Financial information extraction using pre-defined and user-definable templates in the Lolita system

Constantino, Marco

University of Durham

Financial Information Extraction using pre-defined and user-definable Templates in the LOLITA System.

Marco Costantino

Laboratory for Natural Language Engineering,

Department of Computer Science.

Submitted for the degree of

Doctor of Philosophy

Abstract

Financial operators have today access to an extremely large amount of data, both quantitative and qualitative, real-time or historical and can use this information to support their decision-making process.

Quantitative data are largely processed by automatic computer programs, often based on artificial intelligence techniques, that produce quantitative analysis, such as historical price analysis or technical analysis of price behaviour. Differently, little progress has been made in the processing of qualitative data, which mainly consists of financial news articles from financial newspapers or on-line news providers. As a result the financial market players are overloaded with qualitative information which is potentially extremely useful but, due to the lack of time, is often ignored.

The goal of this work is to reduce the qualitative data-overload of the financial operators. The research involves the identification of the information in the source financial articles which is relevant for the financial operators' investment decision making process and to implement the associated templates in the LOLITA system. The system should process a large number of source articles and extract specific templates according to the relevant information located in the source articles.

The project also involves the design and implementation in LOLITA of a user-definable template interface for allowing the users to *easily* design new templates using sentences in natural language. This allows user-defined information extraction from source texts. This differs from most of existing information extraction systems which require the developers to code the templates directly in the system.

The results of the research have shown that the system performed well in the extraction of financial templates from source articles which would allow the financial operator to reduce his qualitative data-overload. The results have also shown that the user-definable template interface is a viable approach to user-defined information extraction. A trade-off has been identified between the *ease of use* of the user-definable template interface and the *loss of performance* compared to hand-coded templates.

Acknowledgements

I would like to thank my supervisor Richard G. Morgan for his advice and support throughout the three years over which this research has been conducted. I would also like to thank Russell J. Collingham for his help and support and Stephen Eckett for his invaluable advice on the financial aspects of the thesis.

I am also grateful for all past and present members of the Laboratory for Natural Language Engineering who have helped provide such a pleasant research and social environment. Thank you to all those people who commented on the various drafts of this thesis, particularly Luisa Mich and Luigi Colazzo.

Financial assistance for this project was received from the Opera Universitaria Society of the University of Trento. I would also like to thank the department of Economics of the University of Trento for awarding me the University of Trento studentship for postgraduate studies abroad.

I would also like to thank you all my friends who supported me during the these three years and, particularly, Edy, Mika, Marianna, Paolo, Wendy, Janice, Giacomo, Max, Xavier and Sergio.

Finally I would like to thank my Mum for her support during my time at the University.

Declaration

The material contained within this thesis has not previously been submitted for a degree at the University of Durham or any other university. The research reported within this thesis has been conducted by the author unless indicated otherwise.

The copyright of this thesis rests with the author. No quotation from it should be published without his prior written consent and information derived from it should be acknowledged.

Contents

List of Figures

Chapter 1

Methodological Introduction

1.1 Introduction

The goal of *information extraction* is to extract specific kinds of information from a source document [Riloff and Lehnert, 1994]. In other words, the input to the system is a document (e.g. a newspaper article), while the output consists of a summary of the contents of the document. For example, the source article could consist of the following text:

BELL ATLANTIC announced it will acquire Tele-Communications Inc with 18 billion dollars. The deal will radically change the US communications industry. It will also be one of the country's biggest ever takeovers. The deal - which is bound to face strong regulatory scrutiny - would be the first full merger between a US telephone company and a cable business at a time when the two industries are converging to create a single, multi-media inter-active entertainment and information business.

An information extraction system will try to produce a template of the original text according to a specification of the information defined during the design of the system. For example, a sensible summary of the article shown above could be:

```
Template: TAKEOVER
    COMPANY_TARGET: Tele-Communications Inc.
    COMPANY_PREDATOR: BELL ATLANTIC
    TYPE_TAKEOVER: FRIENDLY
    VALUE: 18 billion dollars.
    ATTRIBUTION: BELL ATLANTIC
```

The summary presented above is called a *template* which is a structure with a predefined number of slots, where the slots are filled with the information extracted from the source article.

