipe sur le calcul de distances dans le cadre de l'analyse automatique du discours [Timimi & Rouault, 97]. Une autre étude a permis l'identification de relations de variations entre termes [Ibekwe-Sanjuan, 97].

1. Création d'une liste de candidats descripteurs (correspondant au fichier index dont le contenu est vide à l'initialisation).

2. Examen du document à indexer et analyse de chaque candidat descripteur détecté (i.e. étude de tous les syntagmes nominaux et de tous les verbes).

   2.1 Si le candidat descripteur ne se trouve pas encore dans la liste des candidats descripteurs , ajout d'un nouveau candidat descripteur et de son adresse.

   2.2 Si le candidat descripteur se trouve déjà dans la liste des candidats descripteurs, ajout d'une nouvelle occurrence de ce candidat descripteur et de son adresse.

3. Filtrage (à l'aide d'une méthode statistique) des candidats descripteurs à la fin de l'analyse automatique du document ; ce qui permet à l'agent Index de construire un index des descripteurs contenant les syntagmes nominaux et les verbes pertinents.

Exemple d'index :

Supposons que nous disposions de l'index suivant en cours d'élaboration :

```
créer, nombre d'occurrences 2, (adresse1, adresse2)
fichier, nombre d'occurrences 1, (adresse3)
répertoire, nombre d'occurrences 2, (adresse2, adresse4)
système d'exploitation, nombre d'occurrences 1, (adresse1)
```

Supposons que l'on détecte :

- les syntagmes nominaux "les fichiers" à l'adresse5 et "l'indexation" à l'adresse 6,
- et le verbe conjugué "créera" à l'adresse 7.

L'index deviendra alors :

```
créer, nombre d'occurrences 3, (adresse1, adresse2, adresse7)
fichier, nombre d'occurrences 2, (adresse3, adresse5)
indexation, nombre d'occurrences 1, (adresse6)
répertoire, nombre d'occurrences 2, (adresse2, adresse4)
système d'exploitation, nombre d'occurrences 1, (adresse1)
```

I-2-2) Interactions avec les autres agents

Nous avons choisi de retenir en tant que descripteurs, les verbes et les syntagmes nominaux. Ceci nous conduit à définir deux types d'interaction indispensables à l'agent Index :

• Interactions avec Morph pour les verbes

L'agent Morph doit fournir pour chaque phrase tous les verbes qu'elle contient. Morph fournit bien évidemment tous les verbes non ambigus (de catégorie V) qu'il reconnaît lors de sa première levée d'ambiguïtés ; mais aussi, ceux qui après désambiguïsation syntaxique se révèlent finalement non ambigus. Le cas des verbes restant ambigus après analyse syntaxique pose évidemment problème.

L'agent Morph doit aussi fournir les formes au participe passé et au participe présent qui relèvent de la catégorie nominale (notée F).

E*xemples :*

- les verbes non infinitif (ex. "créera") deviennent des candidats descripteurs à l'infinitif ("créer"),
- les verbes à l'infinitif ("ex. : "créer") restent, en tant que candidat descripteur à l'infinitif ("créer"),
- les formes au participe présent noté F (adjectif, participe présent) (ex. : "créant") sont indexées sous leur forme verbale infinitive ("créer"),

- les formes au participe passé noté F (adjectif, participe passé) (ex. : "créée") sont indexées sous leur forme verbale infinitive ("créer").

• Interactions avec Synt pour les syntagmes nominaux

Quant à l'agent Synt, il doit fournir à l'agent Index tous les syntagmes nominaux qu'il détecte. Il faut alors supprimer le déterminant du syntagme, et ramener les nominaux du syntagme à leurs entrées lexicales.

E*xemples :*

- les syntagmes nominaux composés d'un déterminant et d'un nominal (ex. "le système", "ces systèmes") deviennent des candidats descripteurs de type nominal unique ("système"),
- les syntagmes nominaux composés d'un déterminant et de plusieurs nominaux (ex. "les systèmes performants", "ce nouveau système", " le répertoire maître") deviennent des candidats descripteurs de type nominal multiple ("système performant", "nouveau système", "répertoire maître"),
- pour l'instant, en ce qui concerne les syntagmes nominaux plus complexes (ex. "la création de fichiers"), nous supposons qu'ils sont indexés de la même manière ("création de fichier"). Peut-être faudrait-il envisager d'indexer aussi, pour cet exemple, les nominaux "création" et "fichier".

Remarque : le cas des syntagmes prépositionnels ne pose pas de problème dans la mesure où un syntagme prépositionnel est reconnu par la grammaire en constituants comme composé d'une préposition suivie d'un syntagme nominal (ex.: "sur les places" est reconnu "sp((p("*sur*"), sn(d("*les*"), f("*places*"))) ; c'est ce dernier (i.e. sn(d("*les*"), f("*places*"))) qui sera donc étudié comme candidat descripteur.

• Conclusion

Nous avons étudié la possibilité d'utiliser TALISMAN II pour effectuer de l'indexation automatique et nous avons envisagé les conséquences de l'intégration d'un agent Index. De nombreux problèmes, qui relèvent plus de l'indexation automatique, restent en suspens. Notamment, le problème des nominalisations déverbales ("ex. "création") : doit-il faire l'objet des deux entrées lexicales dans l'index ("créer" et "création") ou de seulement de l'entrée nominale ("création"). Ceci pose le problème de la portée (processuelle ou statique) de la nominalisation.

II) Vers une validation de TALISMAN II

Nous présentons plusieurs perspectives possibles de nos travaux. Les deux premiers des travaux que nous présentons ont déjà fait l'objet de collaborations au sein de notre équipe de recherche.

II-1) En linguistique : intégration de la morphologie dérivationnelle

Dans notre équipe, [Clavier, 96a] a élaboré un modèle de la suffixation pour l'analyse automatique du français. Notre système TALISMAN II ne comportant qu'une morphologie flexionnelle, il nous a semblé intéressant d'envisager l'intégration de l'aspect dérivationnel. Notre collaboration nous a donc permis d'évaluer les conséquences de cette intégration [Clavier & al., 96].

• Présentation du module de suffixation

Le modèle linguistique décrit dans [Clavier, 96a] a permis de définir les dictionnaires des unités morphologiques simples (les lexèmes[61] et les suffixes) et une grammaire dérivationnelle de la suffixation.

Cette analyse morphologique suffixale doit permettre, par exemple, la reconnaissance des formes suivantes :

"*rangement*" → "*ranger*" + "*-ment*"
"*feuillage*" → "*feuille*" + "*-age*"

• Description du nouvel agent Morph

Si nous disposions d'un modèle complet de la morphologie intégrant la suffixation, la préfixation et la composition, nous serions amené à prendre position sur le nombre et la répartition des agents dans l'architecture. Dans l'état actuel de l'étude, seules la suffixation et la flexion sont envisagées. Nous avons donc choisi de concevoir le module suffixal comme une extension du modèle flexionnel.

Schématiquement, les différentes composantes du nouvel agent Morph sont les suivantes :

But : trouver la ou les bonnes analyses d'une forme dans son contexte.

Connaissances :

- communes (à la flexion et à la dérivation) : dictionnaire des lexèmes, règles morphologiques contextuelles de désambiguïsation, liste des compatibilités flexion-suffixe,
- spécifiques à la flexion : liste des flexions et de leur compatibilité mutuelle,
- spécifiques à la suffixation : dictionnaire des suffixes.

Compétences :

- communes : module de désambiguïsation,
- spécifiques à la flexion : régularisations des flexions et des bases,
- spécifiques à la suffixation : régularisations des allomorphies sur les suffixes et sur les lexèmes-bases.

Stratégie : Analyse flexionnelle, analyse dérivationnelle, désambiguïsation.

Nous allons illustrer le fonctionnement interne de cet agent Morph par un exemple. Supposons que l'agent doive analyser la forme "*productions*" :

- identification de la flexion "*s*",
- recherche de "*production*" dans le dictionnaire des lexèmes,
  → échec de la recherche (car les mots dérivés ont été exclus du dictionnaire),
- activation du module de la suffixation,
  → identification du suffixe "*tion*",
- recherche de "*produc*" dans le dictionnaire,
  → identification de la racine "*produc*",

---

[61] On appelle "lexèmes" les morphèmes lexicaux : ainsi, dans "*abricotier*", "*abricot*" est un lexème et "*-ier*" est le suffixe.

- validation de "*produc*" associé au lexème verbal "*produire*" avec le suffixe "*tion*",
- obtention de la solution : "*produc*" + "*tion*" + "*s*" (nom déverbal).

Dans cet exemple, la solution est unique, ce qui n'est pas toujours le cas. Ainsi, par exemple, "*dictions*" est analysé morphologiquement comme un verbe (par l'analyse flexionnelle) et comme un nom déverbal (par l'analyse suffixale).

• Conséquences sur le système

L'intégration du module de la suffixation dans l'agent Morph a des conséquences sur les autres agents du système et, en particulier, sur les interactions que l'agent entretient avec les autres. En effet, cette intégration ne modifiera pas les échanges que Morph avait avec les autres agents du systèmes, mais elle créera de nouvelles interactions. Ainsi, la suffixation aura recours à des connaissances issues du niveau syntaxique.
Par exemple, lors de l'analyse des noms déverbaux, la sous-catégorisation des verbes — faisant partie des connaissances de Synt — permet d'interpréter plus finement une segmentation. Ainsi, lors de l'analyse du syntagme nominal "*manquement à son devoir*", la décomposition de la forme "*manquement*", en base et suffixe permet de retrouver le lemme "*manquer*", et, si l'on prend en compte le syntagme prépositionnel "*à son devoir*", l'on peut retrouver le schéma rectionnel C0 V C2 (le complément d'objet indirect C2 étant introduit par la préposition "*à*") associé au lemme "*manquer*". Ceci doit donc permettre de rattacher le syntagme prépositionnel à la forme déverbale "*manquement*" ; et, donc de ne pas envisager le syntagme prépositionnel "*à son devoir*" comme indépendant de "*manquement*", ce qui serait une solution erronée. En fait, cette stratégie permet de faire un choix de rattachement du syntagme prépositionnel et d'éliminer des solutions concurrentes.

• Conclusion

Nous avons présenté un nouvel agent Morph qui a pour vocation de reconnaître et d'interpréter les formes fléchies et suffixées du français. Cet agent doit permettre d'optimiser l'analyse morphologique et de fait, l'analyse syntaxique des textes. En fait, cet analyseur morphologique est plus robuste (cf. [Clavier, 96b]), i.e. il est capable de reconnaître plus de formes pour deux raisons. D'une part, en nous plaçant dans une perspective calculatoire, nous avons donc choisi de diminuer la taille du dictionnaire mais d'augmenter le volume des règles de reconnaissance des formes. Nous pouvons donc reconnaître tous les mots dérivés bien construits. D'autre part, cet analyseur est aussi capable de reconnaître des néologismes (*Jospin* / *jospiniste*). Evidemment, le pendant de cet analyseur est de produire plus de solutions concurrentes qu'il faudra alors désambiguïser.

II-2) En informatique : utilisation des protocoles linguistiques pour la génération

De nombreux travaux ont été réalisés en génération automatique de textes, notamment au sein de notre équipe de recherche : un générateur "stratificationel" (ou séquentiel) fondé sur des règles linguistiques [Balicco, 93], et la définition d'un générateur "noyau" [Ponton, 96]. Ces travaux ont, entre autres, montré les limites de l'approche stratificationnelle pour les systèmes de génération automatique de textes proposés. Nous avons donc tenté de déterminer si une approche distribuée (similaire à celle de TALISMAN II) pouvait être utile au système de génération développé. Nous avons effectué une première spécification d'un système distribué de génération [Antoniadis & al., 96] que nous présentons ici. L'architecture distribuée proposée est de type multi-agents avec une communication directe entre agents cognitifs utilisant les protocoles de communication présentés au chapitre 2.

• Définition des agents

Nous avons défini différents "agents de génération" en reprenant les modules nécessaires à la génération [Balicco, 93] :

L'agent *liaison* traite les contraintes liées à l'application et au(x) destinataire(s). Son rôle consiste à préparer et à valider le travail des autres agents. Il sert aussi de point d'entrée du système puisqu'il reçoit la représentation formelle du texte à produire. L'agent Liaison doit reconnaître le formalisme d'entrée utilisé (dans la mesure où il en accepte plusieurs) ; et il doit transformer cette entrée de manière à ce que le générateur (qui ne comprend qu'un seul formalisme) puisse travailler.

L'agent de *choix lexicaux* : l'agent expert en choix lexicaux a en charge le choix des mots réalisant la structure conceptuelle d'entrée ; cette opération est nécessaire car tout le mécanisme de génération se fait à partir des formes canoniques. Il accomplit sa tâche en faisant appel à d'autres agents.

Par ailleurs, le générateur fonctionne en deux étapes. D'une part, il planifie le contenu du texte et, d'autre part, il réalise la forme de surface. Nous avons conservé cette présentation en deux parties pour regrouper les agents ayant des rôles similaires.

Les agents de planification : leur rôle est de détecter puis de mettre en œuvre une opération linguistique avec des connaissances locales et/ou en faisant appel aux connaissances et aux compétences d'autres agents. Ces agents sont "spécialistes" d'une opération linguistique : ils détectent si l'opération peut être appliquée en fonction du contenu du texte (l'objectif est d'être fidèle au contenu de départ) ; si elle est compatible avec les autres opérations linguistiques détectées (pour ne pas créer de phrases incorrectes ou ambiguës) ; puis ils réalisent l'opération linguistique (cette opération est d'ordre syntaxique).

L'anaphorisation et la *coordination* sont deux exemples d'agents de planification.

Les agents de surface : ils permettent de produire le texte en fonction des décisions prises par les agents de planification. Leur rôle est d'opérer un traitement à la demande d'un autre agent, en général ne faisant pas partie de la classe des agents de surface. Le *générateur morphologique* et les *traitements de surface* sont deux exemples d'agents de surface.

L'agent *Générateur Morphologique* : il possède des connaissances de nature morphologique : il calcule les formes fléchies des nominaux, les formes conjuguées de verbes…

L'agent *Traitement de Surface* : il permet de traiter certains cas de mise en forme du texte à produire ; il insère une majuscule en début de phrase, il traite des élisions ("*le*" "*avion*" devient "*l'avion*"), il effectue le regroupement (s'il est nécessaire) des formes "*de*" + "*le*" en "*du*", il a en charge l'insertion de ponctuations…

• Architecture générale

L'architecture du système que nous avons définie est présentée dans la figure suivante :

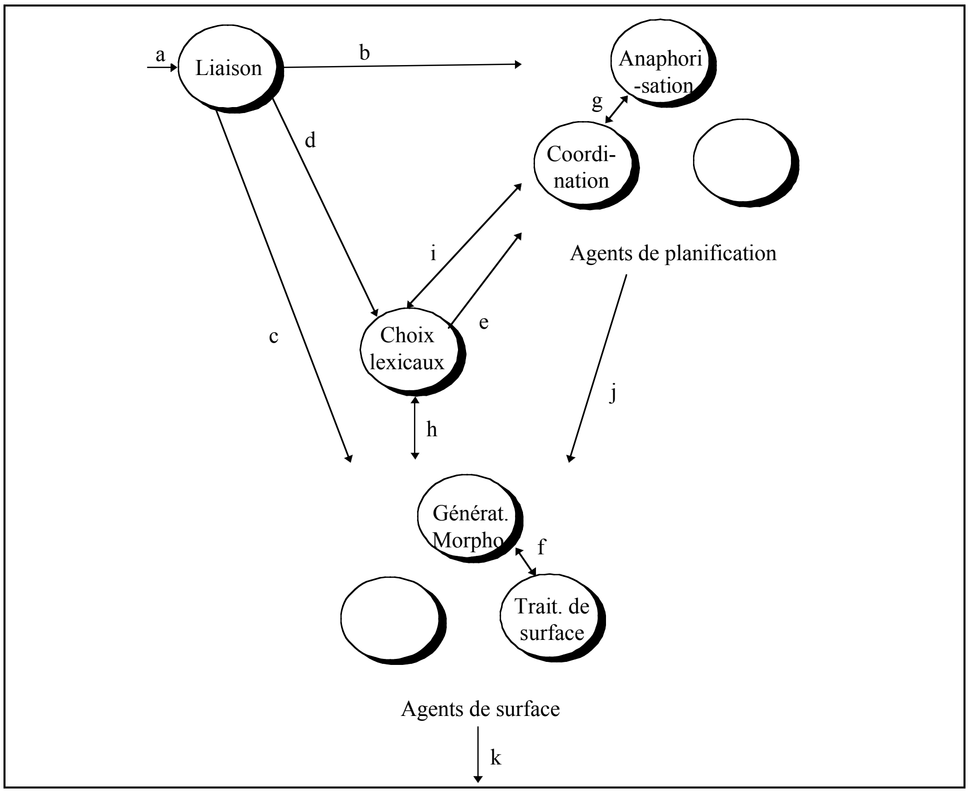


Figure 30 : Architecture du système distribué de génération

Nous avons recensé trois différents types d'interactions :

- Tout d'abord, il existe des envois d'informations qui n'apparaissent qu'une seule fois et qui permettent d'initialiser le système. La représentation formelle du texte ainsi que les caractéristiques de la situation énonciative (utilisateur, buts de l'application...) sont envoyées à l'agent *Liaison* (a). Les caractéristiques sont exprimées par l'agent en terme de paramètres du système et sont passées aux *agents de planification* (b) (ex: opérations linguistiques à privilégier ou non...) et aux *agents de surface* (c) (ex: nombre mots, nombre de phrases...). Ces paramètres sont également adressés à l'agent *Choix lexicaux* (d) (ex: nombre de mots par phrase, vocabulaire plus ou moins spécialisé...) avec la représentation formelle du contenu. Cet agent a pour rôle de sélectionner, en fonction des différents paramètres et en interaction avec les autres agents, les mots pouvant exprimer la représentation formelle donnée et de construire une représentation arborescente du texte. Celle-ci sera ensuite envoyée aux *agents de planification* (e).

- Le second type d'interaction qui apparaît est la coopération entre agents. D'une part, il y aura des coopérations entre les différents agents de traitements (f) (par exemple, les traitements de surface ont besoin d'informations de type morphologique[62]) et entre les différents agents de planification (g) (pour la combinaison et la compatibilité des différentes opérations linguistiques). D'autre part, la coopération avec l'agent *Choix lexicaux* nécessite les compétences des différents agents de surface (h) et des agents de planification (i) à cause de l'interdépendance des choix ; par exemple, un changement de mot (fait par l'agent *Choix lexicaux*) peut entraîner un changement des opérations linguistiques réalisables.

- Enfin, il existe des envois d'informations qui peuvent se produire plusieurs fois et qui achèvent le déroulement du système. En fait, tout au long du second type d'interaction, toutes les solutions possibles, acceptées par les agents de planification, sont envoyées aux agents de surface (j). Ces derniers génèrent alors les solutions attestées par leurs contraintes (qui proviennent d'une part de leurs compétences personnelles et d'autre part, de certains paramètres comme "nombre de phrases minimum") ; ces solutions correspondent à des textes fournis au destinataire (k).

---

[62] Par exemple, pour savoir si "*de*" + "*le*" peut être regroupé en "*du*" (c'est le cas pour "*de le rythme*", mais ce n'est pas le cas de "*de le voir*").

• Exemple de fonctionnement

L'exemple présenté ici n'est décrit que sur quelques points de fonctionnement. La représentation formelle (sous forme de graphes conceptuels [Ponton, 96]) du contenu est la suivante :

(P1) : EVANOUIR <-(agt)- DIMINUER -(obj)-> RESPIRATION
(P2) : EVANOUIR <-(agt)- DIMINUER -(obj)-> RYTHME CARDIAQUE

Les paramètres sont:

- *nombre de phrases minimum*
- *nombre de mots minimum*
- *éviter les répétitions*

Les paramètres sont envoyés aux agents de planification **(b)**, de surface **(c)** et *Choix lexicaux* **(d)**. La représentation formelle est envoyée à l'*agent Choix lexicaux* **(d)**.

Nous ne présentons le choix des mots que pour le concept EVANOUIR. L'agent *Choix lexicaux* traite les propositions consécutivement. Pour (P1), il a le choix entre "*évanouir*" et "*perdre conscience*". Le paramètre "*nombre de mots minimum*" fait que "*évanouir*" est choisi. Pour (P2), les possibilités sont les mêmes. Cette fois, c'est "*perdre conscience*" qui est préféré à "*évanouir*" car il permet d'éviter une répétition (en considérant ce paramètre comme prioritaire sur le nombre de mots). L'agent *Choix Lexicaux* demande alors l'avis aux agents de planification **(i)** et aux agents de surface **(h)**. Pour les agents de surface, la réponse est "*ignore*" car un choix ou l'autre est indifférent en regard des paramètres donnés **(c)**. Par contre, pour les agents de planification, la coordination essaie de rechercher des opérations linguistiques, n'en trouve pas (les mots étant différents) et demande alors au choix des mots une autre solution. L'agent *Choix lexicaux* prend alors la décision de sélectionner "*évanouir*" puisque les opérations linguistiques sont privilégiées.

Les agents de planification détectent une coordination entre (P1) et (P2) et une anaphorisation du sujet de (P2), mais ces deux opérations ne sont pas compatibles **(g)**. Ces deux possibilités sont envoyées aux agents de surface **(j)**. Ces agents préfèrent la coordination à l'anaphorisation (paramètres : nombre de mots et de phrases minimum), puis ils réalisent le texte final ("*S'évanouir diminue la respiration et le rythme cardiaque.*") qu'ils adressent à l'utilisateur **(k)**.

• Conclusion

L'approche distribuée présente au moins deux avantages pour la génération : d'une part, elle permet de simuler convenablement la coopération entre différentes tâches de génération et d'en tirer profit ; d'autre part, elle libère de la nécessité de l'approche stratificationnelle (nécessité de donner la priorité au traitement). Ainsi, les tâches de surface et tâches de planification coopèrent au même niveau pour l'accomplissement du but commun : la production du texte.

II-3) En TAL : Intégration d'autres phénomènes linguistiques

Nous étudions quelques phénomènes linguistiques complexes représentant des préoccupations de recherche au sein de notre équipe ; cette catégorisation n'est pas limitative. Nous avons déjà abordé (dans [Stéfanini & Warren, 96a]) l'analyse de phénomènes linguistiques complexes dans un système multi-agents.
Pour chaque phénomène linguistique, nous en proposons une présentation structurée (dont le contenu n'a pas la prétention d'être exhaustif), avant d'exposer des stratégies possibles de traitement et/ou de résolution de certains cas. Les résolutions proposées sont de deux types : résolution interne à un agent et résolution coopérative.

II-3-1) Transformations

Nous rejoignons l'idée de "grammaire transformationnelle" proposée par Chomsky. Nous utilisons des règles de transformations permettant de ramener des phrases à un ordre canonique[63]. Ainsi, toute grammaire de la proposition assertive pourra analyser ces phrases.

II-3-1-1) Présentation

Les transformations ont pour but de ramener l'ordre des mots d'une phrase dans un ordre canonique ; dans une perpective d'analyse de ces phrases par la grammaire hors-contexte de la proposition assertive que nous avons prototypée (présentée au chapitre 3 et développée en annexe). Les transformations concernent : les constructions clivées, les interrogatives, les impératives, les constructions inversées et les constructions impersonnelles.

- **Les constructions clivées**

Les constructions clivées ont pour but de mettre en relief un constituant avec une anaphore : elles comprennent les phénomènes de la thématisation et de la rhématisation.
Lors de la rhématisation, le "rhème" (l'information nouvelle, importante de la phrase) est introduit par "*c'est*", "*voilà*", "*il y a*"… et suivi d'un introducteur de proposition "*que*", "*qui*",… Ce type de phrase pourrait être décomposé en deux propositions, mais il nous paraît plus pertinent d'effectuer une transformation tout en conservant l'information du rhème.
Lors d'une thématisation, le "thème" (terme désignant une ancienne information) est suivi d'une virgule (ou dislocation) puis d'une reprise pronominale.

Exemple de rhématisation :
"De tous les alliés de l'URSS, *c'est* la Pologne *qui* est apparue le plus acquis à la transparence."

Exemple de thématisation :
"Quant *aux projets, ils* portent sur un accroissement des libertés, sur le recours au référendum, sur la rénovation du

---

[63] Nous nous sommes inspiré des travaux de [Berrendonner, 90].

> rôle du Parlement, sur la lutte contre les inégalités, la réforme du système éducatif, la participation, l'immigration."

• Les interrogatives

Parmi les phrases interrogatives nous distinguons les phrases interrogatives totales des partielles.

- Questions totales

Une question totale est une phrase nécessitant une réponse par *oui* ou par *non* ; en d'autres termes, l'interrogation porte sur la proposition.

> Exemple : "Etes-vous prêt à y travailler avec d'autres organisations syndicales ?"

- Questions partielles

Nous appelons question partielle, toute phrase interrogative nécessitant une réponse différente de *oui* ou *non* ; l'interrogation porte alors sur un constituant de la proposition.

> Exemple : "Comment vous situez-vous aujourd'hui face à la confédération ?"

• Les impératives

Une phrase impérative comportant une absence du sujet peut faire l'objet de transformations.

> Exemple : "Organisons déjà des concours non truqués !"

• Les constructions inversées

Nous qualifierons de constructions inversées toute inversion du sujet par rapport au verbe (n'apparaissant pas seulement dans une phrase interrogative).

> Exemple :
>
> "Il n'est pas possible que nous nous mêlions des affaires intérieures de l'URSS", *a déclaré Mr Ozal*, cité par l'agence semi-officielle Anatolie.

• Les constructions impersonnelles

Une construction impersonnelle est caractérisée par la substitution du réel sujet de la phrase par le pronom "*il*".

> Exemple :
>
> "*Il* n'est pas possible *que nous nous mêlions des affaires intérieures de l'URSS…*"

II-3-1-2) Résolution

Le rétablissement de l'ordre canonique d'une phrase est assuré par l'agent Transf. La détection d'une interrogative ou d'une impérative (ou exclamative) est faite par l'agent Regul grâce à la présence du point d'interrogation ou d'exclamation. Cette information est envoyée à l'agent Transf, qui va alors appliquer ses différentes stratégies. Ce dernier, a souvent recours aux agents Morph et Synt qui permettent de reconnaître la nature des différents constituants, afin de permettre à l'agent Transf de rétablir l'ordre canonique de la phrase.
L'agent TRANSF a donc pour but de transformer une phrase en phrase relevant d'un ordre canonique. Pour cela, il dispose des compétences suivantes : détection et traitement de phrases clivées, de constructions inversées, de phrases impersonnelles, de formes figées, d'interrogatives, d'impératives. Nous proposons des stratégies possibles de transformations de phrase pour rétablir un ordre canonique.

A) Résolution interne à l'agent TRANSF

• Les constructions clivées

Une construction clivée, telle une rhématisation, est introduite par "*c'est*", "*voilà*",… suivi d'un introducteur de proposition "*que*", "*qui*",…
La stratégie interne à l'agent est la suivante :

1. Marquage des formes (entre "*c'est*", "*voilà*",… et de l'introducteur de propositions) par rhème,
2. Suppression des formes "*c'est*", "*voilà*",… et de l'introducteur de propositions,

Exemple :

"De tous les alliés de l'URSS, *c'est* la Pologne *qui* est apparue le plus acquis à la transparence."

→ "De tous les alliés de l'URSS, [la Pologne]$_{\text{rhème}}$ est apparue le plus acquis à la transparence."

• Certains cas de questions totales

Nous distinguons deux types de questions totales : celles introduites par "*est-ce que*" et celles comportant une inversion du sujet non introduites par "*est-ce que*".
Dans le premier cas, l'agent Transf n'a pas besoin des connaissances des autres agents.

1. Suppression des formes "*est-ce que*" et "?",
2. Annotation de la phrase par QTO.

Exemple :

"*Est-ce que* cela ne passe pas par un système d'aides centralisé pour les institutions à caractère national et international, et largement décentralisé pour les établissements de rayonnement local, un peu comme le système mis en place pour l'aide aux arts plastiques *?* "

→ "cela ne passe pas par un système d'aides centralisé pour les institutions à caractère national et international, et largement décentralisé pour les établissements de rayonnement local, un peu comme le système mis en place pour l'aide aux arts plastiques" QTO

B) Résolution par coopération

• D'autres cas de questions totales

Dans le second cas de transformation de questions totales, les traitements suivants sont nécessaires :

1. Suppression des formes "-" et "?",
2. Replacement du sujet avant le verbe, ce qui nécessite des connaissances de nature syntaxique (grâce à l'agent Synt) sur la phrase.
3. Annotation de la phrase par QTO,

Exemple :

" *Sont-elles* les anges ou les démons de cette histoire très compliquée *?*"

→ "*elles sont* les anges ou les démons de cette histoire très compliquée" QTO

Cette stratégie peut être affinée, comme par exemple, en supprimant le pronom "*il*" de la phrase suivante :

Exemple :

"*Cet Etat est-il* voué à disparaître, tôt ou tard, sans laisser de traces, au sein d'une Allemagne réunifiée ?"