Players in the financial market have today access to an extremely large amount of data, both quantitative and qualitative, real-time or historical and can use this information to support their decision-making process.

Quantitative data, such as historical price databases or real-time price information are largely processed by automatic computer programs, often based on artificial intelligence techniques, that produce quantitative analysis, such as historical price analysis, price forecasts or technical analysis of price behaviour.

Differently, little progress has been made in the processing of qualitative data, which mainly consists of financial news articles either from financial newspapers, such as *The Financial Times* or from on-line news services, such as *Dow Jones* or *Bloomberg*. As a result, financial market players are **overloaded with qualitative information** which is potentially extremely useful but, due to the lack of time, is often ignored. In fact, the time needed for reading a financial article (e.g. the article shown in figure 1.1) is quite significant and the operator is often unable to assimilate the relevant information before an investment decision has to be taken.

The goal of this work is to reduce the qualitative data-overload of the financial operators by extracting the relevant information from a large collection of source financial documents. In particular, the work has two main aims.

- **Aim 1: implementation of templates for extracting relevant information from financial articles** The first goal of the research is to implement a set of financial templates in the LOLITA System. The system should

The Argentine government is stepping up pressure on the nation's Congress to approve key economic legislation early in the new year. Legislators failed to vote on two important measures in extraordinary sessions held this week. Debate on the measures - a presidential request for additional taxing and budget-trimming powers and a bill to advance reform of federal-provincial government relations - was again suspended late on Thursday evening after opposition members of the lower house of Congress withdrew from the chamber. Their walk-out denied the chamber a quorum and pushed consideration of the bills into January. The government is now insisting that extraordinary congressional sessions be extended as far as February to win approval of the measures it argues are critical to repairing the nation's finances in 1996. Argentina's Congress normally takes a long southern summer recess over the months of December, January and February. The measure considered to be the most important is President Carlos Menem's request for extraordinary powers to cut expenditure and raise taxes without congressional approval. The government argues that the proposed 'superpowers' are necessary to ensure that the 1996 budget is balanced and International Monetary Fund fiscal targets are met. The second major measure, the ambitiously titled Reform of the State legislation, aims to streamline financial relations between the national government in Buenos Aires and the country's 23 provinces. The federal government argues that elimination of duplication of services in the two tiers of government could save up to Dollars 1.5bn a year but is reluctant to confirm that efficiency measures may see the loss of up to 20,000 state sector jobs. The congressional backlog was cleared somewhat earlier this week when the upper house of Congress, the Senate, approved the 1996 budget law and a hotly debated measure to increase taxes on tobacco products. Argentine financial markets have maintained a firm tone in thin holiday-season trade this week, and investors appear content to give Congress some additional leeway to approve the government-sponsored measures. Economist Raul Buonuome of ING Barings said the markets were willing to accept that the Senate might not sign-off on the measures until March but had expected lower house approval this week.

Figure 1.1: An article from *The Financial Times.*

process a large number of source articles and extract a number of templates with the relevant information for the financial operator identified in the source texts and eliminate the non-relevant information.

- **Aim 2: design and implementation of a user-definable template interface.** The second aim of this research is to design and implement a user-definable template interface for allowing the end-user of the system to easily extract additional information from the source articles by designing new templates. The user-definable template interface should allow the user to *easily* define new templates using natural language sentences. The system should be able to process a large number of source articles and extract a number of templates containing the information corresponding to the definitions provided by the user.

This chapter discusses some important background methodological issues, focusing on the methodology chosen for this work: Natural Language Engineering (NLE).

1.2 Traditional Natural Language Processing Approaches

Natural Language Processing (NLP) has been influenced by many disciplines concerned with the automatic processing of natural language by computers. This section will briefly outline the main three disciplines in which researchers in this field tend to belong.

1.2.1 Cognitive Science

The aim of cognitive scientists is generally to emulate the processes of the brain into a computational model. The target of the researchers in cognitive science specialising in natural language processing is therefore to try to model the brain's communication processes. The work in this area is normally founded on psychological and sometimes physiological experiments on how humans process language, normally performed on children or on people with language disabilities. Differently from other methodologies, the goal of cognitive scientists specialising in natural language processing is therefore not only to design systems which have the same characteristics as humans do, but also to model the processes that govern this behaviours. This research does not always result in a computer program but, often, only in a computational model which could potentially be implemented. However, even if these models were implemented, they may be able to work only in very restricted areas and domains.

1.2.2 Artificial Intelligence

The aim of researchers with a background in artificial intelligence is to simulate human behaviour, without the restriction that the processes that lead to this behaviour have to be the same as those of the human brain. Researchers in artificial intelligence use therefore whatever means to produce human-like behaviour. Differently from cognitive scientists, researchers in artificial intelligence tend to cover a wider scope than the isolated processes studied by cognitive scientists. Artificial intelligence is the background approach adopted in the work presented in this thesis.

1.2.3 Computational Linguistics

The term computational linguistics was originally "*concerned with the application of a computational paradigm to the scientific study of human language*" [Ballard and Jones, 1990]. The advent of computers provided a tool for linguistics to test and then develop linguistic theories. More recently, computational linguistics has expanded to include "*engineering of systems that process or analyse written or spoken language*" [Ballard and Jones, 1990]. In practice, the terms computational linguistics is now used by a various number of researchers working on different aspects of linguistics.

1.3 Natural Language Engineering

As mentioned above, this work is concerned with the automatic extraction of information from financial articles. The methodology chosen for this work is Natural Language Engineering (NLE).

Natural Language Engineering is a recent endeavour which applies ideas and practices of other engineering disciplines to the field of NLP. The EEC LRE programme [lre, 1992] defines Linguistic Engineering as follows:

"Linguistic Engineering (LE) is an engineering endeavour, which is to combine scientific and technological knowledge in a number of relevant domains (descriptive and computational linguistics, lexicology and terminology, formal languages, computer science, software engineering techniques etc.). LE can be seen as a rather pragmatic approach to computerised language processing, given the current inadequacies of theoretical Computational Linguistics."

1.3.1 The General Philosophy of Natural Language Engineering

The goal of traditional approaches to Natural Language Processing, including cognitive science, computational linguistics and artificial intelligence, has been to formulate universal theories able to cover all aspects of language or to develop very restricted theories and prototype to be used in extremely limited domains.

The implementation of systems covering larger domains based on these ideas has been proved a difficult task.

The main idea behind Natural Language Engineering (NLE) is therefore to adopt a set of engineering criteria in the field of Natural Language Processing. Natural Language Engineering concerns with the utilisation of existing technology in order to produce useful systems. As new technologies and theories become available from more traditional sciences, they can be implemented into a natural language engineering system.

The following subsections will detail important aspects of Natural Language Engineering.

1.3.2 Scale

The size of NLE systems must be sufficient for realistic large-scale applications [Smith, 1996]. For the design of a financial application, this aspect is crucial. Financial operators, in fact, must be sure that the system is able to process any sort of input data. Skipping some data, in fact, can lead to a loss of extremely important information, which is essential for the decision-making process. The size of the system must be therefore enough to allow the processing of any sort of financial article. Properties such as the vocabulary size, grammar coverage and the number of word senses are critical.

For an information extraction system, scale also concerns the information resulting from the extraction process which is supplied to the user. This information must be sufficient in scale for representing all the relevant information in the source data. Properties such as the number of templates and slots are therefore essential.

1.3.3 Robustness

The success of a financial NLP application is directly dependent on *robustness*. The application must carry on and try its best to produce some sort of output of the input or, at least, inform the user of the failure. For a real-time use of the financial application, in fact, crashes or failures of the system must be avoided. In case of crashes, the operator would probably waste a considerable amount of time trying to find what and where went wrong. In case the system carries on without informing the user, the operator could loose information which is extremely relevant for the trading decision-making process without realizing it.

1.3.4 Maintainability

Maintainability is a measure of how useful the system is over a long period of time. From a financial point of view, the application must be usable over a sufficiently long period of time to justify the large amount of money which would be needed for the development of a complete application. The system should allow the user to change and update the configuration - i.e. adding new templates to the existing ones, or adding domain specific-knowledge (new terms, new meanings of existing ones or domain-specific rules).

1.3.5 Flexibility

Flexibility or portability is a measure of the ability to modify the system for different tasks in different domains [Smith, 1996]. For a complete financial application based on NLP, flexibility measures the ability of the system to perform tasks which are not necessarily restricted to one particular operator (e.g. broker), but is able to process different kinds of data and produce different kinds of output information according to the type of user.