→ "*Cet Etat est* voué à disparaître, tôt ou tard, sans laisser de traces, au sein d'une Allemagne réunifiée" QTO

• Cas des questions partielles

Une question partielle est détectée (en plus de la présence du point d'interrogation) par la présence d'un introducteur, en début de phrase, comme *que*, *qui*, *où*, *quand*,…

D'une manière générale, les traitements suivants sont effectués :

1. Rétablissement, si nécessaire (c'est-à-dire s'il y a eu inversion sujet-verbe), du sujet avant le verbe (à l'aide de l'agent Synt), avec suppression du trait d'union (s'il existe),
2. Remplacement de l'introducteur de question indirecte par la forme "%",
3. Déplacement de la forme "%" qui remplace un complément du verbe.
4. Suppression du point d'interrogation final et annotation par QPA,

Exemple :

"comment vous situez-vous aujourd'hui face à la confédération ?"

→ "vous vous situez % aujourd'hui face à la confédération" QPA

• Les impératives (ou exclamatives)

Nous ne traitons que les impératives signalées par la présence d'un point d'exclamation. Mais, toutes les phrases comportant un point d'exclamation ne sont pas forcément des phrases impératives ; elles peuvent être des phrases exclamatives lorsque le sujet du verbe est présent (notées EXC). L'agent Transf effectuera les traitements :

1. Suppression du point d'exclamation,
2. Rétablissement, si nécessaire (i.e. si la phrase est une impérative et que le sujet manque), du pronom personnel (en accord avec la personne) à l'aide des informations de Morph,
3. Annotation de la phrase par IMP ou EXC (si aucun sujet ne manque).

Exemples :

"organisons déjà des concours non truqués !"

→ "nous organisons déjà des concours non truqués" IMP

"c'est tout de même un comble : on a fait voter un mort !"

→ "c'est tout de même un comble : on a fait voter un mort" EXC

• Les constructions inversées

Les constructions inversées sont signalées par l'agent Synt qui ne trouve pas de sujet avant un verbe conjugué. L'agent Transf va rétablir l'ordre canonique d'une construction inversée de la manière suivante :

1. Recherche du sujet après le verbe (suivi ou non d'un trait d'union),
2. Vérification de l'accord entre le sujet et le verbe auprès de Synt,
3. Suppression du trait d'union s'il existe, ou des formes "-t-" si elles sont présentes,
4. Rétablissement du sujet avant le syntagme verbal.

Exemples :

"Il n'est pas possible que nous nous mêlions des affaires intérieures de l'URSS", *a déclaré Mr Ozal*, cité par l'agence semi-officielle Anatolie."

→ "… Mr Ozal a déclaré…"

"La commission chargée de la mise en œuvre de l'accord-cadre signé le 31 août à Alger s'est réunie il y a quelque temps à Tripoli, mais, *fait-on* remarquer de source informée à Paris, la concrétisation de cet accord tarde."

→ "La commission chargée de la mise en œuvre de l'accord-cadre signé le 31 août à Alger s'est réunie il y a quelque temps à Tripoli, mais, *on fait* remarquer de source informée à Paris, la concrétisation de cet accord tarde."

"Ce que fait Moscou actuellement, c'est une politique impérialiste, *affirme-t-il*."

→ "Ce que fait Moscou actuellement, c'est une politique impérialiste, *il affirme*."

"Ne vous en faites pas pour la perte de l'Europe de l'Est, *le voit-on* dire."

→ "Ne vous en faites pas pour la perte de l'Europe de l'Est, *on le voit* dire."

• Les constructions impersonnelles

L'agent Transf active le traitement des constructions impersonnelles s'il existe dans une même proposition, l'une des deux constructions suivantes :

a) "il + (-) + être + (-) + ADJ[modalité] + que + (Prop)"
sera transformé en "(Prop) IPS{modalité}".
b) "il + (-) + VERBE [impersonnel,modalité] + (-) + que + (Prop)"
sera transformé en "(Prop) IPS{modalité}".

Exemple :

"Il n'est pas possible [que nous nous mêlions des affaires intérieures de l'URSS]…"

→ [nous nous mêlions des affaires intérieures de l'URSS] IPS {négation de "possible"}

Nous avons proposé quelques stratégies de rétablissement de l'ordre canonique de la phrase, mais nous n'avons vraisemblablement pas recensé tous les cas de phrases relevant d'un ordre non canonique.

II-3-2) Reconstruction d'ellipses

Le phénomène linguistique complexe de la reconstruction d'ellipses est un phénomène assez étudié en TAL. Les travaux effectués concernent souvent la reconstruction d'un syntagme verbal elliptique [Bos, 94], [Lappin & Shih, 96].

II-3-2-1) Présentation

Les travaux de [Sauvage, 92] ont permis de définir l'ellipse comme une figure syntaxique ou stylistique pour laquelle on supprime un ou plusieurs mots de l'énoncé sans que la compréhension en souffre. L'ellipse est un phénomène complexe pouvant concerner : un sujet, un verbe, un complément, voire une proposition entière.

Nous retrouvons ces différentes sortes d'ellipses suivant le type de phrase dans lequel une ellipse peut apparaître :

• Ellipse dans les questions indirectes

La proposition est elliptique, lorsque celle-ci comporte une catégorie Q (*qui* / *quand* / *où* etc.) et que cette même proposition ne comporte aucun verbe.

Exemple d'ellipse d'une proposition :

"Suffit de planter deux électrodes je ne sais plus ***où*** *(planter deux électrodes)*, pour capter je ne sais plus ***quoi*** *(capter)*, et dès que ça commence à s'agiter, à s'emmêler les pinceaux en sous-sol, ça déclenche un signal d'alarme. "

• Ellipse dans les comparatives

Les ellipses sont fréquentes dans les phrases comparatives.

Exemple d'ellipse du syntagme verbal :

"En proportion, nous avons fait plus que vous *(n'avez fait)*, a dit en substance le général Schmitt…"

• Ellipse dans les coordonnées

Une ellipse peut aussi apparaître dans la seconde proposition d'une phrase comportant une coordination inter-propositions.

> Exemple d'ellipse du sujet :
>
> "Il appartient en effet à l'une des grandes familles qui ont traditionnellement contrôlé la Banque de Bilbao et *(il)* compte d'ailleurs aujourd'hui parmi ses principaux actionnaires individuels."

• Ellipse dans les subordonnées

Les phrases avec subordonnées peuvent comporter des ellipses de nature diverse (ellipse des propositions complétives ou infinitives apparaissant dans la proposition principale, ellipse de compléments,…). L'ellipse a toujours lieu dans la subordonnante et non dans la subordonnée.

> Exemple d'ellipse de complément :
>
> "Outre le fait d'avoir, sur ce point, raison d'insister, Jean-François Kahn surprend par son érudition : [qu'il s'agisse de la philosophie des sciences, de l'histoire de France, des philosophes "kantiens"], [il connaît *(la philosophie des sciences, l'histoire de France, les philosophes "kantiens")*]."

II-3-2-2) Résolution

Certaines ellipses sont détectées par les agents Synt ou Segm qui ne trouvent pas un sujet ou un verbe par exemple. Ils en informent l'agent Ellip. L'agent Ellip essaie alors de reconstruire l'ellipse du sujet ou de la proposition avec l'aide d'autres agents.

L'agent ELLIP a pour but de reconstruire les différents types d'ellipses. Il détecte et reconstruit, suivant les cas, l'ellipse du sujet, du verbe, du complément ou de la proposition.

Nous allons proposer quelques heuristiques possibles de rétablissement de certains cas d'ellipses.

Résolution de ELLIP avec l'aide d'autres agents

• Ellipse dans les questions indirectes

Nous proposons une stratégie possible de résolution d'une ellipse dans une question indirecte (cf. [Stéfanini & Warren, 96b]). La proposition est elliptique, lorsque celle-ci comporte une catégorie Q (*qui* / *quand* / *où* etc.) et ne comporte aucun verbe suivant l'introducteur, dans la même proposition. Nous distinguerons deux cas de question indirecte dans l'exemple suivant :

> " [Suffit [de planter deux électrodes] [je ne sais plus ***où***] $_{\text{question indirecte 1}}$, [pour capter [je ne sais plus ***quoi***]] $_{\text{question indirecte 2}}$, et dès que ça commence à s'agiter, à s'emmêler les pinceaux en sous-sol, ça déclenche un signal d'alarme. "

Nous dirons que la première question indirecte relève d'une proposition enchâssant une proposition infinitive ; alors que la seconde question indirecte est introduite par une proposition infinitive n'enchâssant aucune autre proposition. Nous allons traiter ces deux cas :

1^{er} cas : Ellipse d'une proposition "simple" dans une question indirecte

Dans ce cas, la proposition faisant l'objet d'une ellipse n'enchâsse ni une infinitive ni une complétive, l'ellipse est celle de cette proposition.

> Exemple d'une proposition infinitive suivie d'une question indirecte :
>
> "...pour capter je ne sais plus ***quoi*** *(capter)*, et dès que ça commence à s'agiter, à s'emmêler les pinceaux en sous-sol, ça déclenche un signal d'alarme. "

La reconstruction de l'ellipse (ressemblant à la stratégie présentée précédemment) peut être réalisée comme suit :

1. Détection de la catégorie Q sans verbe V après dans une même proposition.
2. Détection de la principale sans complétive ou infinitive à l'aide de l'agent Segm.
3. Rétablissement de la principale.

2^{ème} cas : Ellipse d'une proposition enchâssant une complétive ou une infinitive dans une question indirecte

Dans ce cas, la proposition faisant l'objet d'une ellipse enchâsse une complétive ou une infinitive, comme dans l'exemple suivant où deux interprétations sont possibles :

> "Suffit [de planter deux électrodes] je ne sais plus ***où***,..."
>
> a- suffit de planter deux électrodes.
>
> b- planter deux électrodes.

La reconstruction des ellipses peut être réalisée comme suit :

1. Détection de la catégorie Q sans verbe V après dans une même proposition.
2. Détection de la principale avec complétive ou infinitive avec l'agent Segm.
3. Rétablissement de la seule complétive ou infinitive (sans l'introducteur associé à la complétive ou à l'infinitive) ; et rétablissement de la principale (avec la complétive ou l'infinitive).

Cette stratégie devrait être affinée pour tenir compte du problème de la concordance des temps lors de la reconstruction de la proposition elliptique ainsi que de traits sémantiques ne permettant pas les rétablissements systématiques proposés. De plus cette stratégie est insuffisante pour rendre compte de l'ambiguïté suivante :

> "Ces deux décisions, apparemment contradictoires mais de fait complémentaires, démontrent que la question des rapports de la loi et du traité reste ouverte : le Conseil constitutionnel pourrait utiliser ses nouvelles compétences issues de la révision engagée pour faire prévaloir peu à peu la norme internationale, en constatant a posteriori la non-conformité d'une loi à un traité, et en expliquant *pourquoi*."
>
> Où "pourquoi" introduit une question indirecte pouvant faire référence à :

a. "faire prévaloir peu à peu la norme internationale"

b. "la non-conformité d'une loi à un traité"

• Ellipse dans les comparatives

Une phrase comparative est signalée par la présence de "*plus que*", "*moins que*", "*autant que*",… L'ellipse porte toujours sur le verbe associé ou non au sujet et/ou au complément.

Exemples:

1. "En proportion, nous avons fait plus que vous *(avez fait)*, a dit en substance le général Schmitt…"
2. "Le problème est qu'il accomplit là une révolution sémantique plus que *(il accomplit là une révolution)* conceptuelle."

Nous proposons une stratégie possible de reconstruction de l'ellipse :

1. Détection de la présence de "*plus que*", "*moins que*", "*autant que*",…
2. Détection de l'absence de constituants : le syntagme verbal et éventuellement du sujet et/ou des compléments d'objets directs à l'aide de l'agent Synt.
3. Reconnaissance dans la proposition précédente de ces constituants manquants.
4. Rétablissement de ces constituants manquants.

Bien évidemment, toute phrase comportant "*plus que*", "*moins que*", "*autant que*",… n'est pas forcément une phrase comparative. Dans ce cas, cette phrase ne doit pas comporter de constituants manquants et aucun rétablissement d'ellipse n'est effectué, comme dans l'exemple suivant[64] :

"En revanche, nous ne voulons ***plus que*** certains de nos compatriotes restent des "malgré nous" de la marginalisation afférente à cet extrémisme-là."

• Ellipse dans les coordonnées

Des ellipses peuvent apparaître dans une proposition coordonnée.

Exemple d'ellipse du sujet :

"Il appartient en effet à l'une des grandes familles qui ont traditionnellement contrôlé la Banque de Bilbao et *(il)* compte d'ailleurs aujourd'hui parmi ses principaux actionnaires individuels."

Nous proposons une stratégie possible de rétablissement des constituants elliptiques :

1. Détection d'une proposition coordonnée.
2. Recherche de constituants manquants dans la proposition coordonnée (verbe, sujet et/ou compléments d'objets directs) à l'aide de l'agent Synt.
3. Si des constituants manquent, recherche dans la proposition précédente de ces constituants manquants.
4. Rétablissement de ces constituants manquants dans la proposition coordonnée.

---

[64] Notons que la présence du morphème négatif "*ne*" peut permettre dans cet exemple de reconnaître que cette phrase n'est pas une comparative ; mais comme le montre l'exemple situé à la fin de cette section, le problème peut être plus délicat.

• Ellipse dans les subordonnées

Des cas d'ellipses peuvent apparaître dans des phrases comportant des propositions subordonnées. L'ellipse a toujours lieu dans la proposition subordonnante et non dans la subordonnée.

> Exemple d'ellipse de complément :
>
> "Outre le fait d'avoir, sur ce point, raison d'insister, Jean-François Kahn surprend par son érudition : [qu'il s'agisse de la philosophie des sciences, de l'histoire de France, des philosophes "kantiens"], [il connaît *(la philosophie des sciences, de l'histoire de France, des philosophes "kantiens")*]."

La reconstruction de l'ellipse peut être réalisée comme suit :

1. Détection d'une proposition principale précédée ou suivie d'une subordonnée.
2. Recherche dans la proposition principale d'un constituant manquant : un complément d'objet direct, une complétive ou une infinitive à l'aide de l'agent Synt.
3. Si des constituants manquent, recherche dans la proposition subordonnée de ces constituants manquants.
4. Rétablissement de ces constituants manquants dans la proposition principale elliptique.

Comme nous l'avons vu, les stratégies proposées permettent de rétablir certains cas d'ellipses mais, bien évidemment, pas tous. De plus, d'autres types de propositions peuvent aussi introduire des ellipses comme le montre l'exemple suivant :

> Exemple d'ellipse d'un complément :
>
> "La conclusion que l'on veut amener les Chinois à tirer est simplement que toute révolte contre le régime socialiste ne peut que contraindre ce dernier à réprimer (*toute révolte*)."

Nous avons proposé des stratégies de rétablissement de l'ordre canonique d'une phrase et de reconstruction d'ellipses ; ces stratégies sont en cours de prototypage.

III) Conclusion

Nous présentons un bilan de nos travaux de recherche. Ensuite, nous proposons certaines perspectives de TALISMAN II.

III-1) Bilan

Nous établissons un bilan de ce mémoire, aussi bien au niveau pratique que théorique.

III-1-1) Bilan pratique

Dans le cadre d'une architecture multi-agents, nous avons créé un système d'analyse morpho-syntaxique du français écrit, TALISMAN II, dont la perspective est la résolution de solutions multiples.

- **Désambiguïsation morphologique**

Nous avons développé des règles linguistiques contextuelles de désambiguïsation. Les résultats (présentés au chapitre 3) obtenus par les règles de désambiguïsation sont encourageants : sur un corpus de 1354 formes (dont 40 % relevaient de plusieurs entrées lexicales), seulement 10 % subsistent après une première levée d'ambiguïtés. Par ailleurs, nous avons montré que certaines de ces solutions multiples étaient résolues par la suite : par les compétences de l'agent Synt ou par une nouvelle levée d'ambiguïtés de l'agent Morph.

- **Analyse syntaxique**

Nous avons implanté un analyseur syntaxique en constituants utilisant une grammaire hors-contexte de la proposition assertive et une stratégie syntaxique d'analyse fonctionnelle ; et nous avons proposé différentes stratégies d'élimination de solutions parasites syntaxiques.
L'utilisation d'une telle grammaire standard nécessite la mise en place, notamment pour le traitement de la coordination, de méta-grammaires. La méta-grammaire de la coordination fonctionne à partir d'indices précis, par exemple, la présence d'un coordonnant. Nous avons montré l'utilité des grammaires satellites, notamment d'une grammaire de reconnaissance de syntagmes pour la coordination intra-proposition. Cette grammaire satellite s'appuie sur des indicateurs de structures lexicaux (ex. : un déterminant introduit un syntagme nominal) pour reconnaître la nature du syntagme et réaliser son analyse syntaxique.
Nous avons donc proposé un analyseur syntaxique mixte [Stéfanini & Warren, 97] : l'analyse est descendante lors de l'utilisation de la grammaire standard de la proposition assertive ; elle devient ascendante, lorsque l'analyse est effectuée à partir d'indices présents dans le texte : c'est le cas des grammaires satellites de l'agent Synt et de la méta-grammaire de l'agent Coord.

- **Résolution coopérative**

Nous avons développé les mécanismes de communication de TALISMAN II dans une perspective de résolution de conflits. Nous avons prototypé des protocoles de communication autorisant l'échange d'information et la coopération entre agents. Le système permet la gestion, voire la résolution de conflits : certains conflits sont résolus par les compétences d'un agent, les autres conflits sont conservés.

- **Intégration d'heuristiques linguistiques**

Nous avons montré la nécessité de l'utilisation d'heuristiques linguistiques notamment pour l'étiquetage de formes inconnues par l'agent Morph. Nous pensons que les stratégies de prédiction de l'agent Segm (sur le nombre de propositions et sur la segmentation de la phrase en propositions) doivent être affinées.

III-1-2) Bilan théorique

- **Validation de l'adéquation SMA - Analyse automatique**

Les tests effectués sur TALISMAN II (présentés tout au long de ce mémoire) justifient sa validation en grandeur réelle sur un corpus.

Nous avons montré que TALISMAN II permettait la gestion des conflits inhérents à un système de TAL : toutes les solutions multiples sont conservées dans le système aussi longtemps qu'une stratégie linguistique ne permet pas de les éliminer. Ce système permet d'éliminer de nombreuses ambiguïtés relevant des niveaux d'analyse morphologique et syntaxique. Nous avons proposé de nombreuses autres stratégies pour le traitement d'autres ambiguïtés.

L'adéquation entre système multi-agents et analyse automatique est évidente : le multi-agents permet une certaine souplesse des traitements indispensable ; chaque agent peut disposer de "la bonne information au bon moment".

- **Intégration de nouveaux phénomènes linguistiques complexes**

Nous avons illustré par le développement de l'agent Coord, la place particulière des phénomènes linguistiques complexes ; phénomènes relevant de différents niveaux de représentation et nécessitant des stratégies spécifiques.

Nous avons modélisé les stratégies des agents Transf et Ellip qu'il nous faut maintenant implémenter. Nous proposons aussi (cf. § III-2-1) une première spécification des agents Anaph(orisation) et Nega(tion).

- **TALISMAN II : un système générique**

Nous avons montré que le système était flexible et l'insertion de nouveaux traitements, comme un analyseur morphologique dérivationnel, est aisée.

Nous avons développé la communication dans TALISMAN II. Nous avons proposé des protocoles linguistiques que nous désirons appliquer à la génération automatique de textes.

Enfin, nous avons proposé une application directe de TALISMAN II : l'indexation automatique de gros documents techniques.

III-2) Perspectives

Les perspectives qui s'offrent au système TALISMAN II sont multiples aussi bien au niveau linguistique qu'informatique. Certaines de ces perpectives ont déjà fait l'objet d'une étude approfondie.

Du point de vue linguistique, nous proposons l'amélioration des compétences linguistiques des agents existants ; ainsi que le développement de nouveaux agents possédant d'autres compétences linguistiques (comme la résolution d'anaphores, la détermination de la portée de la négation). Des possibilités d'améliorations ont été proposées tout au long de ce mémoire. Une autre perpective est de coupler le module de traitement de mots inconnus à l'analyseur morphologique dérivationnel ; en particulier, le module de suffixation doit

permettre de prédire des informations, à partir de la nature du suffixe, sur une forme inconnue.
Du point de vue informatique, nous citons deux perspectives qui nous semblent importantes. D'une part, l'amélioration des capacités de raisonnement des agents notamment pour les prises de décisions locales et la gestion de leur mémoire locale. Et d'autre part, l'intégration de TALISMAN II dans la plate-forme multi-agents MAGMA [Demazeau, 95], qui permettrait de valider l'architecture proposée.

Nous présentons plus en détail, des perspectives générales à l'analyse automatique de la langue. Ainsi, nous présentons des exemples d'ambiguïtés (issus de notre corpus) liées à certains phénomènes linguistiques complexes ; nous exposons aussi les niveaux sémantique et pragmatique. Et, nous développons quelques stratégies de résolution (interne et coopérative) dans TALISMAN II.

III-2-1) Modélisation d'autres phénomènes linguistiques complexes

Nous allons étudier deux phénomènes linguistiques complexes dans une perspective d'intégration dans TALSIMAN II : l'anaphore et la négation.

III-2-1-1) Traitement d'anaphores

La résolution d'anaphores est un phénomène très complexe [Lacombe, 96] et largement étudié en TAL. Le problème de la résolution d'une anaphore est double : lister les antécédents possibles et choisir, parmi les membres de cette liste, un antécédent. La plupart des auteurs (en TAL) s'intéressent à l'anaphore pronominale [Reboul, 89]. Les algorithmes de résolution d'anaphores proposés utilisent souvent des contraintes syntaxiques [Stuckardt, 96], [Kennedy & Boguraev, 96] mais aussi des contraintes sémantiques et pragmatiques [Carbonnell & Brown, 88]. Dans notre équipe, une étude de l'anaphore dans le cadre du dialogue Homme-Machine a été réalisée par [Parlermiti, 92]. Une autre étude sur l'anaphore associative a été réalisée notamment par [Berrendonner, 89].

A) Présentation

On appelle anaphorique, un segment du discours qui nécessite de se reporter à un autre segment linguistique du même discours, pour lui donner une interprétation. L'anaphore est la relation qui lie un segment du discours avec son antécédent de nature différente : l'antécédent est souvent un syntagme nominal mais pas toujours (un syntagme verbal, par exemple). L'anaphore est fortement liée au phénomène de pronominalisation, mais il n'y a pas que les pronoms qui peuvent être anaphoriques.
Nous distinguons différents types d'anaphores suivant la catégorie de la forme anaphorique : le plus souvent un pronom préverbal ("*il*", "*le*",…), mais aussi un déterminant possessif ou démonstratif ("*sa*", "*cet*",…), ou un "pronom nominal" ("*le sien*", "*le nôtre*",…).

• Les anaphores introduites par des pronoms préverbaux

En général, l'antécédent est un syntagme nominal se trouvant à gauche de la forme anaphorique. Dans le cas, d'une anaphore inter-phrastiques, l'antécédent est extérieur à la phrase : c'est le plus souvent, un syntagme nominal présent dans la phrase précédente.

Exemple d'anaphore inter-phrastique :

> "Ce fils de pasteur de trente-quatre ans, né à Leipzig, écrit, organise des expositions et des concerts depuis une bonne dizaine d'années. *Il* a même été chanteur dans un groupe rock : Teurer denn je."
>
> Où "*il*" renvoie au fils de pasteur de la phrase précédente.

Exemple d'anaphore intra-phrastique :

> "Les Arméniens ont saboté le relais de télévision qui *leur* permettait de recevoir les images soviétiques."
>
> Où le préverbal "*leur*" renvoie au syntagme nominal "Les Arméniens".

• Les anaphores introduites par des déterminants

De même, l'antécédent est, dans les cas simples, un syntagme nominal à gauche du déterminant anaphorique.

> "On entendra du jazz et des auteurs liront *leurs* textes."
>
> Où le déterminant "*leurs*" renvoie au syntagme nominal "les auteurs".

• Les anaphores introduites par des "pronoms nominaux"

> "Mais ces accords ne sont sans doute pas adaptés au niveau d'exigence qui peut être *le nôtre*, comme celui de ces pays pris individuellement et collectivement."
>
> Où le nominal "*le nôtre*" renvoie au syntagme nominal "le niveau d'exigence".

Bien entendu, l'antécédent d'une forme anaphorique peut se trouver à droite de cette forme : c'est le cas des *cataphores*.

> "Dans *leurs* luttes pour la démocratie, la plupart des pays de l'Est se sont appuyés sur *leur* identité nationale."
>
> Où : - une cataphore : le déterminant "*leurs*" renvoie au syntagme nominal "la plupart des pays de l'Est",
>
> - une anaphore : le déterminant "*leur*" renvoie au syntagme nominal "les pays de l'Est".

Pour [Arrivé & al., 86], ces exemples sont du type "anaphore grammaticale". Il existe d'autres types d'anaphores : citons les "anaphores associatives" traitées par [Berrendonner, 89]. Dans l'exemple suivant (issu de l'auteur) de nombreuses inférences doivent être réalisées :

> "J'aperçois un phare. Le conducteur a oublié d'éteindre."

L'inférence proposée par l'auteur est la suivante :

- un "*phare*" est un ingrédient d'une voiture ; donc il y a une voiture,
- un "*conducteur*" est un ingrédient d'une voiture,
- le lien anaphorique peut être fait entre le "*conducteur*" et le "*phare*".

B) Résolution

Dans les cas simples, les formes anaphoriques sont détectées par l'agent Morph (puisqu'elles concernent certaines catégories morphologiques), qui informe l'agent Anaph de leur présence. Le but de *l'agent ANAPH* est de retrouver l'antécédent d'une forme anaphorique. Il établit le lien entre la forme anaphorique et son antécédent.
Nous présentons quelques stratégies possibles de résolution d'anaphores.

Résolution de ANAPH avec l'aide d'autres agents

• Cas de certaines anaphores

Nous proposons une stratégie triviale possible :

1. Recherche, à gauche de la forme anaphorique, d'un syntagme nominal. Lorsque la forme anaphorique est en début de phrase, la recherche de l'antécédent se fera dans la phrase précédente.
2. Appel à l'agent Synt pour la vérification de l'accord en genre et en nombre entre le syntagme et la forme anaphorique. Si échec, recherche à gauche de ce syntagme d'un autre syntagme nominal, etc.[65]

Exemple :

"L'avertissement est clair pour les Occidentaux : s'*ils* n'occupent pas le terrain, les industriels nippons sont désormais prêts à *le* faire à *leur* place comme en Chine."

Où "*ils*" renvoie aux Occidentaux,
"*le*" renvoie au fait d'occuper le terrain,
"*leur*" renvoie aux Occidentaux.

Les pronoms préverbaux anaphoriques sont, dans cet exemple, "*ils*" et "*le*". Pour le premier préverbal, cette stratégie ne pose pas de problème puisque l'antécédent "*les Occidentaux*" sera facilement retrouvé.
Par contre pour le préverbal "*le*", le premier antécédent possible s'accordant en genre et en nombre est "*le terrain*". Cette stratégie doit être affinée pour prendre en compte le fait que le syntagme verbal "*le faire*" implique de retrouver comme antécédent un syntagme verbal (ou proposition) : "*occuper le terrain*".
En ce qui concerne le déterminant anaphorique "*leur*", il est clair que son antécédent à savoir "*les Occidentaux*" sera difficile à retrouver dans la mesure où le premier syntagme s'accordant en genre et en nombre est "*les industriels nippons*". Il est donc impossible de se fonder dans ce cas sur la vérification des accords ; des connaissances sémantico-pragmatiques sont indispensables.

La résolution d'anaphores est un problème particulièrement complexe nécessitant des connaissances sémantico-pragmatiques. Il nous semble cependant que ces connaissances ne seront pas toujours suffisantes comme le montre l'exemple suivant dans lequel il nous paraît impossible de déterminer si la forme "*auxquelles*" se réfère à "*des chaînes câblées*" uniquement, ou à "*des stations indépendantes et des chaînes câblées*".

Exemple : "Leur audience s'érode au profit des stations indépendantes et des chaînes câblées *auxquelles* 60 % des foyers américains ont accès."

---

[65] En cas d'échec de cette stratégie, il faudrait tester si l'on n'est pas en présence d'une cataphore ; et, donc rechercher un syntagme nominal à droite.

III-2-1-2) Portée de la négation

A notre connaissance, il existe peu de travaux en TAL portant sur le phénomène de la négation et sur la détermination de sa portée. Les études sur la négation peuvent être très variables. [Amsili & Hathout, 96] s'intéressent au traitement de la négation temporelle ; dans notre équipe des travaux sont en cours sur la répercussion de la négation linguistique dans la gestion de la cohérence d'une base de données construite à partir de données textuelles [Siameur & Rouault, 96], [Siameur, 97].

A) Présentation

Nous distinguons plusieurs types de négation :

- **Négation préfixale :** introduite par un préfixe ["*mal*", "*in*",…]

  Exemples : "(*mal*)heureux", "(*in*)certitude",…

- **Négation du constituant :** introduite par [*non*, *pas*, *ne...aucun*,…]

  Exemples :

  "…par respect des lois qui doivent libérer et *non* contraindre…"

  "…et *aucun* affrontement armé entre Soviétiques et Azéris *n'* était signalé…"

- **Négation explétive :** introduite par le seul morphème "*ne*" qui n'a alors aucune valeur négative. Ce type de négation apparaît notamment dans les proposition subordonnées comparatives et dans les propositions subordonnées introduites par certains verbes (comme "*craindre*", "*oser*", "*empêcher*"…).

  Exemples :

  "Mais, dans l'immédiat, elle semble avant tout vouloir *empêcher* qu'on *ne* l'entraîne à des décisions trop rapides."

  "…c'est-à-dire de dépenser l'héritage *avant que* le testament *ne* soit lu…

- **Négation prédicative :** introduite par le morphème "*ne*" et suivi d'un autre morphème négatif.

  - Négation totale : exprimée par les morphèmes [*ne…pas*]

  Exemple : "En l'occurrence, le Lez *n'* étant *pas* classé, …"

  - Négation partielle : exprimée par les morphèmes [*ne…guère*, *ne…plus*, *ne…rien*,…]

  Exemple : "En outre, les chaînes câblées généralistes *ne* se contentent *plus* des "restes"…"

- **Négation restrictive :** introduite par [*ne…que*]

  Exemple : "… le fait que les divisions blindées françaises *ne* comptent *que* 200 chars…"

B) Résolution

La présence d'une négation est la conséquence de l'occurrence de formes telles que "*ne*", "*non*", "*aucun*"… dans une phrase. *L'agent NEGA* essaie alors de reconnaître et déterminer la portée de la négation. Nous proposons des stratégies possibles de détermination de la portée de la négation.

B-1) Résolution interne à l'agent NEGA

• Négation préfixale

La détermination de la signification d'un préfixe négatif peut être effectuée par l'agent NEGA. Cette signification n'est pas évidente à déterminer comme le montrent les exemples suivants :

Exemples :

"impossible" est la négation de "possible"

"imberbe" est la négation de "barbe"

"impertinent" ne correspond pas à la négation de "pertinent".

Pour plus de détail, le lecteur se référera aux travaux en cours sur la négation dans notre équipe de recherche [Siameur, 97].

• Négation du constituant

La portée de la négation d'un constituant est en général facile à déterminer : la négation porte sur le constituant suivant (verbe, nominal, syntagme,…).

Exemples :

"…par respect des lois qui doivent libérer et *non* [contraindre]."

"Il s'agit bien du surendettement des seuls ménages, et *non* [des entreprises]."

"… en constatant a posteriori la *non*-[conformité] d'une loi à un traité,…"

Parfois, l'introducteur d'une négation de constituant (comme "*non*"), relève d'une expression ou tournure stylistique comme le montrent les exemples suivants :

"… ***non*** *seulement* en Espagne *mais aussi* en France et en Amérique latine…"

"… voir déférer ***non*** *pas* des lampistes, *mais* quelques têtes connues."

• Négation explétive

Les négations explétives sont caractérisées par la présence, dans la même proposition, de la seule forme "*ne*", c'est-à-dire sans "*pas*", "*plus*",… Le morphème "*ne*" n'a alors aucune valeur négative, il est donc possible de le supprimer de la proposition.

1. Détection de la forme "*ne*" avec absence de toute forme du type "*pas*", "*plus*",… dans la même proposition.
2. Suppression de cette forme "*ne*".

Exemple :

"Mais, dans l'immédiat, elle semble avant tout vouloir *empêcher qu*'on **ne** l'entraîne à des décisions trop rapides."

"…c'est-à-dire de dépenser l'héritage *avant que* le testament ***ne*** soit lu…

Cependant cette stratégie doit être affinée car ces négations explétives n'apparaissent que dans certains cas : après certaines formes négatives (comme "*personne*", "*aucun*",…), ou dans certaines subordonnées introduites par des verbes comme "*empêcher*, "*craindre*",…, ou encore dans certaines subordonnées comparatives du type "*avant que*", *moins bien que*"). Il est donc indispensable d'ajouter ces critères de reconnaissance de négations explétives afin de rendre compte des cas de négation prédicative totale avec absence de la forme "*pas*" comme dans l'exemple suivant :

"CBS, ABC et NBC ***ne*** peuvent donc toucher les bénéfices de programmes qu'ils ont pourtant largement financés et …"

→ "CBS, ABC et NBC ***ne*** peuvent donc ***pas*** toucher les bénéfices de programmes qu'ils ont pourtant largement financés et …"

• Cas de la négation prédicative

Nous proposons une stratégie possible pour l'agent Nega afin de déterminer la portée d'une négation prédicative totale ou partielle :

1. Détection de la forme "*ne*".
2. Recherche à droite de la forme "*ne*", de la forme "*pas*", "*plus*", "*guère*",…
3. Détermination de la portée de la négation :
   3a. Si les morphèmes sont discontinus, la négation porte sur le verbe conjugué[66] situé entre les deux morphèmes.
   3b. Si les morphèmes sont contigus, la négation porte sur le verbe infinitif les suivant.
4. Regroupement des morphèmes "*ne*" et "*pas*" ou "plus",… et annotation du verbe sur lequel porte la négation par NEGTOT ou NEGPAR.

Exemples :

"… il a dit qu'elle *ne* viendrait *pas* de l'Allemagne fédérale."

→ "… il a dit qu'elle [ne pas] [viendrait]$_{\text{NEGTOT}}$ de l'Allemagne fédérale."

"En outre, les chaînes câblées généralistes *ne* se contentent *plus* des restes…"

→ "En outre, les chaînes câblées généralistes [ne plus] [se contentent]$_{\text{NEGPAR}}$ des restes…"

"Impossible en fait de *ne pas* envisager de négociations, tant les intérêts des deux communes sont liés."

→ "Impossible en fait de [ne pas] [envisager]$_{\text{NEGTOT}}$ de négociations, tant les intérêts des deux communes sont liés."

---

[66] Si le verbe est conjugué avec un auxiliaire, l'on suppose que le regroupement du verbe et de son auxiliaire a été réalisé : "ne" + auxiliaire + morphème négatif + verbe au participe passé → "ne" + [auxiliaire + verbe au participe passé] + morphème négatif

Cette stratégie ne permet pas de traiter le cas où la négation est du type prédicative totale avec absence de la forme "pas" comme dans l'exemple suivant :

> "CBS, ABC et NBC *ne* peuvent donc toucher les bénéfices de programmes qu'ils ont pourtant largement financés et …"
>
> → "CBS, ABC et NBC [ne] [peuvent]$_{\text{NEGTOT}}$ donc toucher les bénéfices de programmes qu'ils ont pourtant largement financés et …"

B-2) Résolution par NEGA avec l'aide d'autres agents

• Cas de la négation restrictive

La négation restrictive n'est pas toujours un cas simple à traiter. En effet, la présence des formes "*ne*" et "*que*" ne suffisent pas à affirmer la présence d'une négation restrictive. En particulier, les connaissances de l'agent Segm sont nécessaires pour déterminer, si dans la phrase la forme "*que*" n'est ou n'est pas un introducteur de propositions.

> Exemples :
>
> 1. "…le fait que les divisions blindées françaises *ne* comptent *que* 200 chars…" (négation restrictive)
> 2. "CBS, ABC et NBC *ne* peuvent donc toucher les bénéfices de programmes *qu*'ils ont pourtant largement financés et …" (négation prédicative totale avec absence de la forme "pas")
> 3. "Une apparente confiance qui reste cependant suspendue à un avis dont personne *ne* sait {quand[67]} alors *que* il sera rendu." (négation explétive)

Dans le premier exemple, l'agent Nega aura raison de reconnaître une négation restrictive. Par contre, dans le second exemple, la phrase comporte une négation prédicative totale dans laquelle la forme "*pas*" est absente ; la forme "*que*" doit alors être analysée par Segm comme un introducteur de propositions. De même, Segm doit analyser la forme "*que*" de la troisième phrase, comme un introducteur de propositions ; et l'agent Nega doit reconnaître que la forme "*ne*" est une négation explétive.

La négation est un phénomène linguistique très complexe, et certains cas peuvent rester ambigus comme dans l'exemple suivant :

> "… les trois dinosaures de la télévision américaine *ne* se partagent *plus que* 60 % de l'audience dans les foyers."
>
> → maintenant, ils se partagent seulement 60 % de l'audience, avant ils en avaient davantage.
>
> → maintenant, ils se partagent plus de 60 % de l'audience, avant ils en avaient seulement 60 %.

---

[67] Lors des régularisations la forme "*quand*" est décomposée en "*alors que*".

III-2-2) Ambiguïtés des niveaux sémantique et pragmatique

Notre objectif, pour ces ambiguïtés des niveaux sémantique et pragmatique est d'exposer pour quelques problèmes, quelques hypothèses quant à leur éventuelle résolution.

III-2-2-1) Ambiguïtés du niveau sémantique

Après une étude de certains types d'ambiguïtés sémantiques, nous étudions leurs répercussions dans TALISMAN II (en supposant qu'un niveau d'analyse sémantique soit implanté).

A) Présentation

• Présentation générale de la sémantique

La sémantique vise à la compréhension du sens d'un énoncé et permet sa représentation. La théorie sémantique doit rendre compte des règles générales conditionnant l'interprétation des énoncés.

Il existe un modèle très répandu du signe linguistique appelé "triangle sémiotique" (pour plus de détail se référer à [Lyons, 77]) ; ce modèle représente tout signe linguistique comme composé de trois éléments :

- le signifiant est la suite de graphèmes qui constitue l'identité matérielle de l'unité,
- le référent est l'objet de l'univers auquel renvoie l'unité linguistique,
- le signifié est le concept abstrait associé au signifiant.

La sémantique a pour objet l'étude des signifiés ; elle s'intéresse aux concepts et non pas aux objets du monde.

Le niveau sémantique est plus complexe à décrire et à formaliser que les précédents. La difficulté d'intégration de ce niveau d'analyse explique le fait que les systèmes qui comprennent un niveau sémantique sont souvent dépendants d'une vision particulière de la sémantique. La plupart des systèmes construisent la représentation sémantique en s'appuyant sur l'analyse syntaxique. Ces systèmes prennent, en général, pour unité d'analyse la phrase.

Nous sommes "*encore loin de savoir construire en grandeur réelle des analyseurs sémantiques généraux qui couvriraient la totalité de la langue et seraient indépendants d'un domaine d'application particulier*" [Fuchs & Victorri, 93].

Une présentation de différentes théories sémantiques orientées vers la logique et la cognition est exposée dans [Fuchs & Le Goffic, 92].

• Présentation des travaux de l'équipe en sémantique

Des travaux sont en cours [Manès-Gallo & Rouault, 96, 97], afin de construire un modèle de sémantique énonciative à large couverture linguistique pour le traitement automatique de la langue. La sémantique tente de déterminer la valeur d'un énoncé en termes de type de procès et d'opérations d'énonciations. Les auteurs ont défini une catégorisation opérationnelle des types d'énoncés. Cette catégorisation des types d'énoncés en fonction des types de procès proposée [Manès-Gallo & Rouault, 92], considère que le signifié d'un verbe peut être soit un processus non résultatif (PNR), soit un processus résultatif (PR), soit un état (E).

Exemples issus des travaux de Maria-Catarina Manès-Gallo et de Jacques Rouault :

PR : "le musicien dépose un perroquet" (le résultat est "un perroquet est déposé")

PNR : "le policier observe le canon à eau" (processus n'engendrant pas de résultat)

E : "le propriétaire prête sa radio" (établissement d'un constat)

• Ambiguïtés liées à la polysémie

Ce type d'ambiguïté nous semble difficilement résoluble au niveau sémantique ; des connaissances sur le contexte du discours peuvent aider à lever l'ambiguïté.

Exemple : "cette livre"

où "livre" renvoie à une unité de mesure qui peut être : une monnaie ou un poids

entrée dictionnaire livre * lif*re
F livre *FEM* SNG NOM FOR COM
(F livre MAS SNG NOM FOR COM)
(V livrer P13 ISU PST FOR ORD)
(V livrer PE2 IPF PST FOR ORD)

• Ambiguïtés sur la fonction des syntagmes

Nous étudierons ce type d'ambiguïté avec des indications sur les schémas syntaxiques des verbes (présentés au chapitre 3) car ce sont des ambiguïtés de syntaxe qui se posent au niveau sémantique.

Les exemples suivants sont identiques dans la mesure où ils présentent une ambiguïté sémantique similaire. Le système doit trouver chaque fois deux solutions : une première solution avec rattachement du complément au verbe en tant que C3, et une seconde solution par lexicalisation où les expressions seraient enregistrées (comme "*tomber dans le panneau*").

Exemples :

1.1 "Les unités militaires se sont trouvées *sous* le feu des terroristes et…"

1.2 "…puisqu'elles seront placées *sous* la responsabilité d'un animateur de la Ville."

2.1. "La communauté musulmane est bien sûr indignée par les mesures discriminatoires qui la visent, mais de là à *tomber dans le panneau* tendu par certains esprits retardataires qui voudraient la transformer en une affaire d'Etat, pour ma part, je m'y refuse."

> 2.2 "Devant le palais de la République, les manifestants sont *tombés*, comme par hasard, *sur Mr Günter Schabowski*, membre du bureau politique."

• Ambiguïtés structurelles de rattachement

Nous entendons par ambiguïtés structurelles de rattachement les ambiguïtés liées à l'identification et à la restitution des prédicats et des arguments.
Ainsi, l'exemple suivant présente une ambiguïté de restitution d'une prédication.

> Exemple :
>
> "C. Barberian chantait Mozart aussi bien que les Beatles."
>
> → C. Barberian chantait Mozart aussi bien que les Beatles chantaient Mozart.
>
> → C. Barberian chantait Mozart aussi bien qu'elle chantait les Beatles.

D'autres types d'ambiguïtés relevant du niveau sémantique sont présentés dans [Fuchs, 96].

B) Résolution

Le modèle sémantique proposé [Manès-Gallo & Rouault, 96] n'a pas encore été implémenté. Nous pouvons cependant émettre des hypothèses quant à la résolution de certaines ambiguïtés dans TALISMAN II.

• Exemple de résolution par les agents sémantiques : certaines ambiguïtés sur la fonction des syntagmes

L'agent Synt, grâce à ses connaissances sur les constructions syntaxiques que peuvent admettre chaque verbe, reconnaît la fonction de certains syntagmes régis par les verbes. Des cas d'ambiguïtés sur la fonction des syntagmes pourront être résolues par le ou les agents du niveau sémantique. Dans l'exemple suivant, nous supposons qu'aucune lexicalisation de l'expression "*se trouver sous le feu de…*" n'existe. Le complément sera analysé comme un complément de localisation C3.

> Exemple :
>
> "[Les unités militaires]C0 [se sont trouvées] [sous le feu des terroristes]C3 et…"

• Exemple de résolution par coopération : d'autres ambiguïtés sur la fonction des syntagmes

Par contre dans l'exemple suivant, nous pensons que l'ambiguïté persiste au niveau sémantique. Seules des connaissances du niveau pragmatique (qui se traduisent par la lexicalisation d'expressions) permettront de désambiguïser la fonction du syntagme et d'opter, dans ce contexte, pour l'expression "*tomber dans le panneau*".

> Exemple :
>
> "La communauté musulmane est bien sûr indignée par les mesures discriminatoires qui la visent, mais de là à [*tomber dans le panneau*] tendu par certains esprits retardataires qui

> voudraient la transformer en une affaire d'Etat, pour ma part, je m'y refuse."

III-2-2-2) Ambiguïtés du niveau pragmatique

Nous nous limitons à l'étude de certaines ambiguïtés pragmatiques liées à l'application ; des ambiguïtés de pragmatique linguistique sont présentées dans [Fuchs, 96].

A) Présentation

• Présentation générale de la pragmatique

Le niveau pragmatique est le niveau maximal de représentation de l'analyse linguistique d'un texte. L'aspect pragmatique du langage s'intéresse aux caractéristiques de l'utilisation du langage (motivations psychologiques des locuteurs, types socialisés de discours, objets du discours,…). Le but de la pragmatique est de rendre compte du fonctionnement des énoncés dans les circuits de communication. Le modèle pragmatique doit assurer la gestion de l'univers du discours ; il s'appuie sur des informations de nature sémantique.

Différentes théories pragmatiques sont exposées dans [Moeschler, 89], [Fuchs & Le Goffic, 92].

• Présentation des travaux de l'équipe en pragmatique

Au sein de l'équipe des travaux en collaboration avec A. Berrendonner sont en cours [Berrendonner & al., 92]. Une des préoccupations du modèle proposé est la gestion des individus du discours, de l'évolution de la mémoire "discursive" et des raisonnements opérés par le discours sur les objets. Une application directe est la résolution des anaphores (dont les anaphores associatives) ; il s'agit en fait, d'une représentation des connaissances véhiculées par le discours. L'implantation d'un formalisme formé de "réseaux" d'objets a donné naissance au système orienté objet SAPHIR [Fredj, 93]. Les réseaux d'objets se révèlent un moyen puissant et riche pour représenter des connaissances liées à des textes en langue naturelle ainsi que les raisonnements mis en jeu. Actuellement des recherches sont effectuées pour assurer la cohérence de la base de connaissances représentant les objets du discours [Siameur & Rouault, 96], [Siameur, 97].

• Ambiguïtés pragmatiques liées à l'application

Les exemples suivants montrent qu'une même forme peut, suivant le contexte, se référer à des objets différents. Dans le premier exemple, la forme "*piste*" renvoie à une route alors que dans le second elle renvoie à une orientation d'enquête, des soupçons.

> Exemples :
>
> 1. "Tandis qu'Alain Prost sortait de sa monoplace et abandonnait, Ayrton Senna faisait appel aux *commissaires de piste* pour le pousser et l'aider à redémarrer. Une manoeuvre autorisée par le règlement dans le seul cas où la voiture reste immobilisée sur la ***piste*** et constitue donc un danger pour les autres pilotes."
>
> 2. "Mr Habré a enfin refusé d'accorder crédit à certaines rumeurs impliquant tel ou tel pays étranger dans l'attentat

> contre un DC-10 de la compagnie française UTA qui a fait 271 morts le 19 septembre dernier. "Pour l'heure, rien, je dis bien rien, ne permet de privilégier telle ou telle ***piste***."

B) Résolution

Bien que le niveau d'analyse pragmatique soit en cours d'élaboration, nous pouvons émettre quelques hypothèses sur la résolution éventuelle d'ambiguïtés pragmatiques liées à l'application.

• Ambiguïtés pragmatiques liées à l'application

Le ou les agents du niveau pragmatique seront peut être en mesure de résoudre l'ambiguïté suivante sur la signification de la forme "*piste*" (une route ou une orientation d'enquête) ; pour cela, ils utiliseront le contexte du discours et des connaissances sur le "monde". Si ces connaissances ne sont pas suffisantes, des connaissances sur l'application seront nécessaires.

Mais il nous semble que toutes les ambiguïtés ne pourront être résolues par le système (en particulier, les ambiguïtés pragmatiques), c'est pourquoi un recours à l'utilisateur ou à un perfectionnement du système sera alors nécessaire.

CONCLUSION GENERALE

L'origine de notre recherche est l'architecture multi-agents TALISMAN. L'objectif principal de nos travaux était le développement, en grandeur réelle, de l'architecture proposée : un système d'analyse morpho-syntaxique bénéficiant d'une architecture distribuée. La production d'ambiguïtés inhérente à tout système général d'analyse de la langue provoque dans un système multi-agents des conflits linguistiques que nous avons étudiés.

Nous avons été amené à développer une seconde version de l'architecture nommée TALISMAN II. Nous avons présenté de nouvelles caractéristiques de ce système :

- La communication dans un système multi-agents : nous avons proposé des protocoles de communication permettant aux agents linguistiques du système de bénéficier des connaissances et/ou compétences des autres agents.
- La résolution interne de solutions multiples : d'une part, au niveau morphologique à l'aide de règles linguistiques contextuelles de levée d'ambiguïtés ; et d'autre part, au niveau syntaxique par la définition de règles d'élimination de trajectoires impossibles.
- Une analyse syntaxique mixte :

Un analyseur syntaxique descendant de la proposition assertive (supportant la définition de règles récursives à gauche). Et, une analyse syntaxique ascendante à partir d'indices pertinents du texte (présence d'un point d'interrogation, d'un coordonnant,…). Cela permet la mise en place de stratégies spécifiques (ou heuristiques linguistiques) en fonction de certains indices présents dans le texte : (1) une grammaire de la coordination activée par la présence d'un coordonnant, (2) la coopération entre les agents Morph(ologie) et Synt(axe) indispensable lors de la présence d'une forme inconnue.

Pour ce qui est des perspectives immédiates de notre travail, citons :

- L'intégration des stratégies de traitements de phénomènes linguistiques complexes tels les agents Transf(ormation) et Ellip(se) que nous avons modélisés.
- L'amélioration des capacités de raisonnement des agents notamment pour les prises de décisions locales et la gestion de leur mémoire locale.

A moyen terme, nous envisageons une validation de nos travaux par (cf. chapitre VI) :

- L'application de TALISMAN II à l'indexation automatique de gros documents techniques.
- L'utilisation de l'architecture et des protocoles de communication pour un système de génération automatique de textes.
- L'intégration d'un analyseur morphologique dérivationnel.

A plus long terme, les perspectives en traitement automatique des langues qui s'offrent à TALISMAN II sont multiples, citons :

- Le développement des agents relevant des niveaux sémantique et pragmatique.
- L'intégration d'agents traitant d'autres phénomènes linguistiques complexes (comme la négation, l'anaphore).

- L'intégration d'une composante robustesse permettant, notamment la détection et la correction d'erreurs.
- L'étude de portabilité de TALISMAN II à une autre langue : le portugais (dans le cadre du projet brésilien Nalamas en cours de réalisation [Nalamas, 97]).

BIBLIOGRAPHIE

[AGNEL, 97] Estelle Agnel, "*Le dictionnaire, un composant clef des systèmes de TAL : modélisation relationnelle et implantation sous Access d'un specimen du dictionnaire Cristal*". Rapport de DEA en Sciences de l'Information et de la Communication, Equipe Cristal-Gresec, Université Stendhal, Grenoble3, Juin 1997.

[AHO & CORASICK, 75] A.V. Aho et M.J. Corasick, "*Efficient string matching : An aid to bibliographic search*". Communications of the ACM, Volume 18, N°6, Juin 1975.

[AHO & ULLMAN, 72] Alfred Aho et Jeffrey Ullman, "*The theory of parsing, translation and compiling*", Volume 1 : Parsing. Prentice-Hall Inc., Englewood Cliffs, Bell Telephone Laboratories, New Jersey, 1972.

[AMSILI & HATHOUT, 96] Pascal Amsili et Nabil Hathout, "*Computational semantics of time/negation interaction*". International Conference on COmputational LINGuistics (COLING'96). University of Copenhagen, Danemark, 05-09 Août 1996.

[ANTOINE, 94] Jean-Yves Antoine, "*Coopération syntaxe-sémantique pour la compréhension automatique de la parole spontanée*". Thèse de Doctorat, Institut de la Communication Parlée, Institut National Polytechnique de Grenoble, 12 Décembre 1994.

[ANTONIADIS & al., 96] Georges Antoniadis, Laurence Balicco, Claude Ponton, Marie-Hélène Stéfanini et Karine Warren, "*Vers une approche distribuée pour la génération automatique de textes*". Colloque Traitement Automatique du Langage Naturel (TALN'96). Campus du CNRS, Marseille, 22-24 Mai 1996.

[ARRIVE & al., 86] Michel Arrivé, Françoise Gadet, Michel Galmiche, "*La grammaire d'aujourd'hui. Guide alphabétique de linguistique française*". Flammarion, Paris, 1986.

[AUSTIN, 62] John Austin, "*How to do things with words*". Oxford, Clarendon Press, 1962.

[BACHIMONT, 90] Bruno Bachimont, "*Cohérence et convergence dans un tableau noir : organisation, formalisation et sémantique de l'architecture de contrôle ABACAB*". Thèse de Doctorat de l'Université Paris VI, spécialité Informatique, Décembre 1990.

[BALICCO, 93] Laurence Balicco, "*Génération de répliques en français dans une interface homme-machine en langue naturelle*". Thèse de Doctorat, Université Pierre Mendès-France, Grenoble, 1993.

[BAUJARD & GARBAY, 90] O. Baujard et C. Garbay, "*A programming environment for distributed expert system design*". Expert System Applications, ExperSys, 1990.

[BERON & al., 95] Frédéric Béron, Bernard Carpuat, Pierre Glize et Christine Régis, "*La résolution de conflit sans négociation : modèle et évaluation*". Troisièmes Journées IAD&SMA, Chambéry, St Baldoph, Mai 1995.

[BERRENDONNER, 83 ou 90] Alain Berrendonner, "*Grammaire pour un analyseur : aspects morphologiques*". Ecole d'été de la DBMIST, Juillet 1983. 2ème édition : Les cahiers du CRISS N°15; Novembre 1990.

[BERRENDONNER, 89] Alain Berrendonner, "*Appendice au calcul des noms*". Rapport interne, 1989.

[BERRENDONNER, 90] Alain Berrendonner, "*Variations sur l'impersonnel*". Maillard éd., Actes du colloque de Grenoble, Mai 1990.

[BERRENDONNER & al., 80] Alain Berrendonner, Bouché, Le Guern, Jacques Rouault, "*Pour une méthode d'interaction pondérée des composants morphologique et syntaxique en analyse automatique du français*". T.A Informations, N°1, 1980.

[BERRENDONNER & al., 92] Alain Berrendonner, Mounia Fredj, Flavio Oquendo et Jacques Rouault, "*Un système inférentiel orienté objet pour des applications en Langue Naturelle*". Proceedings of the 14th International Conference on Computational Linguistics (COLING'92), pp. 461-467, Nantes, 23-28 Juillet 1992.

[BERTHET, 91] Berthet, "*Protocoles de communication*". Mémoire de DEA, ENSIMAG, Juin 1991.

[BERTIER & LALLICH, 97] Marc Bertier et Geneviève Lallich-Boidin, "*Etiquetage grammatical : réflexions issues de notre participation à l'évaluation GRACE*". Quatrième colloque du Traitement Automatique du Langage Naturel (TALN'97), 12-13 Juin 1997, Grenoble.

[BERTIER & al., 97] Marc Bertier, Georges Antoniadis, Christel Froissart, Geneviève Lallich-Boidin, Jacques Rouault, Lahcène Si Ameur et Ismaïl Timimi. Contrat de l'équipe CRISTAL-GRESEC N° 94 K 6427 d'aide à la recherche intitulé "*Indexation de gros documents*" achevé en 1997.

[BINOT & al., 88] J.L. Binot, B. Demoen, K. Hanne, L. Solomon, Y. Vasiliou, W. Von Hahn et T. Wachtel, "*LOQUI : A logic oriented approach to data and knowledge bases supporting natural language interaction*". Proceedings of the ESPRIT 88 Conference, North-Holland, 1988.

[BLANK, 87] Ingeborg Blank, "*Etudes des constructions syntaxiques du verbe en vue d'un traitement automatique*". Les Cahiers du CRISS N°11, Grenoble, 1987.

[BLANK & PALERMITI, 87] Ingeborg Blank et Rosalba Palermiti, "*Rapport sur le comportement syntaxique des verbes de la langue française*". Les Cahiers du CRISS N°11, Grenoble, 1987.

[BOITET & SELIGMAN, 94] Christian Boitet et Mark Seligman, *"The "Whiteboard" architecture : a way to integrate heterogeneous comonents of NLP systems*". International Conference on COmputational LINGuistics (COLING'94), Kyoto, Japon, 5-9 Août 1994.