1.3.6 Integration

There are two related aspects of integration :-

- System components should not make unreasonable assumptions about other components. Similarly, components should only perform tasks for which they have specifically been designed [Smith, 1996]. In the development of a financial information extraction system, the *template-extraction module* should be kept separated from the NLP core system. This separation would allow easier upgrades of the financial application by modifying the different modules of the system.

- System components should be designed and built to assist other components. For example, the financial information extraction module should allow *hyper-*

University of Durham

Financial Information Extraction using pre-defined and user-definable Templates in the LOLITA System.

Marco Costantino

Laboratory for Natural Language Engineering,
Department of Computer Science.

Submitted for the degree of

Doctor of Philosophy

Abstract

Financial operators have today access to an extremely large amount of data, both quantitative and qualitative, real-time or historical and can use this information to support their decision-making process.

Quantitative data are largely processed by automatic computer programs, often based on artificial intelligence techniques, that produce quantitative analysis, such as historical price analysis or technical analysis of price behaviour. Differently, little progress has been made in the processing of qualitative data, which mainly consists of financial news articles from financial newspapers or on-line news providers. As a result the financial market players are overloaded with qualitative information which is potentially extremely useful but, due to the lack of time, is often ignored.

The goal of this work is to reduce the qualitative data-overload of the financial operators. The research involves the identification of the information in the source financial articles which is relevant for the financial operators' investment decision making process and to implement the associated templates in the LOLITA system. The system should process a large number of source articles and extract specific templates according to the relevant information located in the source articles.

The project also involves the design and implementation in LOLITA of a user-definable template interface for allowing the users to *easily* design new templates using sentences in natural language. This allows user-defined information extraction from source texts. This differs from most of existing information extraction systems which require the developers to code the templates directly in the system.

The results of the research have shown that the system performed well in the extraction of financial templates from source articles which would allow the financial operator to reduce his qualitative data-overload. The results have also shown that the user-definable template interface is a viable approach to user-defined information extraction. A trade-off has been identified between the *ease of use* of the user-definable template interface and the *loss of performance* compared to hand-coded templates.

Acknowledgements

I would like to thank my supervisor Richard G. Morgan for his advice and support throughout the three years over which this research has been conducted. I would also like to thank Russell J. Collingham for his help and support and Stephen Eckett for his invaluable advice on the financial aspects of the thesis.

I am also grateful for all past and present members of the Laboratory for Natural Language Engineering who have helped provide such a pleasant research and social environment. Thank you to all those people who commented on the various drafts of this thesis, particularly Luisa Mich and Luigi Colazzo.

Financial assistance for this project was received from the Opera Universitaria Society of the University of Trento. I would also like to thank the department of Economics of the University of Trento for awarding me the University of Trento studentship for postgraduate studies abroad.

I would also like to thank you all my friends who supported me during the these three years and, particularly, Edy, Mika, Marianna, Paolo, Wendy, Janice, Giacomo, Max, Xavier and Sergio.

Finally I would like to thank my Mum for her support during my time at the University.

Declaration

The material contained within this thesis has not previously been submitted for a degree at the University of Durham or any other university. The research reported within this thesis has been conducted by the author unless indicated otherwise.

Contents

List of Figures

Chapter 1

Methodological Introduction

1.1 Introduction

The goal of *information extraction* is to extract specific kinds of information from a source document [Riloff and Lehnert, 1994]. In other words, the input to the system is a document (e.g. a newspaper article), while the output consists of a summary of the contents of the document. For example, the source article could consist of the following text:

BELL ATLANTIC announced it will acquire Tele-Communications Inc with 18 billion dollars. The deal will radically change the US communications industry. It will also be one of the country's biggest ever takeovers. The deal - which is bound to face strong regulatory scrutiny - would be the first full merger between a US telephone company and a cable business at a time when the two industries are converging to create a single, multi-media inter-active entertainment and information business.

An information extraction system will try to produce a template of the original text according to a specification of the information defined during the design of the system. For example, a sensible summary of the article shown above could be:

```
Template: TAKEOVER
    COMPANY_TARGET: Tele-Communications Inc.
    COMPANY_PREDATOR: BELL ATLANTIC
    TYPE_TAKEOVER: FRIENDLY
    VALUE: 18 billion dollars.
    ATTRIBUTION: BELL ATLANTIC
```

The summary presented above is called a *template* which is a structure with a predefined number of slots, where the slots are filled with the information extracted from the source article.