[BOITET & TOMOKIYO, 96] Christian Boitet et Mutsuko Tomokiyo, "*Theory and practice of ambiguity labelling with a view to interactive disambiguation in text and speech MT*". International Conference on COmputational LINGuistics (COLING'96), University of Copenhagen, Danemark, 05-09 Août 1996.

[BOND & GASSER, 88] Bond et Les Gasser, "*Readings in DAI*". Morgan Kaufmann, 1988.

[BOS, 94] Johan Bos, "*Presupposition & VP-Ellipsis*". International Conference on COmputational LINGuistics (COLING'94). Kyoto, Japon, 5-9 Août 1994.

[BOUCHAFFRA, 92] Djamel Bouchaffra, "*Echantillonage multivarié de textes pour les processus de Markov et introduction au raisonnement incertain pour le traitement de la langue naturelle*". Thèse de Doctorat, Université Pierre Mendès France, Grenoble, Novembre 1992.

[BOUCHAFFRA & al., 93] Djamel Bouchaffra, Geneviève Lallich-Boidin et Jacques Rouault, "*Echantillonage stratifié avec restimation ("bootstrap") : application à l'analyse morphologique*". Secondes journées internationales d'analyse statistique de données textuelles, Montpellier, 21 et 22 Octobre 1993.

[BOURON, 93] Thierry Bouron, "*Structures de communication et d'organisation pour la coopération dans un univers multi-agents*". Thèse de Doctorat, LIFIA, Février 1993.

[BRATKO, 88] Ivan Bratko, "*Programmation en PROLOG pour l'intelligence artificielle*". Traduit par Laurent Ricard. InterEditions, 1988.

[BRAUNWARTH & al., 93] Michel Braunwarth, Abdelouahab Mekaouche et Jean-Claude Bassano, "*Utilisation d'un tableau noir dynamique dans un système de recherche documentaire*". In 13th International Conference on Natural Language Processing, pp. 33-44, Avignon, EC2, may, 1993.

[BRUN, 96] Caroline Brun, "*La coordination dans le cadre d'une grammaire LFG du français*". Colloque Etudiant de Linguistique Informatique de Montréal (CLIM'96), Université du Quebec à Montréal, Canada, 08-11 Juin 1996.

[BURMEISTER & al., 93] Birgit Burmeister, Afsaneh Haddadi et Kurt Sundermeyer, "*Generic, Configurable, Cooperation Protocols for Multi-Agent Systems*". In Lecture Notes in Artificial Intelligence 957 ; In the 5th European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW'93), Neuchâtel, Suisse, Août 1993.

[CAILLAUD & al., 94] Bertrand Caillaud, Jean-Yves Antoine, Jean Caelen et Geneviève Caelen-Haumont, "*MICRO : un système multi-agents pour la compréhension de la parole*". 9ème Congrès RFIA, AFCET, Paris, 11-14 janvier 1994.

[CAMPBELL & D'INVERNO, 90] John Campbell et Mark D'Inverno, "*Knowledge interchange protocols*". In Yves DEMAZEAU et Jean-Pierre MÜLLER, editors, Decentralized artificial intelligence. Volume I. Elsevier science pub., North Holland, 1990.

[CARBONELL & BROWN, 88] Jaime carbonell et Ralf Brown, "*Anaphora resolution: a multi-strategy approach*". International Conference on COmputational LINGuistics (COLING'88). Budapest, 22-27 Août 1988.

[CHANG & WOO, 92] Man Kit Chang et Carson Woo, "*SANP : A communication level protocol for negotiations*". In Eric Werner & Yves DEMAZEAU editors, Decentralized Artificial Intelligence 3. Volume III, Elsevier science pub., North Holland, 1992.

[CHANOD & TAPANAINEN, 95] Jean-Pierre Chanod et Pasi Tapanainen, "*Tagging French - Comparing a statistical and a constraint-based method*". Proceedings of the seventh conference of the Europan Chapter of the ACL (EACL'95), p.149-156, Association for Computational Linguistics, Dublin, 1995.

[CHEVRIER, 93] Vincent Chevrier, "*Etude et mise en oeuvre du paradigme multi-agents : de Atome à Gtmas*". Thèse de Doctorat de l'Université de Nancy I, Juin 1993.

[CLAVIER, 96a] Viviane Clavier, "*Modélisation de la suffixation pour le Traitement Automatique du français écrit. Application à la recherche d'information*". Thèse de Doctorat, Université Stendhal, Janvier 1996.

[CLAVIER, 96b] Viviane Clavier, "*Morphologie et reconnaissance des mots inconnus en TAL : compte-rendu de l'évaluation Grâce*". Actes GDR-PRC Séminaire Lexique. Représentations et outils pour les bases lexicales, morphologie robuste. 13-14 Novembre 1996, Grenoble, pp. 97-106.

[CLAVIER & al., 96] Viviane Clavier, Karine Warren, Geneviève Lallich-Boidin et Marie-Hélène Stéfanini, "*Intégration de la morphologie dérivationnelle dans un système distribué d'analyse du français écrit*". Colloque Informatique & Langue Naturelle (ILN'96), Université de Nantes, 09-10 Octobre 1996.

[COCHARD & FROIDEVAUX, 95] Jean-Luc Cochard et Philippe Froidevaux, "*Environnement multi-agents de reconnaissance automatique de la parole en continu*". Troisièmes Journées IAD&SMA, Chambéry, St Baldoph, 1995.

[COHEN & LEVESQUE, 90] Philip Cohen et Hector Levesque, "*Intention is choice with commitment*". Artificial Intelligence, Volume 42, N°3, Mars 1990.

[COLINEAU & MOULIN, 96] Nathalie Colineau et Bernard Moulin, "*Un modèle connexionniste pour la reconnaissance d'actes de dialogue*". Colloque Informatique & Langue Naturelle (ILN'96), Université de Nantes, 09-10 Octobre 1996.

[DAMI & LALLICH, 91] Samir Dami et Geneviève Lallich-Boidin, "*An expert system for french analysis within a multi-mode dialogue to be connected*". In Intelligent Text and Image Handling, Colloque RIAO'91, Barcelone, 02-05 Avril 1991.

[DAMI & al., 92] Samir Dami, Flavio Oquendo et Marie-Hélène Stéfanini, "*Analyse du français dans un univers multi-agents*". Notes et rapports de recherche N°31 du CRISS, Juin 1992.

[DEMAZEAU, 95] Yves Demazeau, "*From cognitive interactions to collective behaviour in agent-based systems*". European Conference on Cognitive Science, Saint-Malo, Avril 1995.

[DEMAZEAU & MÜLLER, 90] Yves Demazeau et Jean-Pierre Müller, "*Decentralized artificial intelligence. Introduction*". In Yves DEMAZEAU et Jean-Pierre MÜLLER, editors, Decentralized Artificial Intelligence, Volume I, Elsevier science publishers, North Holland, 1990.

[DEMAZEAU & al., 94b] Yves Demazeau, Olivier Boissier et Jean-Luc Koning, "*Using Interaction Protocols to Control Vision Systems*". IEEE International Conference on Systems, Man and Cybernetics, San Antonio, Octobre 1994.

[DEROSE, 88] Steven DeRose, "*Grammatical category disambiguation statistical optimization*". Computational Linguistics, Volume 14, N°1, 1988.

[DURFEE & MONTGOMERY, 90] Edmund Durfee et Thomas Montgomery, "*A hierarchical protocol for coordinating multiagent behaviors*". In proceedings of the 8th National Conference on Artificial Intelligence (AAAI-90), Boston, 29 Juillet - 3 Août 1990.

[EARLEY, 70] J.C. Earley, "*An efficient context-free parsing algorithm*". Proceedings of the ACM, Février 1970.

[EPHRATI & ROSENSCHEIN, 93] Eithan Ephrati et Jeffrey Rosenschein, "*Multi-agent planning as a dynamic search for social consensus*". In proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI-93), Volume 1, Chambéry, France, 1993.

[ERCEAU & FERBER, 91] Jean Erceau et Jacques Ferber, "*L'Intelligence Artificielle Distribuée*". La Recherche 233, Volume 22, Juin 1991.

[ERMAN & al., 80] Lee Erman, Frederick Hayes-Roth, Victor Lesser et Raj Reddy, "*The Hearsay-II Speech-Understanding System : Integrating Knowledge to Resolve Uncertainty*". In ACM Computing Surveys, Vol.12, N°2, Juin 1980.

[ERLI, 87] "*SAPHIR, Manuel de description du logiciel*". Société ERLI, 1987.

[ESFANDIARI & al., 95] Babak Esfandiari, Joël Quinqueton, Jean-Luc Koning et Yves Demazeau, "*Systèmes multi-agents et gestion de réseaux*". Actes des 5ème journées nationales PRC-GDR Intelligence Artificielle. Nancy, 01-03 Février 1995.

[EYMARD, 92] Gilbert Eymard, "*Traitement documentaire des sommaires : des mots-clés à l'extraction des connaissances. Application à une documentation technique*". Thèse de Doctorat, CRISS, Université Pierre Mendès France, Novembre 1992.

[EYMARD & al., 92] Gilbert Eymard, Yolla Polity, Georges Antoniadis, Laurence Balicco, Alain Berrendonner, Mounia Fredj, Flavio Oquendo, Jacques Rouault, Geneviève Lallich-Boidin et Marie-Hélène Stéfanini, "*Langue naturelle, dialogue, représentation des connaissances : jalons 91*" ; Notes et rapports de recherche n°25 du CRISS, Février 1992.

[FAY-VARNIER & al., 91] Christine Fay-Varnier, Christophe Fouquere, Gilles Prigent et Pierre Zweigenbaum, "*Modules syntaxiques des systèmes d'analyse du français*". Techniques et Science Informatiques (TSI), Volume 10, n°6, 1991.

[FERBER, 90] Jacques Ferber, "*Eco-problem-solving : how to solve problems by interactions*". Rapport N°5 du LAFORIA, Janvier 1990.

[FERBER, 95] Jacques Ferber, "*Les Systèmes Multi-Agents. Vers une intelligence collective*". InterEditions, Paris, 1995.

[FERBER & GASSER, 91] Jacques Ferber et Les Gasser, "*Intelligence Artificielle Distribuée*". Proceeding of the 11th Workshop Tutorial Avignon "Expert systems and their applications", Avignon, 27-31 Mai 1991.

[FINCK, 94] Denis Finck, "*Application des notions d'objet et d'agent à la traduction automatique*". 9ème Congrès Reconnaissance des Formes et Intelligence Artificielle (RFIA'94), AFCET, Paris, 11-14 Janvier 1994.

[FREDJ, 93] Mounia Fredj, "*SAPHIR : un système d'objets inférentiels. Contribution à l'étude des raisonnements en langue naturelle*". Thèse de Doctorat, Université Pierre Mendès France, Grenoble, Janvier 1993.

[FROISSART, 92] Christel Froissart, "*Robustesse et interfaces homme-machine en langue naturelle*". Thèse de Doctorat, CRISS, Université Pierre Mendès France, Grenoble, Décembre 1992.

[FUCHS, 87] Catherine Fuchs, "*L'ambiguïté et la paraphrase en linguistique*". Actes du colloque de Caen sur "L'ambiguïté et la paraphrase", 9-11 Avril 1987.

[FUCHS, 96] Catherine Fuchs, "*Les ambiguïtés du français*". Collection l'essentiel français. Editions Orphys, 1996.

[FUCHS & LE GOFFIC, 92] Catherine Fuchs et Pierre Le Goffic, "*Les linguistiques contemporaines*". Editions Hachette, 1992.

[FUCHS & VICTORRI, 93] Catherine Fuchs et Bernard Victorri, "*Syntaxe*". Chapitre IV de "Linguistique et Traitements Automatiques des Langues", Editions Hachette, 1993.

[FUM & al., 88] Danilo Fum, Givanni Guida et Carlo Tasso, "*A distributed multi-agent architecture for natural language processing*". International Conference on COmputational LINGuistics (COLING'88), Budapest, 22-27 Août 1988.

[GALLIERS, 90] Julia Rose Galliers, "*The positive role of conflict in cooperative multi-agent systems*". In Distributed Artificial Intellience, Y. DEMAZEAU et J.P. Mûller editors, 1990.

[GASPAR, 91] Graça Gaspar, "*Communication and belief changes in a society of agents*". Towards a formal model of autonomous agents. In Yves DEMAZEAU et Jean-Pierre MÜLLER, editors, Decentralized artificial intelligence, Volume II, Elsevier science publishers, North Holland, 1991.

[GAZDAR & al., 85] G. Gazdar, E. Klein, G.K Pullum et I. Sag, "*Generalized Phrase Structure Grammar*". Harvard University Press, Cambridge, 1985.

[GENESERETH & KETCHPEL, 94] Michael Genesereth et Steven Ketchpel, "*Software Agents*". Communications of the ACM, Vol.37, No.7, Juillet 1994.

[GENTHIAL & al., 96] Damien Genthial, Jacques Courtin et Jacques Ménézo, "*Distributing and porting general linguistic tools*". International Conference on COmputational LINGuistics (COLING'96), University of Copenhagen, Danemark, 05-09 Août 1996.

[GLEIZES & GLIZE, 90] Marie-Pierre Gleizes et Pierre Glize, "*Les systèmes multi-experts*". Technologie de pointe, N°38, édition Hermès, 1990.

[GROSS, 90] Gaston Gross, "*Les mots composés*". Modèles linguistiques, XII:1, Lille, p.47-63, 1990.

[GUIN & al., 93] Dominique Guin, Sophie Billet-Coat, Philippe Reitz, Danièle Hérin-Aimé, "*Protocole comportemental de l'interaction didactique entre un agent artificiel et un agent humain*". PRC-GDR Intelligence Artificielle, GR Didactique, "Environnements interactifs d'apprentissage avec ordinateur". 3ème Journées EIAO de Cachan, Eyrolles, Paris, 3-5 février 1993.

[HATON, 89] Jean-Paul Haton, "*Panorama des systèmes multi-agents*". Dans "Architectures avancées pour l'intelligence artificielle". Onzième journées francophones sur l'informatique, EC2 éditeur, Janvier 1989.

[HATON & al., 91] Jean-Paul Haton, Nadjet Bouzid, François Charpillet, Marie-christine Haton, Brigitte Lâasri, Hassan Lâasri, Pierre Marquis, Thierry Mondot et Amedeo Napoli "*Le raisonnement en intelligence artificielle. Modèles, techniques et architectures pour les systèmes à bases de connaissances*". InterEdition, Paris, 1991.

[IBEKWE-SANJUAN, 97] Fidélia Ibekwe-Sanjuan, "*Recherche des tendances thématiques dans les publications scientifiques. Définition d'une méthodologie fondée sur la linguistique*". Thèse de Doctorat en Sciences de l'Information et de la Communication, Université Stendhal, Grenoble3, le 17 Septembre 1997.

[KAC & RINDFLESCH, 88] Michael Kac et Thomas Rindflesch, "*Coordination in reconnaissance-attack parsing*". International Conference on COmputational LINGuistics (COLING'88), Budapest, 22-27 Août 1988.

[KALLAS, 87] Ghassan Kallas, "*Résolution des solutions multiples en analyse morphologique automatique des langues naturelles. Utilisation des modèles de Markov*". Thèse de Doctorat, CRISS, Université Pierre Mendès-France, Grenoble2, Juin 1987.

[KAPLAN & BRESNAN, 82] R. Kaplan et J. Bresnan, "*Lexical Functional Grammar : A formal system for grammatical representation*". The mental representation of grammatical relations, Bresnan ed., MIT Press, Cambridge, 1982.

[KENNEDY & BOGURAEV, 96] Christopher Kennedy et Branimir Boguraev, "*Anaphora in a wider context: tracking discourse referents*". In proceedings of the 12th European Conference on Artificial Intelligence (ECAI'96), Budapest, Hungary, 11-16 Août 1996.

[KOSKENNIEMI, 83] K. Koskenniemi, "*Two-level morphology : a general computational model for word-form recognition and production*". PhD Thesis, University of Helsinki, 1983.

[KREIFELTS & VON MARTIAL, 91] Thomas Kreifelts et Frank Von Martial, "*A negotiation framework for autonomous agents*". In Yves DEMAZEAU et Jean-Pierre MÜLLER, editors, Decentralized artificial intelligence, Volume II, Elsevier science publishers, North Holland, 1991.

[KUNO & OETTINGER, 62] S. Kuno et A.G. Oettinger, "*Multiple-path syntactic analyser*". Information Processing (IFIP Cong.), Popplewell ed., North-Holland, Amsterdam, pp. 306-311, 1962.

[LACOMBE, 96] Daniel Lacombe, "*Comment chercher l' "antécédent" d'un pronom personnel ?*". Colloque Informatique & Langue Naturelle (ILN'96), Université de Nantes, 09-10 Octobre 1996.

[LALLICH & al., 90] Geneviève Lallich-Boidin, Gérard Henneron et Rosalba Palermiti, "*Analyse du français : achèvement et implantation de l'analyseur morpho-syntaxique*". Les cahiers du CRISS N°16, Novembre 1990.

[LANDER & LESSER, 93] Susan Lander et Victor Lesser, "*Understanding the role of negotiation in distributed search among heterogeneous agents*". In proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI-93), Volume 1, Chambéry, France, 1993.

[LAPPIN & SHIH, 96] Shalom Lappin et Hsue-Hueh Shih, "*A generalized reconstuction algorithm for ellipsis resolution*". International Conference on COmputational LINGuistics (COLING'96), University of Copenhagen, Danemark, 05-09 Août 1996.

[LASAUNIERE, 94] Christaine Lasauniere, "*La reconnaissance du vocabulaire de spécialité*". Mémoire de DEA en Sciences de l'Information et de la Communication, Université Stendhal, 1994.

[LAVELLI & STOCK, 90] Alberto Lavelli et Oliviero Stock, "*When something is missing: Ellipsis, Coordination and the chart*". In proceedings of ". International Conference on COmputational LINGuistics (COLING'90), Helsinki, Août 1990.

[LEBOURG, 88] Valérie Lebourg, "*Modèle linguistique pour la levée automatique des ambiguïtés morphologiques*". Mémoire de DEA en Sciences de l'information, CRISS, Université de Grenoble II, 1988.

[LESNIEWSKI, 89] St. Lesniewski, "*Sur les fondements de la mathématique*". Traduit du polonais par Kalinowski, Hermès, Paris, 1989.

[LETELLIER, 93] Sabine Letellier, "*ECLAIR, un système d'analyse et de correction lexicales multi-experts et multi-lexiques*". Thèse de Doctorat, Université de Paris-Sud, Centre d'Orsay, Décembre 1993.

[LYONS, 77] John Lyons, "*Semantics*", Volume I. Cambridge University Press, 1977.

[MADEC, 96] Henri Madec, "*Simulation d'apprentissage syntaxique par architecture à base neuronale*". Actes du 6ème Colloque Association pour la Recherche Cognitive ; "Sciences Cognitives, Individus et Sociétés". Villeneuve d'Ascq, 10-12 Décembre 1996.

[MAEGAARD & SPANG HANSSEN, 78] B. Maegaard et E. Spang Hanssen, "*Segmentation automatique du français écrit*". Documents de linguistique quantitative, Dunod, 1978.

[MANDIAU, 93] René Mandiau, "*Contribution à la modélisation des Univers Multi-agents : génération d'un plan partagé*". Thèse de Doctorat de l'Université de Valenciennes, Février 1993.

[MANES-GALLO & ROUAULT, 92] Maria-Caterina Manès-Gallo et Jacques Rouault, "*Schémas d'action et types de procès*". Intellectica 1992.

[MANES-GALLO & ROUAULT, 95] Maria-Caterina Manès-Gallo et Jacques Rouault, "*Connaissances véhiculées par les verbes dans le discours : les schémas d'action*". In Verbum, Université de Nancy II, 1995.

[MANES-GALLO & ROUAULT, 96] Maria-Caterina Manès-Gallo et Jacques Rouault, "*Un modèle pour le traitement sémantique des énoncés*". Colloque Informatique & Langue Naturelle (ILN'96), Université de Nantes, 09-10 Octobre 1996,.

[MANES-GALLO & ROUAULT, 97] Maria-Caterina Manès-Gallo et Jacques Rouault, "*Rôles sémantiques et schémas d'énoncés*". Colloque Traitement Automatique du Langage Naturel (TALN'97), Grenoble, 12-13 Juin 1997.

[MARUICHI & al., 90] Takeo Maruichi, Masaki Ichikawa et Mario Tokoro, "*Modeling autonomous agents and their groups*". In Yves DEMAZEAU et Jean-Pierre MÜLLER, editors, Decentralized Artificial Intelligence, Volume I, Elsevier science publishers, North Holland, 1990.

[MATSUBAYASHI & TOKORO, 93] Kei Matsubayashi et Mario Tokoro, "*A collaboration mechanism on positive interactions in multi-agent environments*". In proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI-93), Volume 1, Chambéry, France, 1993.

[MEKAOUCHE & BASSANO, 91] Abdelouahab Mekaouche et Jean-Claude Bassano, "*Analyseur linguistique multi-expert pour la recherche documentaire*". Onzièmes journées Internationales sur les systèmes experts et leurs applications, Avignon, 27-31 Mai 1991.

[MENEZO & al., 96a] Jacques Ménézo, Damien Genthial, Jacques Courtin et Danièle Dujardin, "*La méthode des structures, principe et mise en œuvre dans CELINE, un système multi-agents de détection et correction des erreurs lexicales et syntaxiques*". Colloque Traitement Automatique du Langage Naturel (TALN'96), Campus du CNRS, Marseille, 22-24 Mai 1996.

[MENEZO & al., 96b] Jacques Ménézo, Damien Genthial, Jacques Courtin et Danièle Dujardin, "*Le modèle de l'utilisateur comme modèle linguistiuqe partiel suffisant en détection et correction des erreurs*". Colloque Informatique & Langue Naturelle (ILN'96), Université de Nantes, 09-10 Octobre 1996.

[MINSKY, 91a] Naftaly Minsky, "*The imposition of protocols over open distributed systems*". IEEE Transactions on Software Engineering, Volume 17, N°2, Février 1991.

[MINSKY, 91b] Naftaly Minsky, "*Law-governed systems*". In Software Engineering Journal, Volume 6, N°5, Septembre 1991.

[MMI2, 90] Samir Dami et Geneviève Lallich-Boidin, "*Un système expert d'analyse du français pour un dialogue multi-mode*". Rapport de recherche du CRISS, Université Pierre Mendès-France, Grenoble, Octobre 1990.

[MOESCHLER, 89] Jacques Moeschler, "*Marques linguistiques, interprétation pragmatique et conversation*". Les Cahiers de Linguistique Française N°10 : "En deçà et au-delà de l'analyse du discours." Université de linguistique française, Genève, 1989.

[NALAMAS, 97] Projet NALAMAS (PROTEM-III-CNPQ) coordonné par Vera Lucia Strube de Lima. Rapport d'activité de Septembre 1996 à Août 1997.

[OSAWA, 93] Ei-Ichi Osawa, "*A scheme for agent collaboration in open multi-agent environments*". In proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI-93), Volume 1, Chambéry, France, 1993.

[PAGANELLI, 97] Céline Paganelli, "*La recherche d'information dans les bases de documents techniques en texte intégral. Etude de l'activité des utilisateurs*". Thèse de Doctorat en Sciences de l'Information et de la Communication, Université Stendhal, Grenoble, Juillet 1997.

[PALERMITI, 92] Rosalba Palermiti, "*Les anaphores dans le dialogue homme-machine en langue naturelle écrite : Données recueillies par Magicien d'Oz, dans une situation de recheche documentaire*". Thèse de Doctorat, Université Pierre Mendès-France, Grenoble, Juillet 1992.

[PEARL, 90] Judea Pearl, "*Heuristiques. Stratégies de recherche intelligente pour la résolution de problèmes par ordinateur*". Reproduction de l'ouvrage américain "Heuristics. Intelligent search strategies for computer problem solving". CEPAD, 1990.

[PITRAT, 90] Jacques Pitrat, "*Métaconnaissance. Futur de l'intelligence artificielle*". Editions Hermès, Paris, 1990.

[PLEIAD, 92] Groupe PLEIAD (J.Y. Antoine, B. Caillaud, O. Boissier, Y. Demazeau, J. Sichman, D. Ziebelin, O. Baujard, S. Pesty, M.H. Stéfanini et M. Chaillot), "*Définition taxinomique du vocabulaire utilisé en Intelligence Artificielle Distribuée*". Journées du PRC IA sur les systèmes multi-agents, Nancy, 18 décembre 1992.

[POMPIDOR & VERGNAUD, 95] Pierre Pompidor et Jean-François Vergnaud, "*Coopération et révision des agents d'un système coopératif gérant des bases de données. Application à la traduction automatique du chinois dans un but pédagogique*". Troisièmes Journées IAD&SMA, Chambéry, St Baldoph, 1995.

[PONTON, 96] Claude Ponton, "*Génération automatique de textes en langue naturelle. Essai de définition d'un système noyau*". Thèse de Doctorat en Informatique et Communication, Université Stendhal, Grenoble3, le 26 Novembre 1996.

[POPULAIRE & al., 93] Philippe Populaire, Yves Demazeau, Olivier Boissier, Jaime Sichman, "*Description et implémentation de protocoles de communication en univers multi-agent*". Premières Journées Francophones IAD&SMA, Toulouse, Avril 1993.

[RADY, 83] Mohamed Rady, "*L'ambiguïté du langage naturel est-elle la source du non-déterminisme des procédures de traitement*". Thèse de Doctorat, Université Pierre et Marie Curie, Juin 1983.

[REBOUL, 89] Anne Reboul, "*Résolution de l'anaphore pronominale : sémantique et/ou pragmatique*". Les Cahiers de Linguistique Française N°10 : "En deçà et au-delà de l'analyse du discours." Université de linguistique française, Genève, 1989.

[REITZ, 92] Philippe Reitz, "*Contribution à l'étude des environnements d'apprentissage. Conceptualisation, spécifications et prototypage*". Thèse de Doctorat de l'Université de Montpellier II- Sciences et Techniques du Languedoc, Février 1992.

[ROSENSCHEIN, 93] Jeffrey Rosenschein, "*Consenting agents : negotiation mechanisms for multi-agent systems*". In proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI-93), Volume 1, Chambéry, France, 1993.

[ROUSSANALY & PIERREL, 92] Azim Roussalany et Jean-Pierre Pierrel, "*Dialogue oral homme-machine en langage naturel : le projet DIAL*". Technique et science informatiques, Volume 11, N°2, 1992.

[RUWET, 70] Nicolas RUWET, "*Introduction à la grammaire générative*". Edition PLON, 1970.

[SABAH, 90a] Gérard Sabah, "*CARAMEL : A Computational Model of Natural Language Understanding Using a Parallel Implemenation*". In Proceedings of the 9th European Conference on Artificial Intelligence, Stockholm, Sweden, 6-10 Août 1990.

[SABAH, 90b] Gérard Sabah, "*CARAMEL : A flexible model for interaction between the cognitive processes underlying natural language understanding*". In Proceedings of the 13th International Conference on COmputational LINGuistics (COLING'96) ; Helsinski, Août 1990.

[SABAH, 90c] Gérard Sabah, "*CARAMEL : Un Système Multi-Experts pour le traitement Automatique des Langues*". Modèles linguistiuqes, Tome XII, Fascicule 1, 1990.

[SAUVAGE, 92] Caroline Sauvage-Wintergerst, "*Parallélisme et traitement automatique des langues, application à l'analyse des énoncés elliptiques*". Thèse de Doctorat de l'Université de Paris-sud, Centre d'Orsay, Mars 1992.

[SCHMUTZ, 95] Laurent Schmutz, "*SAGA- An autonomos multi-agent plateform for collective artificial intelligence*". Proceedings of the first international workshop on decentralized intelligent and multi-agent systems ; Krakow, Poland, 22-24 Novembre 1995.

[SEARLE, 69] John R. Searle, "*Les actes de langage : essai de philosophie du langage*". Collection Savoir. Traduction française par Hélène Pauchard. Hermann, Paris 1972. Publié en anglais sous le titre "*Speech Acts*" par Cambridge University Press, en 1969.

[SEARLE, 79] John R Searle, "*Sens et expression, études de théories des actes de langage*". Les éditions de minuit, 1982. Traduction française et préface par joêlle Proust de "*Expression and meaning*", Cambridge University Press, 1979.

[SELLAMI, 89] Mokhtar Sellami, "*Un générateur de moteurs d'inférence pour le développement de systèmes-experts basé sur la logique des prédicats du premier ordre*". Thèse de Doctorat, Université Pierre Mendès-France, Grenoble, 1989.

[SIAMEUR, 97] Lahcène Si Ameur, "*L'étude de la cohérence dans un système de représentation des connaissances pour le traitement des langues*". Thèse de Doctorat en Informatique et Communication, Université Stendhal, Grenoble, 1997 (à paraître).

[SIAMEUR & ROUAULT, 96] Lahcène Si Ameur et Jacques Rouault, "*How the linguistic negation can have an effect in object-based knowledge representation model*". International Conference on COmputational LINGuistics (COLING'96) ; University of Copenhagen, Danemark, 05-09 août 1996.

[SIAN, 91] Sati Singh Sian, "*Adaptation based on cooperative learning in multi-agents systems*". In Yves DEMAZEAU et Jean-Pierre MÜLLER, editors, Decentralized artificial intelligence, Volume II, Elsevier science publishers, North Holland, 1991.

[SINGH, 92] Munindar Singh, "*On the Semantics of protocols among distributed intelligent agents*". In IEEE, Phoenix Conference on Computers and Communications, 1992.

[SMITH & DAVIS, 83] Reid Smith et Randall Davis, "*Negotiation as a Metaphor for distributed problem solving*". Artificial Intelligence, Volume 20, 1983.

[STEFANINI, 88] Marie-Hélène Stéfanini, "*La segmentation automatique du français écrit*". Mémoire de DEA, Université Pierre Mendès France, Grenoble II, spetembre 1988.

[STEFANINI & al., 92] Marie-Hélène Stéfanini, Alain Berrendonner, geneviève Lallich-Boidin et flavio Oquendo, "*TALISMAN : A Multi-Agent System Governed by Linguistics Laws for Natural Language Processing*". Proceedings of the 14th International Conferrence on Computational Linguistics (COLING'92), Nantes, 23-28 juillet 1992.

[STEFANINI, 93] Marie-Hélène Stéfanini, "*TALISMAN : une architecture multi-agents pour l'analyse du français écrit*". Thèse de Doctorat, Université Pierre Mendès France, Grenoble II, Janvier 1993.

[STEFANINI & DEMAZEAU, 95] Marie-Hélène Stéfanini et Yves Demazeau, "*TALISMAN : A Multi-Agent System for Natural Language Processing*". In proceedings of the 12th Brazilian Symposium on Artificial Intelligence ; Campinas, Brazil, Octobre 1995.

[STEFANINI & WARREN, 96a] Marie-Hélène Stéfanini et Karine Warren, "*A distributed architecture for text analysis in French : an application to complex linguistic phenomena processing*". International Conference on COmputational LINGuistics (COLING'96) ; University of Copenhagen, Danemark, 05-09 Août 1996.

[STEFANINI & WARREN, 96b] Marie-Hélène Stéfanini et Karine Warren, "*Analyse de phénomènes linguistiques complexes dans l'architecture TALISMAN*". Colloque Informatique & Langue Naturelle (ILN'96) ; Université de Nantes, 09-10 Octobre 1996.

[STEFANINI & WARREN, 97] Marie-Hélène Stéfanini et Karine Warren, "*La syntaxe dans un univers multi-agents*". Bulletin de Linguistique Appliquée et Générale (BULAG), publié par le centre Lucien Tesnière de l'Université de Franche-Comté, 1997.

[STERLING & SHAPIRO, 86 ou 90] Leon Sterling et Ekud Shapiro, "*L'art de Prolog*". Traduction de "*The art of Prolog*", MIT Press, Massachussets Institute of Technology, 1986. Masson, Paris, 1990, pour la traduction française.

[STOCK & al., 88] Oliviero Stock, Rino Falcone et Patrizia Insinnamo, "*Island Parsing and bidirectional Charts*". International Conference on COmputational LINGuistics (COLING'88) ; Budapest, 22-27 Août 1988.

[STUCKARD, 96] Roland Stuckard, "*Anaphor resolution and the scope of syntactic constraints*". International Conference on COmputational LINGuistics (COLING'96) ; University of Copenhagen, Danemark, 05-09 août 1996.

[THOMSON & THOMPSON, 85] B. Thomson et F. Thompson, "*ASK is transportable in half a dozen ways*". ACM Transactions on Office Information Systems, Volume 3, N°2, Avril 1985.

[TIMIMI & ROUAULT, 97] Ismaïl Timimi et Jacques Rouault, "*La paraphrase comme relation d'équivalence dans l'analyse automatique du discours*". Quatrième colloque du Traitement Automatique du Langage Naturel (TALN'97), 12-13 Juin 1997, Grenoble.

[VANDERVEKEN, 94] D. Vanderveken, "*The logic of speech acts*". Cambridge University Press, 1994.

[VERNADAT & al., 93] François Vernadat, Christophe Antoine et Pierre Azema, "*Sur la spécification et la vérification d'un protocole de négociation*". PRC-GDR Intelligence Artificielle ; Journée "systèmes multi-agents", Montpellier, 17 Décembre 1993.

[VIGNAL, 94] Laurence Vignal, "*La négociation : une utilisation constructive des conflits en résolution de problèmes*". Deuxièmes Journées IAD&SMA ; Voiron, Mai 1994.

[VON MARTIAL, 90] Frank Von Martial, "*Interactions among autonomous planning agents*". In Yves DEMAZEAU et Jean-Pierre MÜLLER, editors, Decentralized

Artificial Intelligence, Volume I, Elsevier science publishers, North Holland, 1990.

[WARREN, 96] Karine Warren, "*Protocoles de communication d'un Système Multi-agents d'analyse morpho-syntaxique du français écrit*". Colloque Etudiant de Linguistique Informatique de Montréal (CLIM'96) ; Université du Quebec à Montréal, Canada, 08-11 Juin 1996.

[WARREN & STEFANINI, 96] Karine Warren et Marie-Hélène Stéfanini, "*Modélisation et Validation de Protocoles de Communication dans l'architecture TALISMAN*". International Conference on Natural Language Processing and Industrial Applications (NLP+IA'96). 04-06 juin 1996, Moncton, New-Brunswick, Canada.

[WEHRLI, 89] Eric Wehrli, "*Deux problèmes d'analyse syntaxique automatique*". Les Cahiers de Linguistique Française N°10 : "En deçà et au-delà de l'analyse du discours" ; Université de linguistique française, Genève, 1989.

[WERNER, 90] Eric Werner, "*Distributed cooperation algorithms*". In Yves DEMAZEAU et Jean-Pierre MÜLLER, editors, Decentralized Artificial Intelligence, Volume I, Elsevier science publishers, North Holland, 1990.

[YANG, 87] Yiming Yang, "*Combining prediction, syntactic analysis and semantic analysis in chinese sentence analysis*". International Joint Conference on Artificial Intelligence (IJCAI-87), Milan, Italie, 1987.

[ZLOTKIN & ROSENSCHEIN, 91] Gilad Zlotkin et Jeffrey Rosenschein, "*Negotiation and goal relaxation*". In Yves DEMAZEAU et Jean-Pierre MÜLLER, editors, Decentralized Artificial Intelligence, Volume II, Elsevier science publishers, North Holland, 1991.

[ZLOTKIN & ROSENSCHEIN, 93] Gilad Zlotkin et Jeffrey Rosenschein, "*A domain theory for task oriented negotiation*". In proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI-93), Volume 1, Chambéry, France, 1993.

[ZWEIGENBAUM & al., 89] Pierre Zweigenbaum, Bruno Bachimont, Jacques Bouaud, Marc Cavazza et Laurent Dore, "*HELENE : Compréhension de comptes-rendus d'hospitalisation*". Deuxième école d'été sur le traitement des langues naturelles, Lanion, ENSAT, Juillet 1989.

ANNEXES

PLAN

A) Présentation de systèmes distribués de TAL ... 225
  1) Le système HEARSEY II ... 227
  2) Le système HELENE ... 227
  3) Le système CARAMEL ... 228
  4) Le système DIALECT 2 ... 229
  5) Le système TALISMAN ... 230
  6) Le système MICRO ... 230
  7) Le système ETCvérif ... 231
  8) Le système de Pompidor et Vergnaud ... 232
  9) Le système CELINE ... 232

B) Présentation de protocoles de communication ... 233
  1) Protocoles d'échanges de Campbell et D'Inverno ... 235
  2) Acquisition de connaissances de Sian ... 236
  3) Protocole d'apprentissage MOSCA ... 238
  4) Smith et Davis : Contract net protocol ... 239
  5) Protocole hiérarchique de Durfee et Montgomery ... 241
  6) Protocole de Kreifelts et Von Martial ... 241
  7) Un protocole adapté à la négociation : Chang et Woo ... 243
  8) Lander et Lesser : recherche négociée ... 245
  9) Protocoles génériques de Burmeister, Haddadi, Sundermeyer ... 247

C) La grammaire hors-contexte de la proposition assertive ... 249

D) Phrases tests pour la désambiguïsation morphologique ... 259

E) Quelques propositions "simples" testées sur TALISMAN II ... 261

A) Présentation de systèmes distribués de TAL

1) Le système HEARSEY II

Le système *Hearsay II* [Erman et al., 80] de compréhension de la parole est fondé sur le principe d'un système multi-agents avec une communication indirecte entre agents par l'intermédiaire d'un tableau noir (ou "blackboard").
La difficulté de la compréhension automatique de la parole réside dans le fait qu'à chaque étape du processus d'interprétation se présente de nouvelles ambiguïtés inhérentes. Les fonctions clés de générer, combiner et évaluer des interprétations hypothétiques sont effectuées par différents programmes indépendants appelés sources de connaissances. Chaque source de connaissance est un module indépendant fonctionnant sous le mode condition-action ; un module peut générer des hypothèses qu'il inscrit sur le tableau noir et modifier des hypothèses existantes. Le tableau noir a deux rôles : représenter les états intermédiaires de l'activité de résolution du problème et communiquer les hypothèses d'une source de connaissance à d'autres, ce qui permet de les activer. Le blackboard est divisé en plusieurs parties correspondant aux différents niveaux de représentation intermédiaire du décodage de la parole (phrase, séquence de mots, mot, syllabe, segment, signal paramétré).
La grammaire syntaxique utilisée est une grammaire hors-contexte ; les nœuds contiennent des contraintes sémantiques et pragmatiques directement liées à la structure grammaticale (grammaire appelée : "grammaire sémantique").
Chaque hypothèse est donc inscrite sur le tableau au niveau d'abstraction adéquat. Les hypothèses possibles d'un niveau forment l'espace de recherche d'une source de connaissance opérant à ce niveau. Ces hypothèses peuvent être compétitives sur un même niveau d'abstraction lorsqu'elles se chevauchent dans le temps et comprennent des interprétations partielles différentes ; elles peuvent aussi être coopératives.
Un contrôle sur les sources de connaissances est effectué par le gestionnaire de tâches. Au début de chaque cycle, le gestionnaire de tâches calcule une priorité pour chaque activité (condition ou action d'un programme d'une source de connaissance) en utilisant les informations globales ; l'activité de priorité la plus élevée est ôtée de la liste des tâches et exécutée. Quand un changement apparaît, le moniteur du tableau noir peut ajouter une activité à faire, dans la liste des tâches ; il met alors à jour les informations globales reflétant les modifications du tableau.

2) Le système HELENE

Le système *HELENE* proposé par [Zweigenbaum & al., 89] est un système pour la compréhension de comptes-rendus d'hospitalisation. Le fait que ces comptes-rendus (rédigés en langage naturel) se présentent sous forme de textes empêche l'accès direct aux informations. Le système (en cours de développement) a pour but d'extraire les informations de ces comptes-rendus pour en stocker une représentation interrogeable.

Les niveaux d'interprétation du texte sont : analyse morpho-lexicale (lexique, modèles de la flexion), analyse syntaxique (grammaire LFG), analyse sémantique (modèles du domaine, lexique sémantique) et analyse événementielle (modèles du texte). L'analyseur syntaxique utilise une grammaire lexicale fonctionnelle (LFG), décomposée en deux modules : un module de construction de la structure de constituants (c-structure) et un module de construction de la structure fonctionnelle (f-structure). Ces deux modules sont couplés à travers un tableau noir multi-dimensionnel.

Le contrôle dans le système est assuré par le tableau noir ABACAB [Bachimont, 90] qui possède un contrôleur. Les connaissances du contrôle précisent comment les modules coopèrent à la construction d'une solution. Le contrôle du tableau noir est un système de manipulation des propriétés des objets que l'on veut construire sur le tableau. Le contrôleur est considéré comme un agent rationnel raisonnant sur ses connaissances, sa représentation du problème et de sa résolution, pour prendre des décisions.

Les hypothèses fondamentales de ce système sont les suivantes :

- L'évolution du tableau est monotone : les modules ne font qu'ajouter des objets sur le tableau sans en supprimer ni en modifier.
- La solution appartient à l'espace de recherche : si le système est livré à lui-même et que les modules saturent leur espace de recherche, alors la solution est construite.
- Les connaissances données au contrôleur sont des heuristiques pour construire la solution au problème posé en un temps et un coût raisonnables.

L'architecture d'HELENE est une architecture de résolution de problèmes dirigée par les buts.

3) Le système CARAMEL

Le système *CARAMEL* (Compréhension Automatique de Récits, Apprentissage et Modélisation des Echanges Langagiers) [Sabah, 90a], [Sabah, 90b], [Sabah, 90c], est un système multi-experts utilisant le parallélisme avec une communication indirecte par tableau noir et un mécanisme de gestion des tâches par agenda.

Le système comprend trois éléments fondamentaux :

- Un ensemble important de processus réalisant les diverses tâches nécessaires.

Il existe des processus liés à *l'analyse de phrases*. En particulier, un analyseur de phrases déterministe fondé sur un grammaire de cas et une grammaire systémique ; cet analyseur construit simultanément une représentation syntaxique et sémantique de la phrase en utilisant un "regard en avant". Les systèmes de résolution d'ellipses [Sauvage, 92] et de gestion des erreurs sont déclenchés, si nécessaire, par le superviseur. Le système multi-agents ECLAIR, développé par S. Letellier [Letellier, 93], est destiné à être intégré dans CARAMEL en tant qu'expert de l'analyse lexicale et du traitement des mots non-attendus.

Les processus liés à *la compréhension de textes* sont : un système de gestion de personnages (qui construit l'image des personnages intervenant dans le texte), des systèmes d'interprétation de récits (qui permettent d'obtenir différents points de vue sur les textes traités afin de construire une représentation globale de l'histoire) et un système de gestion de plans (qui infère les intentions des différents personnages d'après leurs actions et des connaissances générales sur le monde).

Pour *la génération de textes*, le système dispose d'un générateur de phrases.

Ces différents processus sont soit des processus "élémentaires" destinés à compléter des représentations existantes ; soit des processus "complexes" qui correspondent à des enchaînements de processus élémentaires, et qui construisent de nouvelles représentations et peuvent déclencher d'autres processus.

- Une mémoire structurée contenant les connaissances et les représentations nécessaires aux processus ; chaque processus développe son propre tableau noir et le superviseur effectue une synthèse des tableaux des différents processus. La mémoire du système est organisée en trois zones : les phrases données en entrée sont placées dans une "mémoire perceptuelle" (ou mémoire à court terme), diverses représentations de différents niveaux (syntaxique, logique…) sont construites dans une "mémoire de travail". Leur construction utilise des connaissances générales stockées dans la "mémoire à long terme".
- Un superviseur SIROP planifie l'ordre idéal d'enchaînement des divers processus, en fonction du problème à résoudre, et met à jour la mémoire locale. Le superviseur analyse les représentations présentes dans la mémoire de travail, et déduit des informations attachées aux différents processus, ceux qui sont activables. De plus, chaque processus complexe fonctionne de façon analogue à SIROP. Un sous-superviseur mémorise les actions effectuées et déclenche les processus adéquats pour construire la représentation finale. Il a un rôle de planification, de résolution dynamique des problèmes et de communication avec le superviseur.

Le système permet le parallélisme grâce à la répartition des différents processus sur différentes machines d'un réseau qui s'exécutent en parallèle.

4) Le système DIALECT 2

Le système DIALECT 2 [Mekaouche & Bassano, 91], [Braunwarth & al., 93] est un analyseur linguistique multi-experts pour la recherche documentaire avec une communication par tableau noir. Dans le système, plusieurs experts coopèrent pour extraire des documents pertinents proposés, l'essentiel de l'information intéressante pour une application de type documentaire. Chacun des deux modules (analyseur linguistique et module de reformulation) de DIALECT 2 sont fondés sur une architecture de type tableau noir.

L'analyseur linguistique analyse des phrases écrites en langue naturelle et les transforme en un réseau d'énoncés élémentaires. Pour effectuer cette tâche, cinq experts collaborent par l'intermédiaire du tableau noir :

- l'expert des entrées lexicales : fournit au tableau noir la liste des entrées lexicales de chaque mot de la phrase : forme fléchie, formes canoniques et catégories grammaticales.
- l'expert des homographes : résout les ambiguïtés engendrées par les mots possédant plusieurs entrées lexicales. Pour cela, il construit toutes les chaînes syntaxiques possibles qu'il stocke sous une forme arborescente.
- l'expert de lexis : construit les lexis à partir des entrées lexicales du tableau noir.
- l'expert de mots grammaticaux : permet de compléter les lexis lors de la présence de conjonctions de coordination, de pronoms relatifs ou de phrases interrogatives.
- l'expert correcteur : permet une correction automatique d'éventuelles fautes de frappe.

Le module de reformulation a pour but d'augmenter la pertinence des réponses. La méthode utilisée consiste à la construction dynamique de classes sémantiques, avec la distinction de deux types d'utilisateurs : expert ou occasionnel. Pour chaque type d'utilisateur, il existe un paquet de règles de reformulation. Ces règles sont caractérisées par un type de relation évaluant la proximité sémantique entre les mots mis en rapport.

Certains experts de la reformulation et de l'analyseur linguistique fonctionnent en parallèle. Le tableau noir contient un agenda gardant la trace de la progression de l'analyse de la phrase. Il adapte sa stratégie de résolution suivant la situation rencontrée. Pour cela, il possède une base de règles qui lui permet de choisir l'expert qui convient le mieux à la situation. L'activation d'un expert est donc réalisée suivant des données de contrôle stockées sur le tableau noir ; ce dernier, dirige les traitements subis par les objets du tableau noir, en choisissant, à chaque étape, l'expert qui fera le plus avancer la solution.
DIALECT 2 propose deux modes de fonctionnement : un mode "utilisateur occasionnel" (l'utilisateur fournit une requête et le système lui fournit une réponse sans aucune intervention de la part de l'utilisateur) et un mode "utilisateur expert" (l'utilisateur peut alors dialoguer avec le système afin d'améliorer la réponse du système).

5) Le système TALISMAN

TALISMAN [Stéfanini, 93] est une architecture multi-agents gouvernée par des lois, pour l'analyse morpho-syntaxique du français écrit. L'architecture proposée utilise une communication directe entre des agents cognitifs.
Les agents de TALISMAN sont composés de deux modules : *un module linguistique* comportant le but de l'agent et toutes ses compétences, ses stratégies et ses connaissances linguistiques, et *un module de communication* contenant les éléments lui permettant de communiquer avec les autres agents (format des messages, boîte aux lettres, protocole de communication).
La maquette implémentée comporte les cinq agents linguistiques suivants : *Pret* (expert en prétraitement), *Morph* (expert en morphologie), *Transf* (expert en transformation), *Segm* (expert en segmentation) et *Synt* (expert en syntaxe).
La coopération entre agents est possible, dans TALISMAN, par l'envoi de messages. Ces échanges directs de messages sont régulés par un protocole de communication. Le protocole de communication utilisé dans le système est celui de Sati Sian [Sian, 91].
Le contrôle du système est mixte. Il est assuré par l'utilisation de lois inspirées de [Minsky, 91a et b] : les lois globales (pour la gestion du système) et les lois locales (pour le contrôle réparti dans chaque agent).

6) Le système MICRO

Le système MICRO (Modélisation Informatique de la Cognition en Reconnaissance de l'Oral) est un système multi-agents pour la compréhension de la parole [Caillaud & al., 94]. L'architecture de ce système est fondée sur l'environnement MAPS [Baujard & Garbay, 90] qui distingue :

- les connaissances descriptives : données, hypothèses, résultats,
- les connaissances opératoires et décisionnelles : règles et méta-règles.

A chacune de ces classes de connaissances correspond un type d'agent : les agents KS (Knowledge Server) représentent les connaissances descriptives et les agents KP

(Knowledge Processor) correspondent aux connaissances opératoires et décisionnelles. Les agents du système communiquent par envois de messages et réagissent à des événements extérieurs. Les agents KS communiquent usuellement avec des agents KP (soumission de données et de problèmes) ; et réciproquement, les agents KP avec des agents KS (transmission et demande d'informations). Les agents KS et KP sont supervisés par d'autres agents appelés "méta-KP" chargés du contrôle des tâches. MAPS permet l'exécution en parallèle des processus attachés aux agents.

Les agents sont rassemblés en 5 groupes : perception, acoustico-phonétique (analytique), linguistique[68] (analytique), prosodie (holistique), et dialogue.

Ces groupes correspondent aux deux niveaux d'articulation de la langue : linguistique d'une part, et acoustico-phonétique d'autre part.

Dans le groupe acoustico-phonétique "simulé", un agent a été ajouté. Cet agent est spécialisé dans l'unification des solutions partielles des KP du groupe ; il se nomme "KP unification". Pour cet expert d'unification, les auteurs ont testés trois stratégies ; cette expérience leur a montré qu'il n'existait pas de stratégie unique ayant des performances acceptables. Ils proposent donc la mise en place d'une stratégie variable suivant le contexte, choisie parmi les trois stratégies qui, tour à tour, sont plus ou moins performantes. Cette méta-stratégie règle le comportement des agents ou conditionne le processus d'unification.

L'objectif des auteurs était de simuler une partie du système pour tester le contrôle à mettre en œuvre ; ils ont donc conclu à la nécessité d'adopter une stratégie adaptative.

7) Le système ETCvérif

Le système développé par *Cochard et Froidevaux* [Cochard & Froidevaux, 95] est un environnement multi-agents de reconnaissance automatique de la parole en continu. Cet environnement multi-agents a été conçu dans le cadre du projet de recherche ETCvérif (Environnement de Traitement Coopératif appliqué à la VERIFication d'échantillons de parole). Il comprend des agents macroscopiques et microscopiques qui cohabitent et communiquent pour créer une dynamique de résolution de problèmes partiellement répartie ; le système utilise un agent de négociation qui rythme le processus de résolution.

L'objectif est de concevoir une plate-forme multi-agents qui intègre un modèle de coopération fondé sur la recherche négociée d'une solution. Le système doit favoriser la réutilisabilité de programmes existants, faciliter l'intégration de nouvelles sources de connaissances et autoriser la gestion de résultats concurrents.

Dans l'architecture proposée, les traitements spécifiques à la résolution d'un problème sont encapsulés dans des entités réactives nommées agents macroscopiques et les résultats élémentaires produits par chaque agent macroscopique donnent naissance à d'autres entités réactives nommées agents microscopiques.

Chaque agent macroscopique a un espace de solutions admissibles à parcourir, pour un problème donné.

Chaque agent microscopique doit se mettre en correspondance avec d'autres agents microscopiques dans le but d'établir des relations de voisinage qui sont, soit horizontales (quand les agents contiennent des informations comparables), soit verticales (pour les liens de dépendance entre entrée et sortie d'un agent macroscopique). L'interprétation des relations horizontales et verticales qu'un agent établit avec son environnement, lui permet de définir une mesure de confiance en soi de sa propre donnée.

---

[68] L'analyseur syntaxique utilise une grammaire LFG.

Des îlots de confiance sont obtenus en interrogeant les agents microscopiques sur leur facteur de confiance en soi ; et un agent de négociation rythme le processus de calcul de la confiance du système. Le modèle de négociation est fondé sur un principe de croissance monotone des îlots de confiance, ainsi lorsqu'un résultat est jugé suffisamment fiable, il ne sera jamais remis en cause jusqu'à la fin de l'exécution.

8) Le système de Pompidor et Vergnaud

Le système développé par *Pompidor et Vergnaud* [Pompidor & Vergnaud, 95] est un système multi-agents pour la traduction automatique du chinois dans un but pédagogique. La traduction du chinois en français est vue comme une coopération de connaissances hétérogènes. Le système comporte un agent cognitif de contrôle général. Le mode de communication entre agents est directe, et utilise des protocoles de communication.

Dans le système, les outils informatiques intégrés sont des bases de données gérées soit par des agents cognitifs, soit par un ensemble d'agents réactifs. Le système comporte :

- un agent cognitif gérant le dialogue avec l'utilisateur et le contrôle général du système,
- un agent cognitif gérant un dictionnaire de mots chinois-français,
- un ensemble d'agents réactifs gérant une base de structures grammaticales chinoises,
- un ensemble d'agents réactifs gérant une base de structures grammaticales françaises,
- un ensemble d'agents réactifs gérant une base de phrases types,
- un ensemble d'agents réactifs gérant une base d'informations contextuelles.

Les agents réactifs coopèrent pour analyser la phrase chinoise et synthétiser la phrase française correspondante. Le comportement d'un agent réactif suit une logique stimulus-réponse. Les agents réactifs communiquent par envois de messages à l'aide d'un protocole de communication. L'état de stabilité des agents réactifs, atteint lors de la raréfaction des messages, correspond à une émergence possible de solutions.

L'agent de contrôle a pour tâches :

- adresser la requête de l'utilisateur à l'agent gérant le dictionnaire,
- adresser le niveau d'expertise du chinois (communiqué par l'utilisateur) aux agents réactifs gérant les bases pour que ceux-ci ne se chargent que des informations de ce niveau,
- surveiller les agents réactifs pour détecter l'émergence des différentes solutions et les remonter à l'utilisateur,
- gérer les règles de contrôle de la coopération et de la révision des connaissances.

9) Le système CELINE

CELINE (CorrEcteur LexIco syNtaxiquE) [Genthial et al., 96], [Ménezo et al., 96a et b] est un système multi-agents de détection et de correction des erreurs lexicales et syntaxique. Il présente une architecture pyramidale et hiérarchique. Le système comprend un superviseur, des pilotes par secteurs d'activités et des agents de travail ; dans ce SMA mixte, cohabitent des agents réactifs et des agents cognitifs et la communication se fait par envois de messages. Ce système permet une acquisition d'expériences par les agents, une évolution dynamique de leurs accointances et de leur crédibilité, ce qui entraîne une optimisation de leur efficacité. Les agents suivants négocient dans le but de corriger et de générer des phrases syntaxiquement correctes :

- un pilote des activités lexicales propose des corrections,
- un pilote des activités syntaxiques et des agents d'analyse syntaxique valident ou refusent les différentes combinaisons syntaxiques,
- un agent, utilisant les modèles cachés de Markov, peut aider au choix de formes,
- un pilote de l'activité de vérification des accords et un agent de détection et de correction des fautes d'accord corrigent les accords,
- un générateur morphologique fournit les graphies correctes,
- un agent formate le texte en ajoutant une majuscule en début de phrase et avec un seul blanc entre deux mots consécutifs.

Le superviseur coordonne l'action de plusieurs pilotes. Chaque pilote est responsable d'un secteur d'activité (lexicale, syntaxique, accords en genre et en nombre, suivi statistique,…) et coordonne le travail de plusieurs agents de travail. Chaque agent est spécialiste d'un domaine étroit mais peut dépendre de plusieurs pilotes.

CELINE est un système ouvert acceptant des lexiques ou analyseurs divers et les couvertures lexicales et grammaticales sont donc celles de ces lexiques ou analyseurs. Au niveau syntaxique, le système peut utiliser deux prototypes : le premier fondé sur les structures de dépendances et le second sur la méthode des cartes et utilisant une grammaire de constituants.

B) Présentation de protocoles de communication

1) Protocoles d'échanges de Campbell et D'Inverno

J. Campbell et M. D'Inverno [Campbell & D'Inverno, 90] sont partis de l'hypothèse que tout agent entamant un échange de messages a une intention : tout échange est la conséquence d'un plan ou d'un but d'un agent. Pour ces auteurs, l'échange de connaissance est probablement la forme la plus complexe de communication entre agents. En particulier, la première parole d'un agent initialisant une communication sera gouvernée par son but courant le plus important.

Campbell et D'Inverno effectuent une liste des différents tons possibles du dialogue :

action requesting (demander une action), *information seeking* (rechercher une information), *information probing* (sonder une information), *information checking* (contrôler une information), *instructing* (enseigner), *informing* (informer), *understanding an unexpected event* (comprendre un événement inattendu), *warning* (prévenir de faire attention), *advising* (avertir), *persuading* (persuader), *promising* (promettre), *bargaining* (marchander), *impressing* (impressionner), *intimidating* (intimider), *threatening* (effrayer), *commanding* (commander), *encouraging* (encourager), *expressing of one's own state* (exprimer son état personnel), *offending* (choquer), *misleading* (tromper), *amusing* (amuser).