Players in the financial market have today access to an extremely large amount of data, both quantitative and qualitative, real-time or historical and can use this information to support their decision-making process.

Quantitative data, such as historical price databases or real-time price information are largely processed by automatic computer programs, often based on artificial intelligence techniques, that produce quantitative analysis, such as historical price analysis, price forecasts or technical analysis of price behaviour.

Differently, little progress has been made in the processing of qualitative data, which mainly consists of financial news articles either from financial newspapers, such as *The Financial Times* or from on-line news services, such as *Dow Jones* or *Bloomberg.* As a result, financial market players are **overloaded with qualitative information** which is potentially extremely useful but, due to the lack of time, is often ignored. In fact, the time needed for reading a financial article (e.g. the article shown in figure 1.1) is quite significant and the operator is often unable to assimilate the relevant information before an investment decision has to be taken.

The goal of this work is to reduce the qualitative data-overload of the financial operators by extracting the relevant information from a large collection of source financial documents. In particular, the work has two main aims.

- **Aim 1: implementation of templates for extracting relevant information from financial articles** The first goal of the research is to implement a set of financial templates in the LOLITA System. The system should

The Argentine government is stepping up pressure on the nation's Congress to approve key economic legislation early in the new year. Legislators failed to vote on two important measures in extraordinary sessions held this week. Debate on the measures - a presidential request for additional taxing and budget-trimming powers and a bill to advance reform of federal-provincial government relations - was again suspended late on Thursday evening after opposition members of the lower house of Congress withdrew from the chamber. Their walk-out denied the chamber a quorum and pushed consideration of the bills into January. The government is now insisting that extraordinary congressional sessions be extended as far as February to win approval of the measures it argues are critical to repairing the nation's finances in 1996. Argentina's Congress normally takes a long southern summer recess over the months of December, January and February. The measure considered to be the most important is President Carlos Menem's request for extraordinary powers to cut expenditure and raise taxes without congressional approval. The government argues that the proposed 'superpowers' are necessary to ensure that the 1996 budget is balanced and International Monetary Fund fiscal targets are met. The second major measure, the ambitiously titled Reform of the State legislation, aims to streamline financial relations between the national government in Buenos Aires and the country's 23 provinces. The federal government argues that elimination of duplication of services in the two tiers of government could save up to Dollars 1.5bn a year but is reluctant to confirm that efficiency measures may see the loss of up to 20,000 state sector jobs. The congressional backlog was cleared somewhat earlier this week when the upper house of Congress, the Senate, approved the 1996 budget law and a hotly debated measure to increase taxes on tobacco products. Argentine financial markets have maintained a firm tone in thin holiday-season trade this week, and investors appear content to give Congress some additional leeway to approve the government-sponsored measures. Economist Raul Buonuome of ING Barings said the markets were willing to accept that the Senate might not sign-off on the measures until March but had expected lower house approval this week.

Figure 1.1: An article from *The Financial Times.*

process a large number of source articles and extract a number of templates with the relevant information for the financial operator identified in the source texts and eliminate the non-relevant information.

- **Aim 2: design and implementation of a user-definable template interface.** The second aim of this research is to design and implement a user-definable template interface for allowing the end-user of the system to easily extract additional information from the source articles by designing new templates. The user-definable template interface should allow the user to *easily* define new templates using natural language sentences. The system should be able to process a large number of source articles and extract a number of templates containing the information corresponding to the definitions provided by the user.

This chapter discusses some important background methodological issues, focusing on the methodology chosen for this work: Natural Language Engineering (NLE).

1.2 Traditional Natural Language Processing Approaches

Natural Language Processing (NLP) has been influenced by many disciplines concerned with the automatic processing of natural language by computers. This section will briefly outline the main three disciplines in which researchers in this field tend to belong.