Pour chacun de ces tons, les auteurs supposent possible la construction de protocoles. L'idée est qu'un agent voulant entamer une communication va choisir un des protocoles. Les auteurs introduisent pour la formalisation des échanges, des prédicats logiques (*desire* (désirer), *assume* (supposer), *intend* (avoir l'intention de), *know* (savoir), *expect* (attendre),…) ainsi que des axiomes (ex. : A *intend* X → A *expect* X).

Les auteurs ont détaillé plusieurs protocoles, mais nous n'exposerons que le protocole de recherche d'action qui nous paraît le plus intéressant de ceux développés.

• Exemple du protocole *ACTION SEEKING* (recherche d'action)

Dans ce cas, l'agent A veut que l'agent B fasse l'action X.

L'agent A dispose de l'état mental suivant :

1. A désire X
2. A suppose (B capable-de-faire X)
3. A a-l'intention-de C où R(C) → B a-l'intention-de X
4. A désire (B a-l'intention-de X)
5. A suppose (B (non-a-l'intention-de) X)
6. A suppose : a) B désire X
   b) B (non désire) X

L'agent A va comparer cet état mental courant avec les préconditions de chaque protocole. A va alors choisir le protocole *Action Seeking* car c'est celui dont les préconditions 1. et 2. correspondent le plus à son état mental. L'agent A va alors communiquer C car le résultat de C (= R(C)) conduit B à avoir l'intention de réaliser l'action X.

Plusieurs cas peuvent alors se produire :

1er cas : B ne peut pas faire X ; il en informe A et donne ses raisons ; puis
- A insiste,
ou - A renonce et raye l'hypothèse 2.

2ème cas : B ne comprend pas la requête de A, alors
- B demande à A de répéter la question,
ou - B demande à A de clarifier certains points.

3ème cas : B désire effectivement X comme le supposait A (hypothèse 6-a), dans ce cas B va faire X.

4ème cas : B ne désire effectivement pas X comme le supposait A (hypothèse 6-b), A va devoir insister et même éventuellement utiliser d'autres tons du dialogue comme en lui faisant peur par exemple.

5ème cas : B ne désire pas X contrairement à ce que pensait A (hypothèse 6-a fausse) ; B en informe A qui va mettre à jour ses hypothèses et reconsidérer sa requête.

Campbell et D'Inverno ont montré la capacité des protocoles de communication à décrire toutes les intentions possibles.
Il semble cependant que parmi les 21 tons, tous n'ont pas réellement d'utilité (*bargaining* (marchander), *amusing* (amuser),…). De plus, aucune implémentation n'a été faite et elle paraît complexe (à cause des prédicats, des axiomes et des 21 protocoles).

2) Acquisition de connaissances de Sian

Sati Sian [Sian, 91] est parti du principe que les systèmes intelligents sont caractérisés par leur capacité à s'adapter en apprenant de leur environnement. Chaque agent utilise ses informations locales pour inférer des hypothèses partielles et ensuite interagit avec les autres pour modifier et/ou augmenter son degré de confiance en ses hypothèses.
Pour communiquer avec l'environnement et les autres agents, les agents vont utiliser un langage de communication ; cette communication va permettre à chaque agent d'acquérir des connaissances sur les autres agents, leurs capacités et leurs résultats. La coopération entre agents va leur permettre d'apprendre des expériences des autres et de compléter leurs hypothèses partielles : cela s'avère utile dans le cas de résolution par des agents de problèmes distincts (besoin de connaissances complémentaires), équivalents (besoin de l'expérience des autres) ou interdépendants (les connaissances et les expériences des autres peuvent être intéressantes).

Le langage développé par Sati Sian est un langage commun comportant 9 primitives qui sont :
- *assert* (affirme) : représente une assertion non modifiable,
- *propose* (propose) : permet de proposer une nouvelle hypothèse,
- *modify* (modifie) : donne une version modifiée de l'hypothèse précédemment proposée,
- *agreed* (accord) : exprime un accord sur une hypothèse proposée,
- *disagree* (désaccord) : exprime un désaccord sur une hypothèse,
- *noopinion* (sans opinion) : n'exprime aucune opinion sur l'hypothèse,
- *confirm* (confirme) : permet de confirmer l'hypothèse,
- *accept* (accepte) : exprime l'acceptation d'une hypothèse,
- *withdraw* (rejet) : exprime le rejet de l'hypothèse proposée.

L'implémentation réalisée (MALE = Multi-Agent Learning Environment) comporte un tableau d'interaction structuré par niveaux. Les agents connectés au tableau utilisent un langage de communication CONIC via l'échange d'envois de messages.
Sian a effectué 3 cas de test : un test sans apprentissage, un test utilisant l'apprentissage avec sa propre connaissance et un test avec apprentissage coopératif ; c'est ce dernier qui a donné les meilleurs résultats.
Un agent comporte trois modules :

- un module mémorisant l'expérience appelé "*Experience Store*",
- un module d'apprentissage appelé "*Learning Sub-System*" permettant d'apprendre à partir d'expériences (pour confirmer une hypothèse) et par interaction avec le tableau noir (les agents doivent donner leur opinion sur les hypothèses du tableau),
- un module de traitement pour générer les actions, appelé "*Performance Component*", contenant un ensemble de règles par défaut ou opérationnelles auxquelles sont associés des degrés de confiance.

Le but de Sian est de permettre à chaque agent d'utiliser des informations locales pour inférer des hypothèses partielles, puis d'interagir avec les autres agents pour ensemble corriger et améliorer le degré de confiance dans leurs hypothèses. Une fonction d'évaluation fournit à l'agent qui a soumis une hypothèse, le coefficient de confiance obtenu à partir des réponses de tous les autres agents. Ces protocoles comportent 3 phases nécessaires pour apprendre par communication :

- la génération d'hypothèses initiales par un des agents (problèmes : quand et comment générer l'hypothèse),
- la coopération pour parvenir à une version acceptée par tous les agents, concernant cette hypothèse (problème : évaluation des hypothèses des autres),
- intégration des informations résultant des échanges dans l'ensemble des connaissances de chacun des agents (problème : amalgame d'opinion).

L'enchaînement des 9 primitives pour une hypothèse constituent le protocole de Sian (cf. Figure 31).

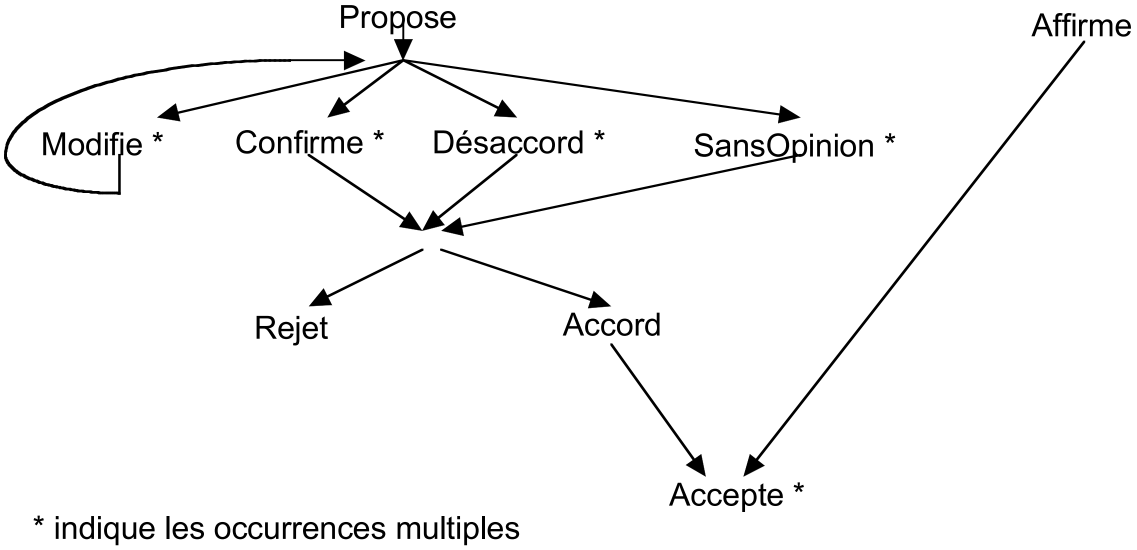


Figure 31 : Protocole d'interaction de Sian

Ce schéma donne les séquences possibles pour l'utilisation des primitives. Toute interaction débute avec un *Propose* ou un *Affirme*. A la primitive *Propose*, les autres agents peuvent répondre par un *Modifie*, *Confirme*, *Désaccord* ou un *SansOpinion*. L'hypothèse modifiée aura le même statut que l'hypothèse proposée et doit donc générer le même type de réponse. Selon la valeur du coefficient de confiance, une hypothèse peut être acceptée (*accord*) ou retirée (*rejet*). Dans le cas où elle serait acceptée, il se peut que tous les agents l'acceptent. Quant à un *affirme*, il ne peut produire en réponse qu'un *accepte*.

L'intérêt principal de ce protocole est qu'il permet d'accroître la performance des agents qui coopèrent. Les agents n'ont pas d'informations complètes c'est pourquoi ils doivent coopérer. La coopération est utile car elle permet aux agents d'apprendre des expériences des autres et de compléter le tableau de leurs hypothèses partielles. L'interaction permet d'accroître le degré de confiance dans une hypothèse acceptée ; donc l'hypothèse a plus de chances d'être correcte. Un point important est de remarquer que la démarche proposée évite de maintenir un modèle des croyances des autres agents. Les échanges formulés permettent d'obtenir à un instant donné une représentation partielle des opinions des agents par rapport à un sujet par l'intermédiaire du tableau.

Un inconvénient est que le système d'apprentissage comprend un temps d'exécution élevé car il est coûteux en envoi de messages. L'acquisition et l'amélioration de la qualité des informations se fait ici en commun. Cela implique une implémentation assez complexe des capacités de raisonnement.

3) Protocole d'apprentissage MOSCA

Le protocole MOSCA (cf. Figure 32) modélise une situation typique d'apprentissage [Reitz, 92]. L'apprentissage est réalisé par l'interaction de cinq différents types d'entités : le Maître, l'Oracle, la Sonde, le Client et l'Apprenti. Les différents types de message que les agents peuvent s'envoyer correspondent aux interactions entre les différents éléments intervenant dans un processus d'apprentissage automatique.

L'Apprenti est un résolveur de problèmes qui possède un échantillon de problèmes résolus à partir duquel il travaille.

Les quatre entités Maître, Oracle, Sonde et Client sont des idéalisations de rôles distincts :

- Un Oracle produit des problèmes correctement résolus, dont les solutions ne peuvent être contestées. L'Apprenti reçoit donc de l'Oracle des problèmes résolus qu'il mémorise et qui forment son échantillon d'apprentissage.
- Une Sonde produit des problèmes résolus dont la solution n'est pas nécessairement correcte. Ainsi, l'Apprenti reçoit les solutions de la Sonde qu'il compare à son échantillon d'apprentissage. Il peut alors argumenter avec le Maître pour valider ou non son hypothèse apprise.
- Un Maître contrôle la production des problèmes résolus de la Sonde au travers de requêtes. Cette requête contient l'énoncé du problème à résoudre. Le Maître exprime aussi des critiques irréfutables sur toute argumentation de l'Apprenti.
- Un Client soumet des problèmes à résoudre à l'Apprenti et en attend une solution.

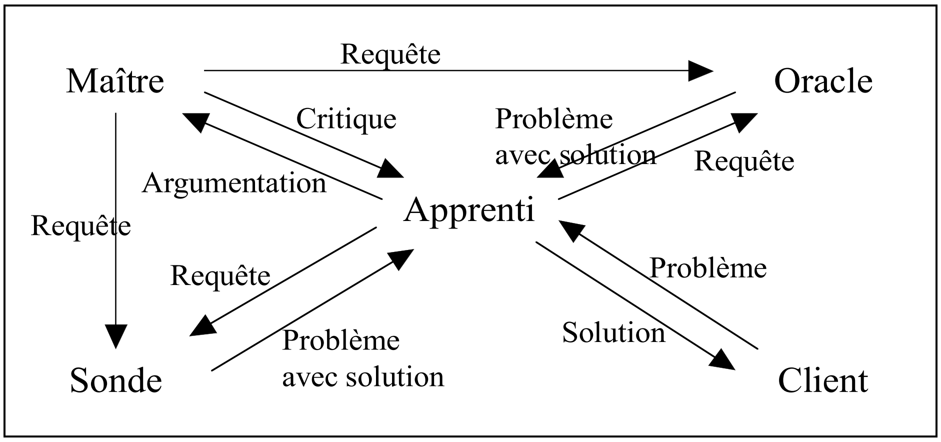


Figure 32 : Protocole d'apprentissage MOSCA

Ces cinq rôles sont à partager par deux acteurs : la machine et son usager. L'auteur a imaginé différentes situations d'apprentissage. Par exemple, l'usager peut instruire le système ; l'usager est alors le maître, l'Oracle et la Sonde, et le système est l'Apprenti[69]. C'est le cas lorsqu'un expert utilise un système d'aide à la modélisation. L'usager peut aussi être instruit par le système ; l'usager est alors l'Apprenti et le système comprend les quatre autres rôles. En tant que Sonde, le système propose des exercices dont il attend des solutions. En tant qu'Oracle, il indique la solution à un exercice (sans qu'elle soit contestable) obligeant l'élève à reconsidérer ce qu'il a appris. De même, en tant que Maître, il envoie une critique incontestable. Le protocole MOSCA a été utilisé [Guin & al., 93] pour l'apprentissage d'un élève par une interaction didactique entre l'enseignant artificiel et l'élève (i.e. l'Apprenti). En effet, l'environnement proposé se prête à la modélisation de ce type de situation d'apprentissage. La modélisation des cinq rôles différents ne nous paraît cependant pas aisée. La spécification, en particulier, des connaissances du Maître pouvant juger toute solution, est difficile.

4) Smith et Davis : Contract net protocol

Ce protocole de réseau de contrat [Smith & Davis, 83] permet une allocation dynamique et distribuée des tâches. Le manager va décomposer ses tâches en sous-tâches puis il va faire un appel d'offre sur une tâche. Les agents susceptibles d'effectuer cette tâche vont lui faire des propositions. Le manager attribuera la tâche au travailleur de son choix (en fonction de certains critères) et le contrat sera établi.

Ce protocole reproduit donc la négociation entre agents sous la forme de l'offre et de la demande ; il permet de formaliser et contrôler la répartition des sous-tâches entre les agents constituant le système. Ce protocole de résolution de problèmes permet d'obtenir un comportement coopératif entre les agents en s'appuyant sur la métaphore du système de l'offre et de la demande.

Le protocole de communication du réseau contractuel se décompose de la manière suivante (cf. Figure 33) :

1) Appel d'offres : un agent, appelé manager, décompose une de ses tâches en sous-tâches puis envoi un appel d'offre sous la forme d’un message de proposition (l'offre).

---

[69] Le rôle de Client est alors inexistant.

Cette offre est composée : des spécifications des conditions d'éligibilité, de la description partielle de la tâche, de la description de la forme de la réponse attendue et du temps d'expiration de la demande.

2) Propositions : les agents intéressés par le traitement de cette tâche, appelés contractants, envoient une proposition au manager ; les contractants doivent répondre aux conditions d'éligibilité de la tâche ; la proposition exposant ses caractéristiques pour réaliser la tâche, peut éventuellement contenir une demande d'informations supplémentaires sur le problème à traiter.

3) Attribution du contrat : le manager évalue les offres et attribue la tâche à l’agent, appelé agent contractant, qui semble le plus adapté à la sous-traitance.

4) Etablissement du contrat : le manager va alors annoncer son accord à l'agent élu en spécifiant la tâche et éventuellement des informations supplémentaires ayant pu être demandées.

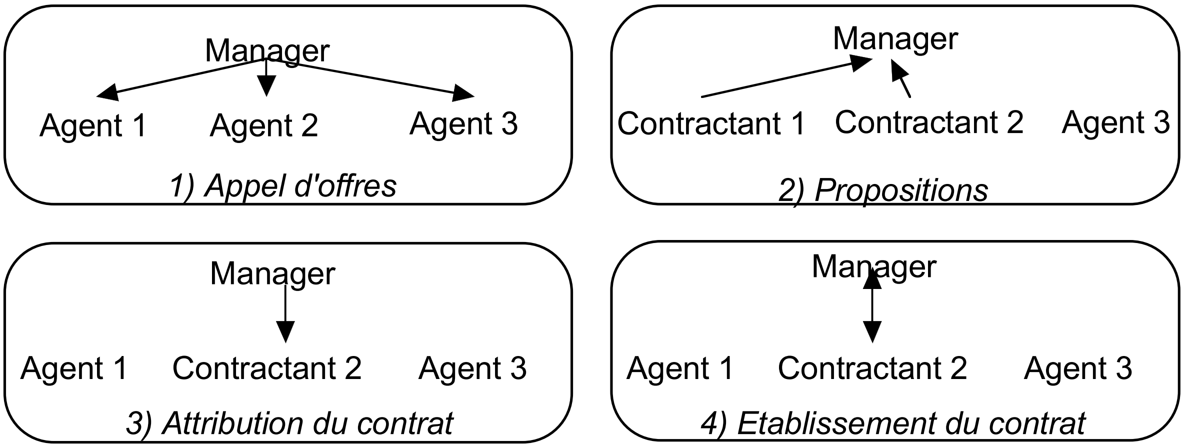


Figure 33 : Protocole de réseau de contrats

Un réseau d'agents se construit ainsi dynamiquement pour résoudre une tâche globale. Le contrôle est distribué car un même agent peut être travailleur ou manager pour des tâches différentes. Pour cela, les agents tiennent à jour un tableau de structures correspondant aux contrats des tâches qu'ils ont sous-traités. C'est le seul modèle de connaissances extérieures qui est implanté dans chaque agent.

Les auteurs ont défini un format des messages ainsi qu'un langage de communication ; un message véhicule toutes les informations nécessaires aux différentes phases du protocole.

Ce protocole agit comme une sorte de structure de contrôle de haut niveau pour l'allocation de tâches. Il présente les avantages suivants :

- la distribution des décisions : le contrôle décentralisé dans les agents,
- la symétrie des positions : les agents managers et contractants n'ont pas de rapport de hiérarchie entre eux,
- l'extensibilité (l'ajout d'un agent ne change rien).

Mais les inconvénients de ce protocole sont :

- le coût des envois de messages est augmenté du fait de la diffusion d'information lors de la propositions de tâches,
- le protocole est rigide car il ne permet pas de procédure d'apprentissage dans le cas d'une tâche déjà traitée entre agents (s’ils ont déjà travaillé ensemble) ni de procédure de résiliation de contrat lorsque les agents veulent sortir du protocole.

5) Protocole hiérarchique de Durfee et Montgomery

Dans ce protocole [Durfee & Montgomery, 90], chaque agent possède des comportements associés à la réalisation de son but. Ces comportements sont composés de différentes informations (réparties dans les six dimensions suivantes : *qui*, *quoi*, *quand*, *où*, *comment*, *pourquoi*) et sont organisés de manière hiérarchique. Les auteurs supposent de plus, que les agents ne savent pas, a priori, avec qui ils vont interagir. Ils ont défini un protocole qui a été utilisé pour le problème général de la détection et de la résolution de conflits de ressource par des robots.
Le protocole se déroule de la manière suivante :

- Chaque robot commence par envoyer à tous les autres robots du système des informations concernant son identité et son but.
- Ensuite, les robots construisent une liste indiquant la hiérarchie des robots en fonction des priorités de leur but.
- Le robot de la plus haute autorité envoie l'information concernant son comportement le plus abstrait aux autres robots (inférieurs dans la hiérarchie sociale).
- Chaque robot (inférieur) compare alors ses informations locales avec ce qu'il a reçu et recherche l'existence de conflits ; un conflit se produit si deux robots sont dans une même région (*où*) dans le même intervalle de temps (*quand*). Si un conflit est détecté, l'agent inférieur doit le traiter. Soit, il le résout en modifiant la région ou l'intervalle de temps ; soit, il décide d'explorer plus en détail les aspects du comportement qui peuvent provoquer un conflit. Le supérieur et l'inférieur s'échangent des informations de plus en plus détaillées jusqu'à ce que l'inférieur découvre que le conflit n'aura pas lieu, ou qu'il résolve le conflit.
- Quand le robot supérieur a reçu un message de tous les agents l'informant qu'il n'y a plus de conflits, il passe le contrôle au prochain robot dans la hiérarchie sociale. Le processus se répète alors : le nouveau supérieur interagit avec les inférieurs pour résoudre les éventuels conflits. Ce mécanisme se poursuit jusqu'à ce qu'il n'y ait plus de robots inférieurs ; les conflits auront alors été tous résolus. Mais quand un inférieur modifie son comportement pour éviter un conflit avec le supérieur courant, il doit vérifier auprès des supérieurs précédents qu'il ne crée pas de nouveaux conflits. Si un nouveau conflit est apparu, le robot inférieur utilise la même technique pour résoudre le conflit avec l'ancien supérieur et regarde ensuite les conflits potentiels avec l'actuel robot supérieur.
- Enfin, quand le dernier robot de la hiérarchie sociale a eu le contrôle, les robots vont pouvoir commencer leurs activités.

Quand toutes les activités des différents robots sont terminées, le protocole est de nouveau initialisé pour le prochain but du système.
Ce protocole permet donc une forme de coopération appropriée aux domaines où, d'une part, les agents ne savent pas, a priori, avec qui ils vont interagir, et dans lesquels, d'autre part, il est possible d'établir une hiérarchie sociale entre agents.

6) Protocole de Kreifelts et Von Martial

Pour ces auteurs [Kreifelts & Von Martial, 91], la négociation est pratiquée par des conversations avec un agent "Coordination" qui fait le médiateur entre les autres agents. Le besoin de négociation est déclenché par l'existence de relations positives ou négatives [Von

Martial, 90] entre les plans des différents agents ; les relations entre les plans des agents sont la conséquence des interactions de plans élaborés individuellement qui s'exécutent dans un environnement commun. La détection des relations négatives (celles qui empêchent la réalisation d'un plan) est cruciale pour empêcher les conflits. La détection des relations positives (celles qui vont être bénéfiques aux plans des agents) peut aider à utiliser le potentiel favorable qui peut exister parmi les différents plans.

Les agents planifient leurs activités puis coordonnent leurs plans grâce à la communication ; pour cela ils vont envoyer leurs plans à l'agent Coordination. Ce dernier va chercher les actions nécessitant une négociation. Les agents négocieront par l'intermédiaire de l'agent Coordination donc indirectement. Mais, comme le soulignent les auteurs, ce rôle de l'agent Coordination peut être réparti dans plusieurs agents auquel cas il n'y a plus d'agent Coordination.

Les différentes phases de l'activité de la coordination sont :

- détection de la relation positive ou négative entre deux plans,
- sélection du prochain sujet de négociation,
- sélection de la stratégie grâce à des heuristiques,
- négociation : l'agent coordination cherche un compromis acceptable par tous les agents.

Le protocole de communication (ou règles de conversation) élaboré permet de relier les états et les types de message (cf. Figure 34). Ainsi, le modèle de conversation a trois composants : l'état de la conversation, le type de message que l'on peut échanger, les règles de la conversation qui spécifient dans quel état quel type de messages un agent peut envoyer ou recevoir.

Les types de message que les auteurs introduisent pour la communication sont les suivants :

- *action* : un agent annonce à l'agent Coordination le détail de l'action planifiée qu'il va entreprendre ; ce type de message démarre la conversation.
- *proposal* (proposition) : l'agent Coordination propose une modification de l'action dans le but d'atteindre un compromis en fonction des autres actions.
- *approval* (approbation), *counter* (contre), *rejection* (rejet) : réactions possibles des agents à une proposition.
- *resolution* (résolution), *failure* (échec) : confirmation positive ou négative de l'agent Coordination ; clôt une phase de négociation.
- *disappearance* (disparition) : annonce la raison de la disparition de la négociation (causée par le rejet d'une action dans une autre négociation).
- *back-out request* (demande renouvelée), *re-proposal* (proposition renouvelée) : initialise une nouvelle négociation pour un compromis déjà atteint par l'agent Coordination ou par d'autres agents.
- *execution* (exécution), *withdrawal* (rejet total) : l'agent annonce qu'il a exécuté ou qu'il n'exécutera pas l'action ; ce type de message termine la conversation car il n'existe plus rien à négocier.

Les états possibles de la conversation sont :

initial (initial), *planned* (planifié), *proposed* (proposé), *answered* (répondu), *committed* (compromis), *unresolved* (non résolu), *re-proposed* (re-proposé), *back-out-requested* (re-demandé), *re-answered* (re-répondu), *executed* (éxécuté) ou *withdrawn* (rejeté).

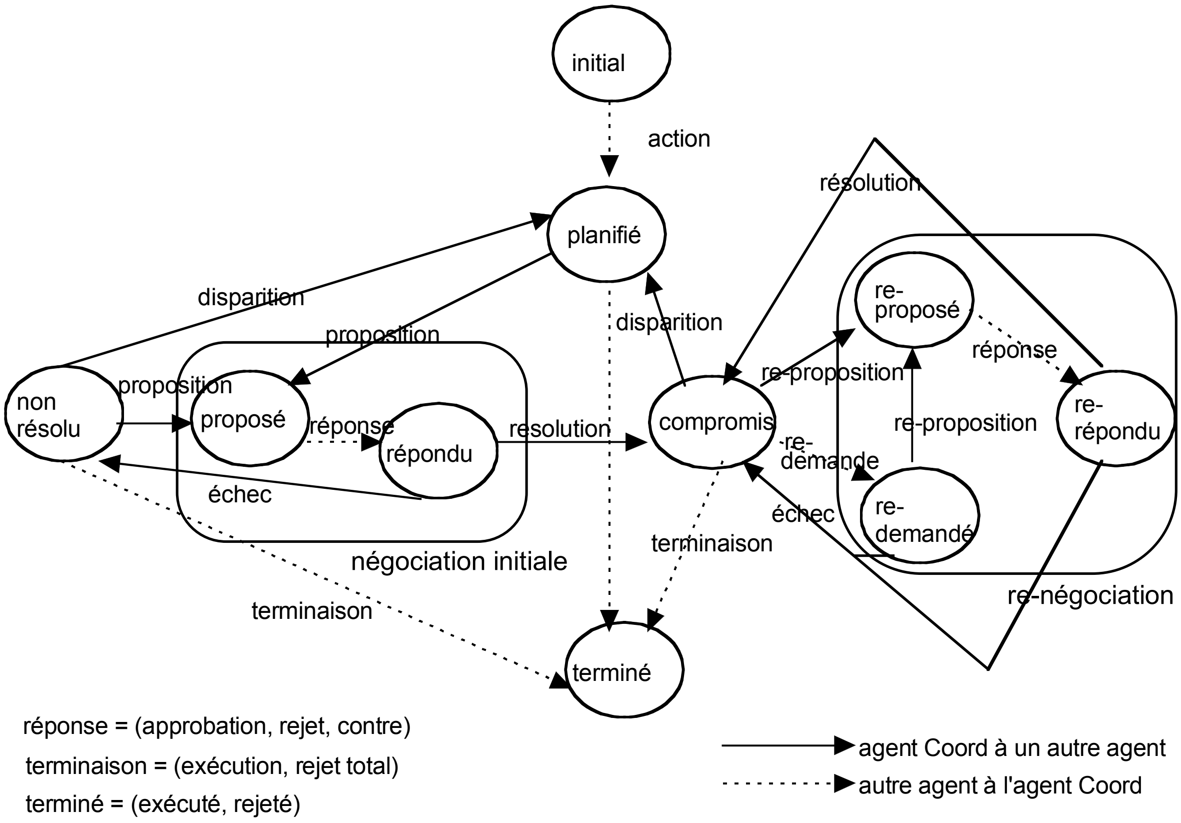

initial
action
planifié
disparition
proposition
non résolu
proposition
proposé
réponse
répondu
résolution
échec
négociation initiale
terminaison
disparition
compromis
résolution
re-proposé
réponse
re-proposition
re-proposition
re-répondu
-re-demande
re-demandé
échec
terminaison
re-négociation
terminé
réponse = (approbation, rejet, contre)
terminaison = (exécution, rejet total)
terminé = (exécuté, rejeté)
agent Coord à un autre agent
autre agent à l'agent Coord

Diagramme de transition d'état pour la négociation

Figure 34 : Protocole de Kreifelts et Von Martial

Ce protocole est approprié à la résolution de conflits par l'intermédiaire d'un agent Coordination qui gère la négociation entre agents via la communication.