1.2.1 Cognitive Science

The aim of cognitive scientists is generally to emulate the processes of the brain into a computational model. The target of the researchers in cognitive science specialising in natural language processing is therefore to try to model the brain's communication processes. The work in this area is normally founded on psychological and sometimes physiological experiments on how humans process language, normally performed on children or on people with language disabilities. Differently from other methodologies, the goal of cognitive scientists specialising in natural language processing is therefore not only to design systems which have the same characteristics as humans do, but also to model the processes that govern this behaviours. This research does not always result in a computer program but, often, only in a computational model which could potentially be implemented. However, even if these models were implemented, they may be able to work only in very restricted areas and domains.

1.2.2 Artificial Intelligence

The aim of researchers with a background in artificial intelligence is to simulate human behaviour, without the restriction that the processes that lead to this behaviour have to be the same as those of the human brain. Researchers in artificial intelligence use therefore whatever means to produce human-like behaviour. Differently from cognitive scientists, researchers in artificial intelligence tend to cover a wider scope than the isolated processes studied by cognitive scientists. Artificial intelligence is the background approach adopted in the work presented in this thesis.

1.2.3 Computational Linguistics

The term computational linguistics was originally "*concerned with the application of a computational paradigm to the scientific study of human language*" [Ballard and Jones, 1990]. The advent of computers provided a tool for linguistics to test and then develop linguistic theories. More recently, computational linguistics has expanded to include "*engineering of systems that process or analyse written or spoken language*" [Ballard and Jones, 1990]. In practice, the terms computational linguistics is now used by a various number of researchers working on different aspects of linguistics.

1.3 Natural Language Engineering

As mentioned above, this work is concerned with the automatic extraction of information from financial articles. The methodology chosen for this work is Natural Language Engineering (NLE).

Natural Language Engineering is a recent endeavour which applies ideas and practices of other engineering disciplines to the field of NLP. The EEC LRE programme [lre, 1992] defines Linguistic Engineering as follows:

"Linguistic Engineering (LE) is an engineering endeavour, which is to combine scientific and technological knowledge in a number of relevant domains (descriptive and computational linguistics, lexicology and terminology, formal languages, computer science, software engineering techniques etc.). LE can be seen as a rather pragmatic approach to computerised language processing, given the current inadequacies of theoretical Computational Linguistics."

1.3.1 The General Philosophy of Natural Language Engineering

The goal of traditional approaches to Natural Language Processing, including cognitive science, computational linguistics and artificial intelligence, has been to formulate universal theories able to cover all aspects of language or to develop very restricted theories and prototype to be used in extremely limited domains.

The implementation of systems covering larger domains based on these ideas has been proved a difficult task.

The main idea behind Natural Language Engineering (NLE) is therefore to adopt a set of engineering criteria in the field of Natural Language Processing. Natural Language Engineering concerns with the utilisation of existing technology in order to produce useful systems. As new technologies and theories become available from more traditional sciences, they can be implemented into a natural language engineering system.

The following subsections will detail important aspects of Natural Language Engineering.

1.3.2 Scale

The size of NLE systems must be sufficient for realistic large-scale applications [Smith, 1996]. For the design of a financial application, this aspect is crucial. Financial operators, in fact, must be sure that the system is able to process any sort of input data. Skipping some data, in fact, can lead to a loss of extremely important information, which is essential for the decision-making process. The size of the system must be therefore enough to allow the processing of any sort of financial article. Properties such as the vocabulary size, grammar coverage and the number of word senses are critical.

For an information extraction system, scale also concerns the information resulting from the extraction process which is supplied to the user. This information must be sufficient in scale for representing all the relevant information in the source data. Properties such as the number of templates and slots are therefore essential.

1.3.3 Robustness

The success of a financial NLP application is directly dependent on *robustness*. The application must carry on and try its best to produce some sort of output of the input or, at least, inform the user of the failure. For a real-time use of the financial application, in fact, crashes or failures of the system must be avoided. In case of crashes, the operator would probably waste a considerable amount of time trying to find what and where went wrong. In case the system carries on without informing the user, the operator could loose information which is extremely relevant for the trading decision-making process without realizing it.

1.3.4 Maintainability

Maintainability is a measure of how useful the system is over a long period of time. From a financial point of view, the application must be usable over a sufficiently long period of time to justify the large amount of money which would be needed for the development of a complete application. The system should allow the user to change and update the configuration - i.e. adding new templates to the existing ones, or adding domain specific-knowledge (new terms, new meanings of existing ones or domain-specific rules).