7) Un protocole adapté à la négociation : Chang et Woo

Ce protocole SANP (= Speech Act based Negotiation Protocol) [Chang & Woo, 92] permet de résoudre les conflits à l'aide d'une négociation évoluée et flexible. Il permet de prendre en compte le résultat des échanges. Les connaissances de l'agent incluant également ses capacités de raisonnement et de décision, sont représentées par des équations, tandis que les faits correspondent à des variables instanciées.
Les règles de ce protocole pour guider la conversation sont :

- attendre une réponse avant d'émettre une nouvelle requête,
- la réponse est limitée par le protocole utilisé.

Les messages échangés sont composés de deux champs :

- une "fonction" correspondant à un acte de dialogue,
- le contenu.

Ces actes de dialogue sont : *make claim* (requête), *agree* (accord), *withdraw* (rejet), *argumentative attack* (attaque argumentée), *postpone* (différer), *admit defeat* (défaite admise), *argumentative defense* (défense argumentée), *repeated attack* (attaque répétée), *insisting* (insiste), *retreat* (battre en retraite), *offer compromise* (offre d'un compromis),…

Ce protocole comprend six principales étapes :

1- situation initiale : les agents s'assurent qu'ils possèdent les connaissances suffisantes pour discuter du problème,

2- demande et réponse (cf. Figure 35) : l'attaquant commence par adresser sa requête (un plan, une demande d'action ou une assertion) et il réceptionne la réaction de l'interlocuteur (le défenseur),

3- attaque : si le défenseur n'approuve pas la requête l'attaquant va commencer à l'attaquer en fournissant des arguments,

4- phase tactique (cf. Figure 36) : le défenseur commence par défendre sa position, l'attaquant attaque les arguments du défenseur ; ce processus continue jusqu'à ce que les agents aient identifié leurs différences afin de les limiter ; les agents peuvent alors employer différentes tactiques : demande d'un délai, offre d'un compromis, insister sur ses arguments,

5- acheminement vers un accord : les agents offrent des compromis et contrent les arguments des autres,

6- résultat final : le résultat de la négociation peut être un compromis unilatéral, un compromis bilatéral, une attente ou un appel à l'autorité supérieure.

Ainsi, la phase initiale permet à l'attaquant de proposer un sujet de discussion au défenseur :

- Si le défenseur n'a pas la connaissance nécessaire, il envoie le message *Statut inconnu*, puis la conversation se termine.
- Sinon, le défenseur envoie un accusé de réception et le dialogue peut se poursuivre. Dans ce cas (où le défenseur possède les connaissances suffisantes), l'attaquant enchaîne par l'envoi d'une requête (cf. Figure 35). Le défenseur peut l'accepter ou la rejeter :
    - Si le défenseur accepte, les agents vont aboutir à un compromis.
    - S'il la rejette, les agents vont entrer dans la phase de retraite (aboutissant au rejet de la requête) ou dans la phase "tactique" (dans laquelle le défenseur a plusieurs choix, cf. Figure 36).

Par ailleurs, afin d'éviter toute boucle dans le protocole, les échanges du même type sont limités à cinq occurrences.

Par conséquent, dans tous les cas les agents aboutiront avec ce protocole :

- à un compromis offert par l'attaquant ou le défenseur,
- au rejet de la requête,

ou - au recours à un arbitre qui va trancher la question.

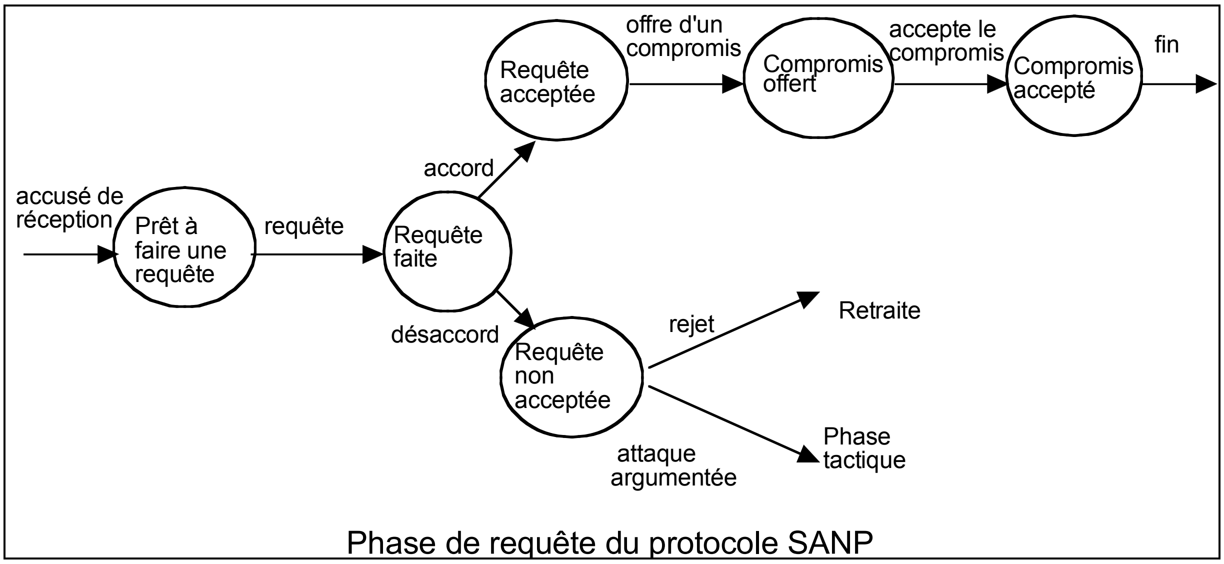

accusé de réception
Prêt à faire une requête
requête
Requête faite
accord
Requête acceptée
offre d'un compromis
Compromis offert
accepte le compromis
Compromis accepté
fin
désaccord
Requête non acceptée
rejet
Retraite
attaque argumentée
Phase tactique
Phase de requête du protocole SANP

Figure 35 : Protocole de Chang et Woo (Phase 2)

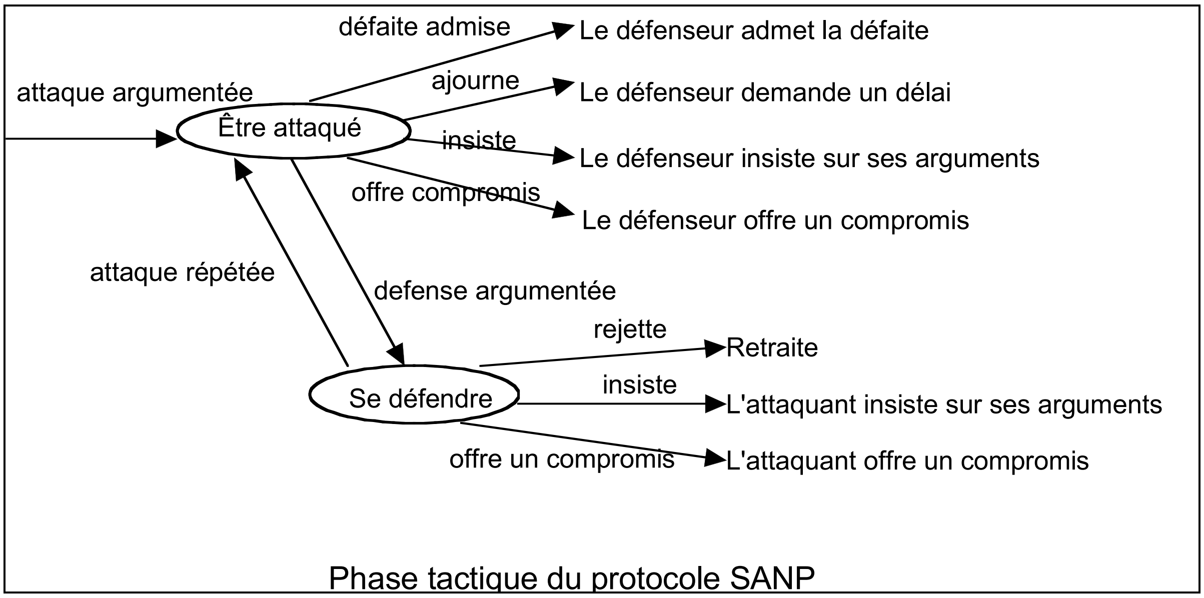

attaque argumentée
Être attaqué
défaite admise
Le défenseur admet la défaite
ajourne
Le défenseur demande un délai
insiste
Le défenseur insiste sur ses arguments
offre compromis
Le défenseur offre un compromis
attaque répétée
defense argumentée
Se défendre
rejette
Retraite
insiste
L'attaquant insiste sur ses arguments
offre un compromis
L'attaquant offre un compromis
Phase tactique du protocole SANP

Figure 36 : Protocole de Chang et Woo (Phase 4)

Ce protocole montre que les protocoles de communication sont des outils performants pour contrôler les échanges et guider les agents vers une solution finale sans contrôle centralisé. Cependant, ce protocole est complexe (langage de communication important et réponses possibles nombreuses), sa mise en œuvre paraît donc difficile.

8) Lander et Lesser : recherche négociée

Les auteurs [Lander & Lesser, 93] ont étudié le rôle de la négociation pour la résolution de conflits parmi des agents hétérogènes. Ils présentent un algorithme qui reconnaît explicitement et exploite les conflits.

Cet algorithme de recherche négociée utilise le conflit comme source d'informations de contrôle pour diriger les activités de recherche des agents, dans leur quête pour une solution mutuellement acceptable. L'algorithme est implémenté dans l'environnement multi-agents TEAM qui fournit l'infrastructure nécessaire à la communication et à la coopération. Les agents de TEAM sont coopératifs dans le sens où ils acceptent des solutions qui ne sont pas forcément localement optimales. Une proposition ne représente pas la solution complète mais une partie de la solution qui interagit avec d'autres. La proposition devient solution quand elle est acceptée par tous les agents.
Dans la recherche négociée, les agents intercalent les tâches suivantes :

- recherche locale d'une solution à certains sous-problèmes,
- intégration de solutions locales à des sous-problèmes (dans une solution partagée),
- négociation pour définir et affiner l'espace de recherche partagé des agents,
- évaluation et réévaluation des solutions émergeantes.

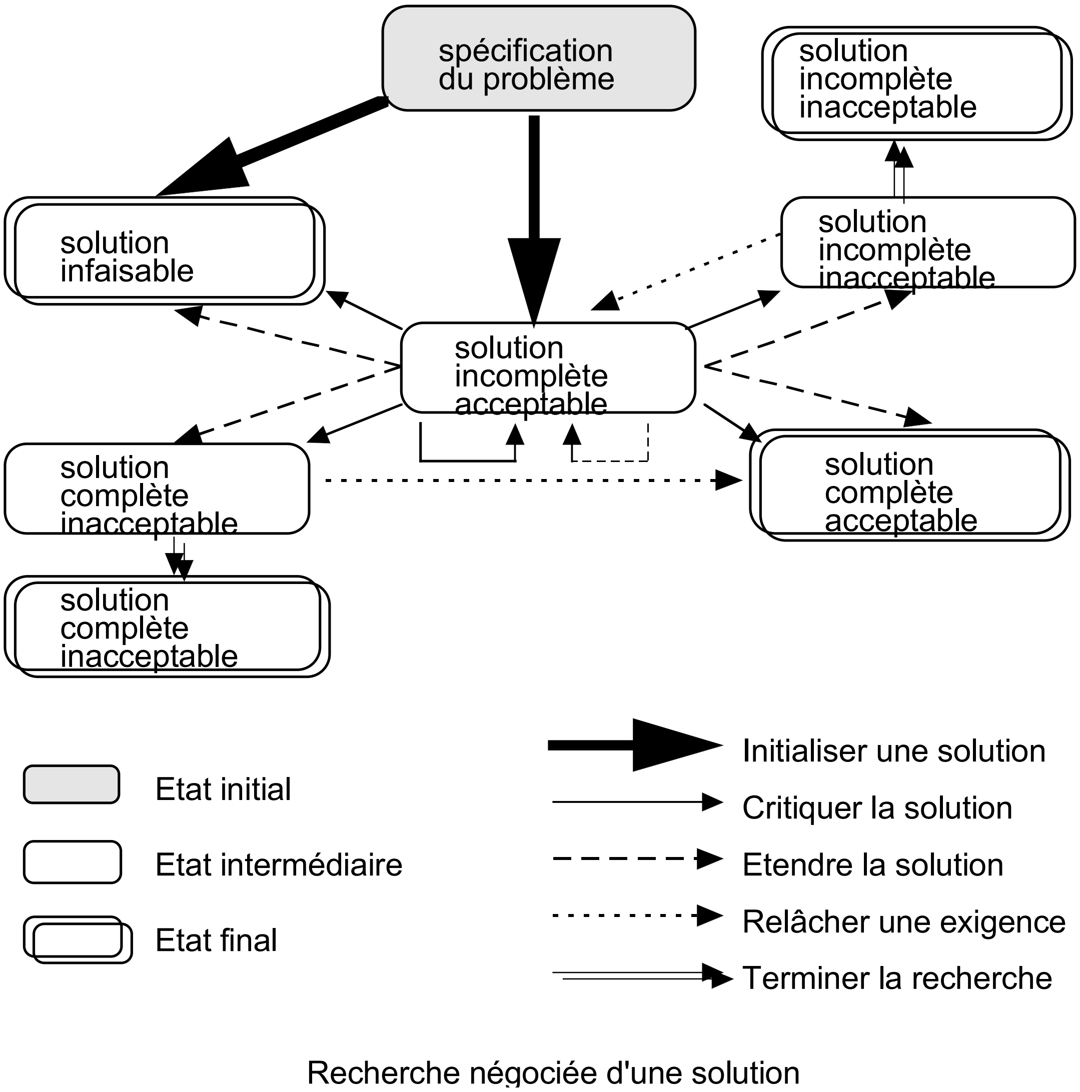
L'état d'une solution est défini par :

- son acceptabilité : acceptable, inacceptable ou infaisable,
- sa complétion : incomplète ou complète i.e. tous les agents ont pu évaluer (critiquer ou étendre) la solution,
- état de la recherche : initial ou final.

Les opérateurs de base de l'algorithme, définis par les auteurs, sont :

- *initiate-solution* (initialiser une solution) : cet opérateur est utilisé par un agent pour générer une proposition qui sera utilisée comme la base d'une nouvelle solution composée. Cette proposition de base est créée à partir d'exigences : exigences locales imposées par la spécification du problème, et des exigences externes apprises d'autres agents. Cet opérateur est exécuté par un ou plusieurs agents au démarrage du système, et peut éventuellement être exécuté de nouveau si les solutions proposées précédemment ont été rejetées ou si des solutions alternatives sont désirées.
- *critique-solution* (critiquer une solution) : cet opérateur est utilisé pour : évaluer une solution composée (sans générer de nouvelle composante) et indiquer en retour si des conflits sont détectés. En d'autres termes, un agent utilise cet opérateur pour évaluer partiellement ou complètement une solution composée spécifiée. L'entrée de cet opérateur est une solution composée initialisée par un autre agent. La sortie est une évaluation et, éventuellement, une information de conflit.
- *extend-solution* (étendre une solution) : cet opérateur est appliqué par un agent pour : ajouter une composante à une solution composée, évaluer la solution composée dans une perspective locale et fournir une information en retour si des conflits ont été détectés. Cet opérateur utilise en entrée une solution composée ayant été initialisée par un autre agent, et il fournit en sortie, une proposition, une évaluation ou une information de conflit (s'il existe).
- *relax-solution-requirement* (relâcher une exigence) : cet opérateur est appliqué pour choisir une exigence locale à relâcher, mettre à jour la base de données locale pour opérer le relâchement et réévaluer les solutions existantes à la lumière du relâchement. Les auteurs considèrent ce relâchement des exigences d'une solution comme une partie nécessaire de la recherche négociée.
- *terminate-search* (terminer la recherche) : cet opérateur est utilisé par TEAM pour changer la phase de recherche de l'algorithme de *initial* à *fermé* quand un certain nombre de propositions acceptables ont été trouvées.

Le passage d'une solution de l'état initial (la spécification du problème) à l'état final peut être représenté par un diagramme de transition d'états (cf. Figure 37).


spécification du problème
solution infaisable
solution incomplète inacceptable
solution incomplète inacceptable
solution incomplète acceptable
solution complète inacceptable
solution complète acceptable
solution complète inacceptable
Etat initial
Etat intermédiaire
Etat final
Initialiser une solution
Critiquer la solution
Etendre la solution
Relâcher une exigence
Terminer la recherche

Recherche négociée d'une solution

Figure 37 : Protocole de Lander et Lesser

Ce protocole permet de résoudre les conflits via la négociation ; il est simple grâce aux 5 opérateurs de base et aux 6 états possibles ; seules les issues suivantes sont possibles : obtention d'une solution complète acceptable, d'une solution infaisable ou d'une solution inacceptable (complète ou non).

9) Protocoles génériques de Burmeister, Haddadi, Sundermeyer

Les auteurs [Burmeister & al., 93] proposent des protocoles de coopération génériques fondés essentiellement sur les actes de langage. La sémantique d'un message est décrite par le contenu qui comprend :

- *le type du message* peut être *inform* (informe), *query* (demande) ou *command* (commande),
- *le descripteur*, qui renvoie à une ressource, un comportement (plan) ou une intention (but, rôle,…), est combiné au type du message,

- *le texte* est une application dépendant de la construction syntaxique de la proposition actuelle,
- *l'agent* correspond au nom de l'agent auquel se réfère la proposition (expéditeur, receveur ou un autre agent).

Les messages sont préparés et envoyés par des procédures d'envois. Pour chaque type de message, il existe une procédure de traitement dont la tâche est d'interpréter et de s'occuper de la réponse au message reçu. Les protocoles sont représentés d'une manière graphique ; le début est le nom du protocole avec quelques paramètres et l'arbre étiqueté représente les étapes possibles du dialogue dans ce protocole. Les nœuds de l'arbre représentent les états du dialogue, les liens représentent les transitions d'un état du dialogue à un autre, et les branches correspondent aux transitions alternatives. La racine du nœud d'un arbre de protocole caractérise le protocole. Les autres nœuds sont des appels à d'autres protocoles (sous-protocoles).

Les auteurs ont défini trois protocoles primitifs (cf. Figure 38) : *informing* (informer), *querying* (demander) et *commanding* (commander).

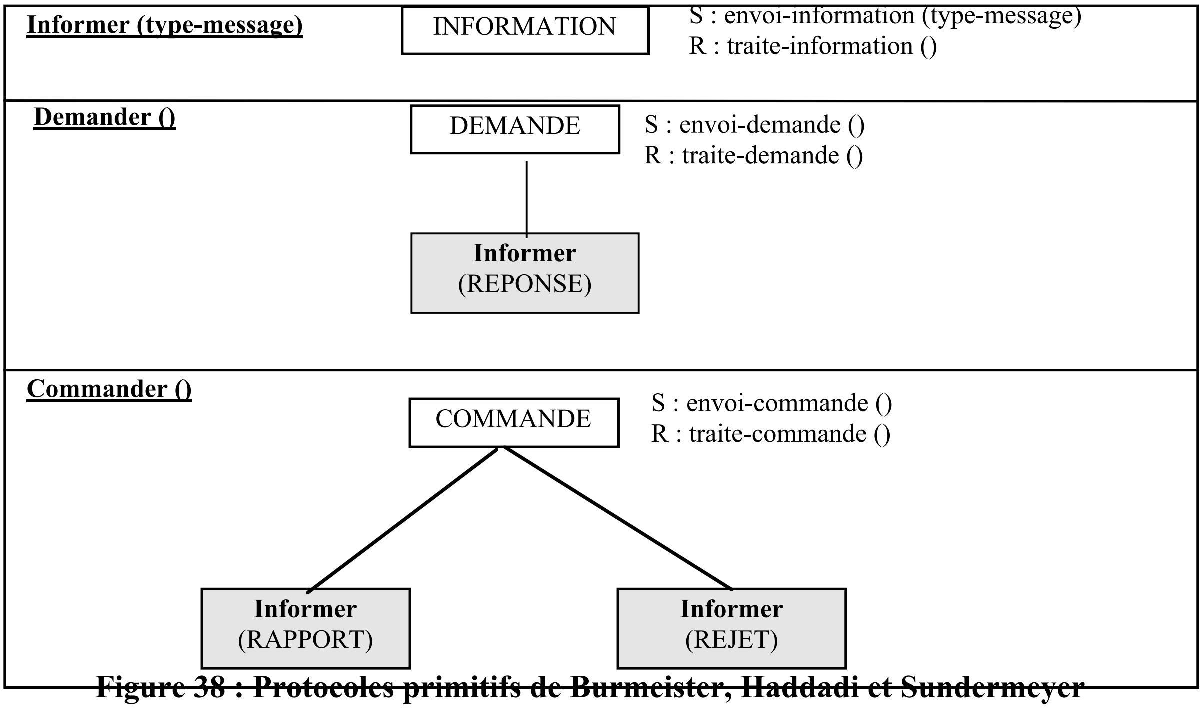

Informer (type-message)
INFORMATION
S : envoi-information (type-message)
R : traite-information ()
Demander ()
DEMANDE
S : envoi-demande ()
R : traite-demande ()
Informer (REPONSE)
Commander ()
COMMANDE
S : envoi-commande ()
R : traite-commande ()
Informer (RAPPORT)
Informer (REJET)

Figure 38 : Protocoles primitifs de Burmeister, Haddadi et Sundermeyer

Le protocole *informing* sert à fournir une information au destinataire. Le protocole *querying* est utilisé pour obtenir des informations. Quant au protocole *commanding*, il permet à un agent de commander l'exécution d'un comportement, par défaut, au destinataire.

Ces protocoles peuvent être utilisés tels quels pour de simples interactions ou peuvent faire parti de protocoles plus complexes. En effet, ils peuvent être utilisés pour construire des protocoles plus complexes (cf. Figure 39), comme *offering* (offrir), *requesting* (requérir), et *proposing* (proposer). Le protocole *requesting* ressemble au protocole *commanding* et le protocole *proposing* à *offering* sauf qu'ils (*requesting* et *proposing*) permettent la négociation.

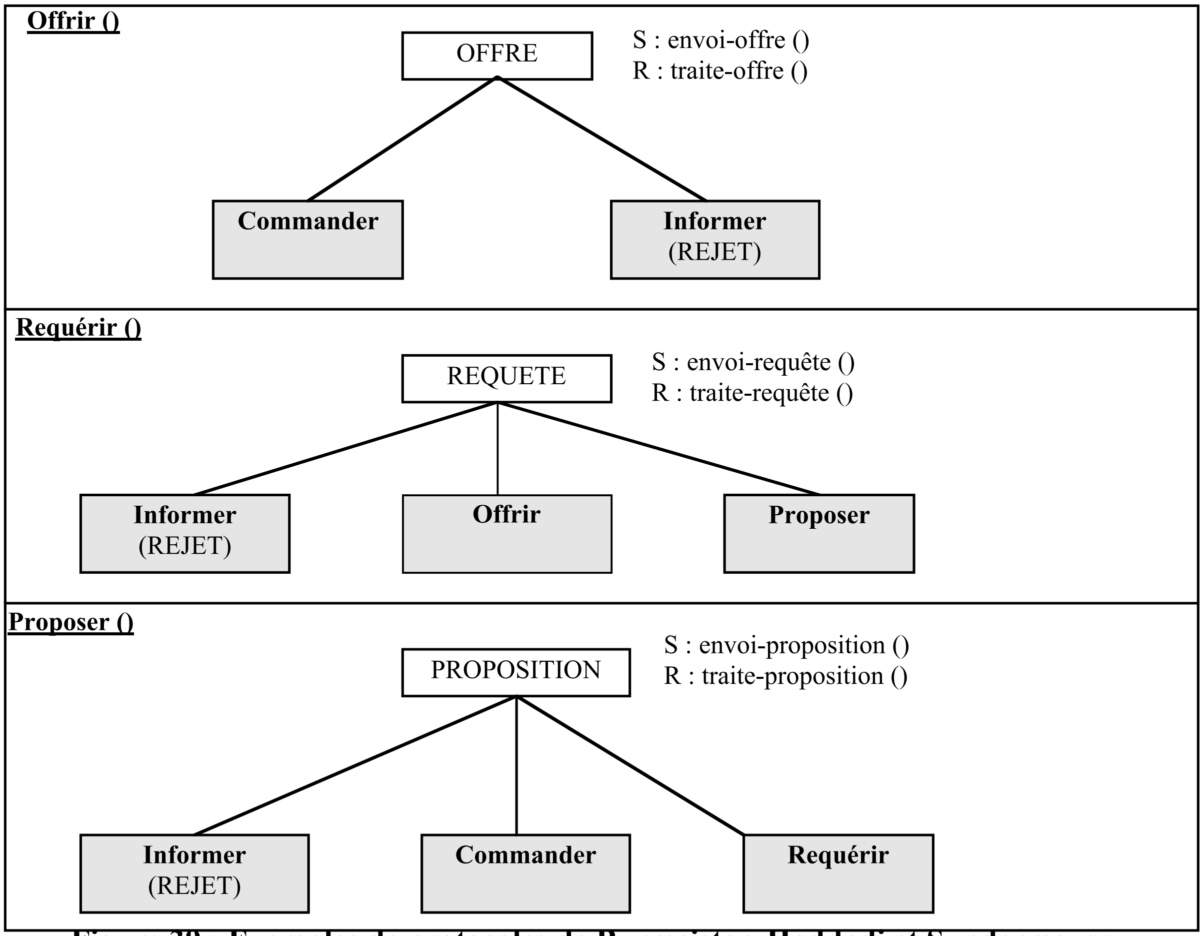

Offrir ()
OFFRE
S : envoi-offre ()
R : traite-offre ()
Commander
Informer
(REJET)
Requérir ()
REQUETE
S : envoi-requête ()
R : traite-requête ()
Informer
(REJET)
Offrir
Proposer
Proposer ()
PROPOSITION
S : envoi-proposition ()
R : traite-proposition ()
Informer
(REJET)
Commander
Requérir

Figure 39 : Exemples de protocoles de Burmeister, Haddadi et Sundermeyer

Les auteurs ont donc présenté des protocoles génériques, construits à partir des types de message et des procédures de traitement des messages. A partir des trois protocoles primitifs, il est aisé de construire d'autres protocoles plus complexes.