1.3.5 Flexibility

Flexibility or portability is a measure of the ability to modify the system for different tasks in different domains [Smith, 1996]. For a complete financial application based on NLP, flexibility measures the ability of the system to perform tasks which are not necessarily restricted to one particular operator (e.g. broker), but is able to process different kinds of data and produce different kinds of output information according to the type of user.

1.3.6 Integration

There are two related aspects of integration :-

- System components should not make unreasonable assumptions about other components. Similarly, components should only perform tasks for which they have specifically been designed [Smith, 1996]. In the development of a financial information extraction system, the *template-extraction module* should be kept separated from the NLP core system. This separation would allow easier upgrades of the financial application by modifying the different modules of the system.

- System components should be designed and built to assist other components. For example, the financial information extraction module should allow *hyper-*

links to other sources of information even if, initially, no information is linked.

1.3.7 Feasibility

This aspect concerns ensuring that constraints on the running of the system are acceptable [Smith, 1996]. Parameters such as hardware requirements should be taken into account. For real applications (e.g. a financial application), this aspect is crucial: the application must be able to run on an *existing* and *affordable* machine. Feasibility comprises also efficiency. A financial application must be *efficient* in the sense that the execution-time must be acceptable for the operator's need. In the development of a financial information extraction system this aspect is crucial: the application must be able to run on an *existing* machine at a acceptable speed to be useful to the financial operator.

1.3.8 Usability

The system must satisfy a need - i.e. there must be a set of users in the market who can benefit from using the system [Garigliano, 1995]. For a real financial application this aspect is largely the most important. Most of the existing information extraction tools that have been developed and tested within government agencies and scientific environments, in fact, lack of usability, in the sense that they would not be particularly useful for the financial operator. For example, the tasks of the MUC-5 [DAR, 1993] and MUC-6 [DAR, 1995] competitions would be not immediately applicable in a financial application.

The design of a financial application must therefore take into consideration first of all the reaction of the potential customers and how the innovation will be accepted. The technology behind NLP systems, in fact, is today not yet completely understood within the financial community and operators are not prepared to deal with it. Even a simple concept as "template" can be obscure. Therefore, the key-point must be to design a product which can first of all be understood by the potential customers and to find a relevant market sector that needs it. The danger would otherwise be to produce an extremely sophisticated tool which, however, does not satisfy a user need. As a obvious consideration, the systems should be user-friendly.

1.3.9 The use of a full range of techniques

NL engineered systems should use a full range of AI techniques [Smith, 1996]. The designers of the financial application should therefore use any technique available which can improve the system's performance.

1.3.10 Cost-Benefit Analysis

The designer of a financial tool based on Natural Language Processing should reach a balance between two or more aspects of Natural Language Engineering. If, for example, a simple algorithm might not lead to the same *robustness* of a more complicated one but will improve the *feasibility* of the application (for example in terms of speed), then the first one can be chosen.

1.3.11 Motivation for Adopting the NLE Approach in finance

The Natural Language Engineering approach has been studied for the design of large NLP systems [Garigliano, 1995]. In our view, however, the NLE principles can be used for a successful design of a real financial application based on natural language processing. Principles such as *usability*, *feasibility*, *scale* and *robustness* are essential not only for a NLP system, but for any financial application which must be designed to work in real situations and can be successfully used as *general support*, *explanatory* or *forecasting* financial tool. The design of the LOLITA financial information extraction system follows these principles which have been employed in the design of the system as well as in its implementation.

1.4 Methodological criteria for success

The goal of this work is to provide evidence for or against the following tasks:

- implement a set of templates which represent relevant information for the financial operator in the LOLITA System.
- design and implement an user-definable template interface for allowing the end-user of the system to *easily* define and extract additional information from the source articles in the form of templates.