C) La grammaire hors-contexte de la proposition assertive

```
/***********************************************************/
/********* GRAMMAIRE DU SYNTAGME NOMINAL *********************/

/* REGLE 31 : N'' -> D' + N' */     /* "un serveur" */
clauses(sn(S0,S,Liste_regles,sn[Liste_synt]),
    [d1(S0,S1,Liste_regles1,Liste_synt1),
    n1(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 31 : SN -> D' + N'",
                 Liste_regles1,Liste_regles2]]).

/* REGLE 32 : N'' -> K + P + N'' */
/* REGLE 33 : N'' -> K */

/* REGLE 34 : N'' -> N'' + N'' */
clauses(sn(S0,S,Liste_regles,sn[Liste_synt]),/*"un reseau ethernet"*/
    [sn(S0,S1,Liste_regles1,Liste_synt1),
    sn(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 34 : SN -> SN + SN",
                 Liste_regles1,Liste_regles2]]).

/* REGLE 35 : N'' -> PIN */

/* REGLE 36 : N'' -> N' */  /* simple reecriture d'un nominal */
clauses(sn(S0,S,Liste_regles,sn[Liste_synt]),
    [n1(S0,S,Liste_regles1,Liste_synt),
    Liste_regles = ["Regle 36 : SN -> N'",Liste_regles1]]).

/* REGLE 37 : N' -> N + SP */   /* "la migation vers FDDI" */
clauses(n1(S0,S,Liste_regles,Liste_synt),
    [n(S0,S1,Liste_regles1,Liste_synt1),
    sp(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 37 : N' -> N + SP",
                 Liste_regles1,Liste_regles2]]).

/* REGLE 38 : N' -> N + SP + SP */
clauses(n1(S0,S,Liste_regles,Liste_synt),
    [n(S0,S2,Liste_regles1,Liste_synt1),
    sp(S2,S1,Liste_regles2,Liste_synt2),
    sp(S1,S,Liste_regles3,Liste_synt3),
    Liste_synt = [Liste_synt1,Liste_synt2,Liste_synt3],
    Liste_regles = ["Regle 38 : N' -> N + SP + SP",
                 Liste_regles1,Liste_regles2,Liste_regles3]]).
```

```
/* REGLE 39 : N' -> N + SP + SP + SP */
clauses(n1(S0,S,Liste_regles,Liste_synt),
    [n(S0,S3,Liste_regles1,Liste_synt1),
    sp(S3,S2,Liste_regles2,Liste_synt2),
    sp(S2,S1,Liste_regles3,Liste_synt3),
    sp(S1,S,Liste_regles4,Liste_synt4),
    Liste_synt = [Liste_synt1,Liste_synt2,Liste_synt3,Liste_synt4],
    Liste_regles = ["Regle 39 : N' -> N + SP + SP + SP",
    Liste_regles1,Liste_regles2,Liste_regles3,Liste_regles4]]).

/* REGLE 40 : N' -> N */
clauses(n1(S0,S,Liste_regles,Liste_synt),
    [n(S0,S,Liste_regles1,Liste_synt),
    Liste_regles = ["Regle 40 : N' -> N",Liste_regles1]]).

/* REGLE 43 : N -> N + SP */
clauses(n(S0,S,Liste_regles,Liste_synt),
    [n(S0,S1,Liste_regles1,Liste_synt1),
    sp(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 43 : N -> N + SP",
                Liste_regles1,Liste_regles2]]).

/* REGLE 44 : N -> A' + N */
clauses(n(S0,S,Liste_regles,Liste_synt),    /* "petite erreur" */
    [a1(S0,S1,Liste_regles1,Liste_synt1),
    n(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 44 : N -> A' + N",
                Liste_regles1,Liste_regles2]]).

/* REGLE 45 : N -> N + A''  */
clauses(n(S0,S,Liste_regles,Liste_synt),   /* "serveurs locaux" */
    [n(S0,S1,Liste_regles1,Liste_synt1),
    a2(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 45 : N -> N + A2",
                Liste_regles1,Liste_regles2]]).


/****************************************************************/
/*************** REGLES DU SYNTAGME PREPOSITIONNEL **************/

/* REGLE 46 : P -> W' + P  */
clauses(p(S0,S,Liste_regles,Liste_synt),    /* "près de la gaine" */
    [w1(S0,S1,Liste_regles1,Liste_synt1),
    p(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 46 : P -> W' + P",
                Liste_regles1,Liste_regles2]]).

/* REGLE 47 : SP -> P + SN */
clauses(sp(S0,S,Liste_regles,sp[Liste_synt]),
    [p(S0,S1,Liste_regles1,Liste_synt1),
    sn(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 47 : SP -> P + SN",
                Liste_regles1,Liste_regles2]]).

/* REGLE 48 : A'' -> A' + SP */
clauses(a2(S0,S,Liste_regles,Liste_synt),
    [a1(S0,S1,Liste_regles1,Liste_synt1),
    sp(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 48 : A'' -> A' + SP",
                Liste_regles1,Liste_regles2]]).

/* REGLE 49 : A'' -> A' + SP + SP */
clauses(a2(S0,S,Liste_regles,Liste_synt),
    [a1(S0,S2,Liste_regles1,Liste_synt1),
    sp(S2,S1,Liste_regles2,Liste_synt2),
    sp(S1,S,Liste_regles3,Liste_synt3),
    Liste_synt = [Liste_synt1,Liste_synt2,Liste_synt3],
    Liste_regles = ["Regle 49 : A'' -> A' + SP + SP",
                Liste_regles1,Liste_regles2,Liste_regles3]]).

/* REGLE 50 : A'' -> A' */
clauses(a2(S0,S,Liste_regles,Liste_synt),
    [a1(S0,S,Liste_regles1,Liste_synt),
    Liste_regles = ["Regle 50 : A'' -> A'",Liste_regles1]]).

/* REGLE 51 : A' -> W' + A */
clauses(a1(S0,S,Liste_regles,Liste_synt),
    [w1(S0,S1,Liste_regles1,Liste_synt1),
    a(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 51 : A' -> W' + A",
                Liste_regles1,Liste_regles2]]).

/* REGLE 52 : A' -> A */
clauses(a1(S0,S,Liste_regles,Liste_synt),
    [a(S0,S,Liste_regles1,Liste_synt),
    Liste_regles = ["Regle 52 : A' -> A",Liste_regles1]]).

/* REGLE 55 : D' -> D + D */
clauses(d1(S0,S,Liste_regles,Liste_synt),
    [d(S0,S1,Liste_regles1,Liste_synt1),
    d(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 55 : D' -> D + D",
                Liste_regles1,Liste_regles2]]).

/* REGLE 56 : D' -> P + D  (ou` P="de" + D="le,la,les") */
clauses(d1(S0,S,Liste_regles,Liste_synt),
    [p_de(S0,S1,Liste_regles1,Liste_synt1),
    d_l(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 56 : D' -> P=de + D=le,la,les",
          Liste_regles1,Liste_regles2]]).

/* REGLE 57 : D' -> D   */
clauses(d1(S0,S,Liste_regles,Liste_synt),
    [d(S0,S,Liste_regles1,Liste_synt),
    Liste_regles = ["Regle 57 : D' -> D",Liste_regles1]]).

/* REGLE 58 : K -> W  */
/* REGLE 59 : K -> D + N   */
/* REGLE 60 : K -> D + A'   */

/* REGLE 61 : W' -> W + W' */
clauses(w1(S0,S,Liste_regles,Liste_synt),
    [w(S0,S1,Liste_regles1,Liste_synt1),
    w1(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 61 : W' -> W + W'",
                  Liste_regles1,Liste_regles2]]).

/* REGLE 62 : W' -> W  */
clauses(w1(S0,S,Liste_regles,Liste_synt),
    [w(S0,S,Liste_regles1,Liste_synt),
    Liste_regles = ["Regle 62 : W' -> W",Liste_regles1]]).


/*************************************************************/
/****************** GRAMMAIRE DU SYNTAGME VERBAL *************/

/* REGLE 12 : SV -> Y[IN1] + V'' */
clauses(sv(S0,S,Liste_regles,sv[Liste_synt]),
    [yin1(S0,S1,Liste_regles1,Liste_synt1),
    v2(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 12 : SV -> Y[IN1] + V''",
                  Liste_regles1,Liste_regles2]]).

/* REGLE 13 : SV -> V'' */
clauses(sv(S0,S,Liste_regles,sv[Liste_synt]),
    [v2(S0,S,Liste_regles1,Liste_synt),
    Liste_regles = ["Regle 13 : SV -> V''",Liste_regles1]]).

/* REGLE 14 : V'' -> W[NEG] + V'' */
clauses(v2(S0,S,Liste_regles,Liste_synt),
    [wneg(S0,S1,Liste_regles1,Liste_synt1),
    v2(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 14 : V'' -> W[NEG] + V''",
                  Liste_regles1,Liste_regles2]]).

/* REGLE 15 : V'' -> V'' + N'' */
clauses(v2(S0,S,Liste_regles,Liste_synt),
    [v2(S0,S1,Liste_regles1,Liste_synt1),
    sn(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 15 : V'' -> V'' + SN",
                  Liste_regles1,Liste_regles2]]).

/* REGLE 16 : V'' -> V'' + SP */
clauses(v2(S0,S,Liste_regles,Liste_synt),
    [v2(S0,S1,Liste_regles1,Liste_synt1),
    sp(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 16 : V'' -> V'' + SP",
                  Liste_regles1,Liste_regles2]]).

/* REGLE 17 : V'' -> V'' + W' */
clauses(v2(S0,S,Liste_regles,Liste_synt),
    [v2(S0,S1,Liste_regles1,Liste_synt1),
    w1(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 17 : V'' -> V'' + W'",
                  Liste_regles1,Liste_regles2]]).

/* REGLE 18 : V'' -> V'' + A'' */
clauses(v2(S0,S,Liste_regles,Liste_synt),
    [v2(S0,S1,Liste_regles1,Liste_synt1),
    a2(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 18 : V'' -> V'' + A''",
                 Liste_regles1,Liste_regles2]]).

/* REGLE 19 : V'' -> V'' + PIN (je vois jouer) */

/* REGLE 20 : V'' -> V''[MOD] + V''[INF] */
clauses(v2(S0,S,Liste_regles,Liste_synt),   /* je dois enlever */
    [v2(S0,S1,Liste_regles1,Liste_synt1),
    v2(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 20 : V'' -> V'' + V''",
                 Liste_regles1,Liste_regles2]]).

/* REGLE 21 : V'' -> Y + V' */
clauses(v2(S0,S,Liste_regles,Liste_synt),
    [y(S0,S1,Liste_regles1,Liste_synt1),
    v1(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 21 : V'' -> Y + V'",
                 Liste_regles1,Liste_regles2]]).

/* REGLE 22 : V'' -> Y + Y + V' */
clauses(v2(S0,S,Liste_regles,Liste_synt),
    [y(S0,S2,Liste_regles1,Liste_synt1),
    y(S2,S1,Liste_regles2,Liste_synt2),
    v1(S1,S,Liste_regles3,Liste_synt3),
    Liste_synt= [Liste_synt1,Liste_synt2,Liste_synt3],
    Liste_regles = ["Regle 22 : V'' -> Y + Y + V'",
                 Liste_regles1,Liste_regles2,Liste_regles3]]).

/*  REGLE  23  :  V''  ->  V'(IMP)  +  Y  */  /*  regle  modifiee  V'(IMP)
devient V(IMP) */
clauses(v2(S0,S,Liste_regles,Liste_synt),
    [vimp(S0,S1,Liste_regles1,Liste_synt1),
    y(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt= [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 23 : V'' -> V(IMP) + Y",
                 Liste_regles1,Liste_regles2]]).

/* REGLE 24 : V'' -> V'(IMP) + Y + Y */
/* regle modifiee V'(IMP) devient V(IMP) */
clauses(v2(S0,S,Liste_regles,Liste_synt),
    [vimp(S0,S2,Liste_regles1,Liste_synt1),
    y(S2,S1,Liste_regles2,Liste_synt2),
    y(S1,S,Liste_regles3,Liste_synt3),
    Liste_synt= [Liste_synt1,Liste_synt2,Liste_synt3],
    Liste_regles = ["Regle 24 : V'' -> V(IMP) + Y + Y",
                 Liste_regles1,Liste_regles2,Liste_regles3]]).

/* REGLE 25 : V'' -> V' */
clauses(v2(S0,S,Liste_regles,Liste_synt),
    [v1(S0,S,Liste_regles1,Liste_synt),
    Liste_regles = ["Regle 25 : V'' -> V'",Liste_regles1]]).

/* REGLE 26 : V' -> V' + Y[IN1] */
clauses(v1(S0,S,Liste_regles,Liste_synt),
    [v1(S0,S1,Liste_regles1,Liste_synt1),
    yin1(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 26 : V' -> V' + Y[IN1]",
                Liste_regles1,Liste_regles2]]).

/* REGLE 27 : V' -> V + V */
clauses(v1(S0,S,Liste_regles,Liste_synt),
    [v(S0,S1,Liste_regles1,Liste_synt1),
    v(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 27 : V' -> V + V",
                Liste_regles1,Liste_regles2]]).

/* REGLE 28 : V' -> V */
clauses(v1(S0,S,Liste_regles,Liste_synt),
    [v(S0,S,Liste_regles1,Liste_synt),
    Liste_regles = ["Regle 28 : V' -> V",Liste_regles1]]).

/* REGLE 29 : PIN -> SV[INF] (je préfère "le garder") */
/* REGLE 30 : PIN -> Q + SV[INF] (... "de relier des appareils") */


/************************************************************/
/****************** GRAMMAIRE DE LA PROPOSITION **************/

/* REGLE 1 : PP''-> W[PRO] PP' */
clauses(prop2(S0,S,Liste_regles,proposition[Liste_synt]),
    [wpro(S0,S1,Liste_regles1,Liste_synt1),
    prop1(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 1 : PP''-> W[PRO] PP'",
                Liste_regles1,Liste_regles2]]).

/* REGLE 2 : PP''-> W[PRO] T[TVP2] PP'' */
clauses(prop2(S0,S,Liste_regles,proposition[Liste_synt]),
    [wpro(S0,S2,Liste_regles1,Liste_synt1),
    tvp2(S2,S1,Liste_regles2,Liste_synt2),
    prop1(S1,S,Liste_regles3,Liste_synt3),
    Liste_synt = [Liste_synt1,Liste_synt2,Liste_synt3],
    Liste_regles = ["Regle 2 : PP''-> W[PRO] T[TVP2] PP''",
                Liste_regles1,Liste_regles2,Liste_regles3]]).

/* REGLE 3 : PP''-> PP' */
clauses(prop2(S0,S,Liste_regles,proposition[Liste_synt]),
    [prop1(S0,S,Liste_regles1,Liste_synt),
    Liste_regles = ["Regle 3 : PP''-> PP'",Liste_regles1]]).

/* REGLE 4 : PP'-> SP + PP */
clauses(prop1(S0,S,Liste_regles,Liste_synt),
    [sp(S0,S1,Liste_regles1,Liste_synt1),
    prop0(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 4 : PP'-> SP + PP",
                Liste_regles1,Liste_regles2]]).

/* REGLE 5 : PP'-> SP + T[VP2] + PP */
clauses(prop1(S0,S,Liste_regles,Liste_synt),
    [sp(S0,S2,Liste_regles1,Liste_synt1),
    tvp2(S2,S1,Liste_regles2,Liste_synt2),
    prop0(S1,S,Liste_regles3,Liste_synt3),
    Liste_synt = [Liste_synt1,Liste_synt2,Liste_synt3],
    Liste_regles = ["Regle 5 : PP'-> SP + T[VP2] + PP",
                 Liste_regles1,Liste_regles2,Liste_regles3]]).

/* REGLE 6 : PP'-> W' + PP */
clauses(prop1(S0,S,Liste_regles,Liste_synt),
    [w1(S0,S1,Liste_regles1,Liste_synt1),
    prop0(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 6 : PP'-> W' + PP",
                 Liste_regles1,Liste_regles2]]).

/* REGLE 7 : PP'-> W' T[VP2] PP */
clauses(prop1(S0,S,Liste_regles,Liste_synt),
    [w1(S0,S2,Liste_regles1,Liste_synt1),
    tvp2(S2,S1,Liste_regles2,Liste_synt2),
    prop0(S1,S,Liste_regles3,Liste_synt3),
    Liste_synt = [Liste_synt1,Liste_synt2,Liste_synt3],
    Liste_regles = ["Regle 7 : PP'-> W' T[VP2] PP",
                 Liste_regles1,Liste_regles2,Liste_regles3]]).

/* REGLE 8 : PP'-> PP */
clauses(prop1(S0,S,Liste_regles,Liste_synt),
    [prop0(S0,S,Liste_regles1,Liste_synt),
    Liste_regles = ["Regle 8 : PP'-> PP",Liste_regles1]]).

/* REGLE 9 : PP-> SV N'' */
clauses(prop0(S0,S,Liste_regles,Liste_synt),
    [sv(S0,S1,Liste_regles1,Liste_synt1),
    sn(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 9 : PP-> SV N''",
                 Liste_regles1,Liste_regles2]]).

/* REGLE 10 : PP-> N'' SV */
clauses(prop0(S0,S,Liste_regles,Liste_synt),
    [sn(S0,S1,Liste_regles1,Liste_synt1),
    sv(S1,S,Liste_regles2,Liste_synt2),
    Liste_synt = [Liste_synt1,Liste_synt2],
    Liste_regles = ["Regle 10 : PP-> N'' SV",
                 Liste_regles1,Liste_regles2]]).

/* REGLE 11 : PP-> SV */
clauses(prop0(S0,S,Liste_regles,Liste_synt),
    [sv(S0,S,Liste_regles1,Liste_synt),
    Liste_regles = ["Regle 11 : PP-> SV",Liste_regles1]]).


/*********************************************************/
/************* REGLES DE RECONNAISSANCE DES FORMES ***********/

/* REGLE 41 : N -> F[NOM] */
clauses(n([f,Forme,Var|S],S,"Regle 41 : N -> F[NOM]",fnom[Forme]),
[dedans(nom,Var)]).

/* REGLE 42 : N -> F[NAN] */
clauses(n([f,Forme,Var|S],S,"Regle 42 : N -> F[NAN]",fnom[Forme]),
[dedans(nan,Var)]).

/* REGLE 53 : A -> F[ADJ] */
clauses(a([f,Forme,Var|S],S,"Regle 53 : A -> F[ADJ]",fadj[Forme]),
[dedans(adj,Var)]).
```

```
/* REGLE 54 : A -> F[NAN] */
clauses(a([f,Forme,Var|S],S,"Regle 54  :  A  ->  F[NAN]",fadj[Forme]),
[dedans(nan,Var)]).

clauses(vimp([v,Forme,Var|S],S,"Regle de reconnaissance d'un verbe
imperatif",v[Forme]),[dedans(ipf,Var)]).

clauses(yin1([y,Forme,Var|S],S,"Regle  de  reconnaissance  d'un
preverbal in1",y[Forme]),[dedans(in1,Var)]).

clauses(wneg([w,Forme,Var|S],S,"Regle de reconnaissance d'un adverbe
negatif",w[Forme]),[dedans(neg,Var)]).

clauses(wpro([w,Forme,Var|S],S,[],w[Forme]),
                [dedans(pro,Var)]).

clauses(tvp1([t,Forme,Var|S],S,[],t[Forme]),
                [dedans(vp1,Var)]).

clauses(tvp2([t,Forme,Var|S],S,[],t[Forme]),
                [dedans(vp2,Var)]).


/***********************************************************/
/******** PRIMITIVES (definies pour la meta-programmation)*****/

primitive(write(_)).
primitive(line).
primitive(outml(_)).
primitive(=(_,_)).

primitive(dedans(_,_)).

primitive(p(_,_,_,_)).
primitive(d(_,_,_,_)).
primitive(p_de(_,_,_,_)).
primitive(d_l(_,_,_,_)).
primitive(w(_,_,_,_)).
primitive(v(_,_,_,_)).
primitive(y(_,_,_,_)).
primitive(t(_,_,_,_)).

/***********************************************************/
/*************************** LES FAITS **********************/

v([v,Forme,Var|S],S,"Regle de reconnaissance d'un verbe",v[Forme]).
y([y,Forme,Var|S],S,"Regle de reconnaissance d'un preverbal",
       y[Forme]).
w([w,Forme,Var|S],S,"Regle de reconnaissance d'un adverbe",w[Forme]).
p_de([p,"de",Var|S],S,"Regle de reconnaissance de  P=de",p[de]).
d_l([d,"le",Var|S],S,"Regle de reconnaissance de D=le",d[le]).
d_l([d,"la",Var|S],S,"Regle de reconnaissance de D=la",d[la]).
d_l([d,"les",Var|S],S,"Regle de reconnaissance de D=les",d[les]).
p([p,Forme,Var|S],S,"Regle de reconnaissance d'une preposition",
    p[Forme]).
d([d,Forme,Var|S],S,"Regle de reconnaissance d'un determinant",
    d[Forme]).
t([t,Forme,Var|S],S,"Regle de reconnaissance d'une ponctuation",
    t[Forme]).
```

D) Phrases tests pour la désambiguïsation morphologique

Phrase 1 : La seule incertitude concerne l'ordre hiérarchique dans lequel les deux hommes seront sans doute appelés à travailler ensemble.
Phrase 2 : Forte de ses positions majoritaires, l'UC, pour l'instant, fait la sourde oreille.
Phrase 3 : Toute l'opposition, du centre droit à l'extrême gauche, est visée.
Phrase 4 : La veille, des unités de la garde présidentielle s'étaient emparées, souvent brutalement, d'une quarantaine de responsables et de militants politiques.
Phrase 5 : Les rues de Port-au-Prince sont restées calmes dimanche et aucun dispositif militaire particulier n'a été mis en place.
Phrase 6 : La peur face à des bandes de civils armés qui sont réapparues dans Port-au-Prince explique aussi la prudence du plus grand nombre.
Phrase 7 : Le GEPI offre un éventail très large de spécialités et permet de mettre des moyens pédagogiques en commun.
Phrase 8 : La croissance s'est ralentie dans les biens d'équipements professionnels et les biens de consommation, et s'est interrompue dans les autres secteurs.
Phrase 9 : Les capacités de production restent néanmoins fortement sollicitées, ajoute la Banque de France.
Phrase 10 : La demande s'est, dans l'ensemble, stabilisée malgré la fermeté des commandes étrangères.
Phrase 11 : Sur le marché intérieur, elle s'est surtout affaiblie dans l'automobile.
Phrase 12 : Moins de deux mois avant le congrès du Parti socialiste, à Rennes, Mr Fabius bat la campagne.
Phrase 13 : Même s'il lui a été reproché de confisquer cet anniversaire, près de neuf mille militants s'étaient réunis autour de lui.
Phrase 14 : La scène remonte au début des années 30.
Phrase 15 : Les poissons sont de taille.
Phrase 16 : L'usine, qui devrait être implantée à Eloyes (Vosges) représente un investissement d'environ 3,7 milliards de yens.
Phrase 17 : Elle fabriquera, dans un premier temps, le produit liquide qui entre dans le processus des photocopies ainsi que des pièces détachées pour la filiale de Minolta en RFA.
Phrase 18 : A partir de 1992, cette usine devrait se lancer dans la fabrication de photocopieurs et d'imprimantes.
Phrase 19 : Lors de son ouverture, elle emploiera une centaine de personnes.
Phrase 20 : A Palavas, il s'agit simplement d'embellir l'entrée de la station par la création d'un port sur le Lez de 30 anneaux, dont les travaux démarrent tout juste.
Phrase 21 : L'inauguration est programmée pour l'ouverture de la prochaine saison.
Phrase 22 : En amont du fleuve, à Lattes, le projet de Port-Ariane est plus ambitieux.
Phrase 23 : Les sondages archéologiques terminés, une société d'économie mixte sera créée fin janvier.
Phrase 24 : Les premiers coups de pioche devraient être donnés à la fin de l'année pour une première tranche commercialisée en 1992.
Phrase 25 : A Montpellier, la dimension du projet interdit toute échéance précise.

Phrase 26 : La somptueuse maquette de Ricardo Bofill exposée auparavant dans le grand hall de la mairie a été rangée au placard.
Phrase 27 : Entre-temps, l'architecte catalan, initialement annoncé comme responsable de la réalisation de Port-Mariane, s'est vu récusé pour ce poste.
Phrase 28 : Le nombre d'appartements effectivement réalisés dépendra du rythme de croissance de la ville : entre 10 000 et 30 000 habitants, selon les prévisions.
Phrase 29 : Reste les problèmes techniques à résoudre pour rendre navigable un fleuve qui, après deux mille ans d'histoire, a cessé de l'être au début du siècle.
Phrase 30 : Impossible en fait de ne pas envisager de négociations, tant les intérêts des deux communes sont liés.
Phrase 31 : A Lattes, la commune a convaincu le principal propriétaire de vendre.
Phrase 32 : Une apparente confiance qui reste cependant suspendue à un avis dont personne ne sait quand il sera rendu.
Phrase 33 : Sept cents exemplaires de ces couteaux de luxe seront produits chaque année en sept modèles différents.
Phrase 34 : La chorégraphe a décidé de nous emmener sur des sommets, on lui souhaite bon courage.
Phrase 35 : Plusieurs moments d'une grâce touchante ne suffisent pas à la sauver.
Phrase 36 : La pièce est composée de deux parties trop distinctes et le propos devient manichéen.
Phrase 37 : Toutefois, le plaisir de céder ou de résister à la tentation semble finalement bien tiède.
Phrase 38 : On aurait aimé plus de force et moins de délicatesse, plus de chair et moins d'éther.
Phrase 39 : Soixante-quinze animateurs recrutés et rémunérés par la municipalité ont déjà été formés pour y intervenir.
Phrase 40 : Ces mesures devraient être appliquées dès les prochaines vacances de février.
Phrase 41 : Tout prédestinait, il est vrai, cet archétype de l'aristocratie financière basque à occuper une telle fonction.
Phrase 42 : Cette possibilité n'a été accordée que pour les obligations d'une durée inférieure à six mois.
Phrase 43 : Les voitures circulant dans les rues sont rares, le blocus qu'imposent les Arméniens empêchant les livraisons d'essence.
Phrase 44 : Les bruits les plus fous circulent.
Phrase 45 : Le Nakhitchevan est presque coupé du monde.
Phrase 46 : Ses frontières sont difficilement franchissables.
Phrase 47 : Les communications téléphoniques sont interrompues.
Phrase 48 : Les Arméniens ont saboté le relais de télévision qui leur permettait de recevoir les images soviétiques.
Phrase 49 : C'est par avion qu'arrive l'essentiel du ravitaillement, en provenance de Bakou.
Phrase 50 : La France, malgré sa situation économique favorable, n'a pas pu rester insensible à ces dépressions.
Phrase 51 : Le mouvement se propageait immédiatement sur les autres places.
Phrase 52 : Aucune de ces perspectives n'a provoqué de réactions vraiment visibles.
Phrase 53 : Ces derniers démentaient être à l'origine des achats.
Phrase 54 : Ces achats ne visent pas à prendre le contrôle du Club, ont précisé ces dirigeants, reconnaissant que les deux sociétés ont de nombreuses synergies possibles.
Phrase 55 : Les trois hommes avaient été condamnés notamment pour des agressions à main armée.

Phrase 56 : Deux sessions complémentaires doivent réunir leurs subordonnés directs dans la capitale autrichienne, sur des sujets plus techniques.
Phrase 57 : On dit que le général Moisseiev, qui recevra en avril prochain son homologue français en visite officielle à Moscou, en a pris note.
Phrase 58 : De tous les alliés de l'URSS, c' est la Pologne qui est apparue le plus acquis à la transparence.
Phrase 59 : Dans leurs luttes pour la démocratie, la plupart des pays de l'Est se sont appuyés sur leur identité nationale.
Phrase 60 : Ce fils de pasteur de trente-quatre ans, né à Leipzig, écrit, organise des expositions et des concerts depuis une bonne dizaine d'années.
Phrase 61 : En revanche, depuis deux ans, dans quelques discothèques, des disc-jockeys acceptaient de diffuser ces cassettes autoproduites.
Phrase 62 : Une culture s'épanouissait discrètement en vase clos.
Phrase 63 : C'était un cadeau mais aussi un hommage à leur pugnacité.
Phrase 64 : En octobre dernier, un rassemblement lancé à l' initiative d'un certain nombre de musiciens a été ponctué de discours significatifs.
Phrase 65 : Des échanges ont déjà lieu entre les deux Allemagnes.
Phrase 66 : Nous sommes fiers d'avoir tenu dans une situation difficile.
Phrase 67 : L'ouverture, c' est un espoir et une crainte.
Phrase 68 : Si nous sommes capables d'articuler notre identité dans un domaine, nous serons forts.
Phrase 69 : La NATPE est d'abord un marché de syndication.
Phrase 70 : Mais la loi les en empêche depuis vingt ans.
Phrase 71 : De plus, la stricte séparation entre production et diffusion leur interdit de pratiquer la syndication.
Phrase 72 : Producteurs et vendeurs ne l'entendent évidemment pas de cette oreille.
Phrase 73 : Son voyage en Europe avait une forte dimension électorale.
Phrase 74 : L'exercice est délicat pour un pays aussi fier, mais il est indispensable.
Phrase 75 : Quand il était ministre de la défense, manifestement il était poussé par le bien de la France.
Phrase 76 : Il a été très surpris par les événements, très peiné.
Phrase 77 : Assister à un tel bouleversement me rend heureuse de vivre à l'époque où je vis.

E) Quelques propositions "simples" testées sur TALISMAN II

Proposition 1 : La seule incertitude concerne l'ordre hiérarchique.
Proposition 2 : Des échanges ont lieu entre les deux Allemagnes.
Proposition 3 : L'opposition est visée.
Proposition 4 : Une culture s'épanouissait discrètement en vase clos.
Proposition 5 : La peur explique aussi la prudence.
Proposition 6 : Ce fils de pasteur organise des expositions.
Proposition 7 : L'usine représente un investissement.
Proposition 8 : Le produit entre dans le processus des photocopies.
Proposition 9 : Les travaux démarrent.
Proposition 10 : On lui souhaite bon courage.
Proposition 11 : Le propos devient manichéen.
Proposition 12 : Les voitures circulant dans les rues sont rares.
Proposition 13 : Ses frontières sont difficilement franchissables.
Proposition 14 : Les communications téléphoniques sont interrompues.
Proposition 15 : Les Arméniens ont saboté le relais de télévision.
Proposition 16 : Le ravitaillement en provenance de Bakou arrive par avion.
Proposition 17 : La séparation leur interdit la syndication.
Proposition 18 : Une première tranche sera commercialisée en 1992.
Proposition 19 : C'était un cadeau.
Proposition 20 : Son voyage en Europe avait une forte dimension électorale.
Proposition 21 : Il est indispensable.
Proposition 22 : Il était ministre de la défense.
Proposition 23 : L'UC fait la sourde oreille.
Proposition 24 : Le GEPI offre un éventail de spécialités.
Proposition 25 : La dimension du projet interdit toute échéance précise.
Proposition 26 : Les intérêts des deux communes sont liés.