Three main criteria will be employed to evaluate the success of the work described in this thesis:

- the *performance* of the system in the extraction of the pre-defined and user-defined templates will be evaluated using the MUC-6 standard evaluation measures [Chinchor and Dungca, 1995] *precision*, *recall* and *'F' measure.* In particular, the performance of the templates defined using the user-definable template interface will be compared with the performance of the pre-defined templates. The performance of the system in extracting pre-defined templates from a relevant number of financial articles should be reasonably high to be useful for the financial operator, while the drop in performance of user-defined templates compared to pre-defined templates should be low.
- the *ease of use* of the user-definable template interface will be evaluated using specific measures giving an indication of the difficulties encountered by potential users in defining new templates and the time taken for entering new templates definitions. The potential users should be able to enter new templates definitions in a very limited time and with a very limited amount of errors in the templates definitions.
- the identification of the relationship between the *ease of use* of the user-definable templates and the possible *loss of performance* compared to hand-coded templates definitions.

1.5 Context of this Work: the LOLITA project

The work will be carried out as part of the LOLITA (Large-Scale Object-Oriented Interactor, Translator and Analyser) System. LOLITA is a large-scale Natural Language Processing System which has been under development at the University of Durham for the past 9 years. The LOLITA System will be described in detail in chapter 4.

1.6 Logical Progression of the Thesis

This thesis is organised according to the following plan:

Chapter 1: methodological introduction. (this chapter), provides important methodological information about the work presented in the thesis. This chapter describes in detail the problem and aims of the thesis. It also explains in detail the Natural Language Engineering methodology adopted and provides the criteria for success for these methods;

Chapter 2: finance and financial tools. This chapter analyses the most important financial tools and techniques currently used in finance, based on conventional techniques or artificial intelligence;

Chapter 3: information retrieval and information extraction. This chapter discusses the most important techniques, systems and evaluation techniques in the field of information retrieval and information extraction in relation to this work. In particular, the chapter focuses on the TREC (information retrieval) and MUC (information extraction) competitions;

Chapter 4: the LOLITA System. This chapter analyses in more detail the LOLITA System, outlining the main techniques, modules and applications available in the system;

Chapter 5: information extraction in the LOLITA System. This chapter analyses in detail the way in which the information extraction tasks are currently handled in the LOLITA System;

Chapter 6: definition of the financial templates. This chapter presents the solution for the problem aim 1. The relevant financial information in the source financial articles are identified and represented in specific financial templates;

Chapter 7: implementation of the financial templates in the LOLITA System. This chapter presents the implementation issues of the pre-defined templates defined in chapter 6;

Chapter 8 design of the user-definable template interface. This chapter presents the solution for the problem aim 2 and presents the design of the user-definable template interface;

Chapter 9: implementation of the user-definable template interface. This chapter presents the implementation of the user-definable interface described in chapter 8;

Chapter 10: results and evaluation. This chapter reports the results of the evaluation of this work referring to the project aims 1 and 2;

Chapter 11: conclusions. This chapter presents the final conclusions for the project aims and discusses the project forthcomings and directions for future research.

Chapter 2

Finance and Financial Tools

2.1 Introduction

Finance is an extremely broad discipline, including many different areas: personal finance, public finance, financial economics, corporate finance, trading, international finance etc. Finance is present at any level of our society. Drawing its exact borders is therefore extremely difficult.

In the context of this work, we will only concentrate on one extremely important area of finance, the financial market, which consists of the *capital market* and the *money market.*

The *Capital Market* is the market for financial assets other than money or near-monies [Polakoff and Durkin, 1981]. Such a market is often called an "exchange", for the essential function of facilitating the purchase and sale or exchange of long-term claims or securities. Since exchange presupposes some specialization and standardization, the capital market, in fact, consists of many separate, but interrelated, markets of varying degrees of organization, such as the Stock Exchange and the over-the-counter market (which is not, in fact, an organized exchange). It is customary to consider the *Money Market* quite separate from the *Capital Market.* The *Money Market* could be named the "near-money market", since is the market for near-money financial assets or claims. It is also usual to draw the line between

Capital and *Money* markets at a maturity of one year for the securities [Polakoff and Durkin, 1981][1]. The bulk of trading in the capital market is in outstanding securities, the so-called *secondary market.* The *secondary market* is the market for securities that have been previously issued, and the securities traded in the secondary market have been already sold once. The *primary market,* instead, consists of new issues of securities. The *secondary market* is much more important than the *primary market* for two reasons: firstly because ongoing markets are indispensable for the effective marketing of new issues; secondly, trading in outstanding securities is important in its own right because the ease with which an asset can be sold facilitates wealth-holders' constant efforts to optimize the satisfactions derived from their portfolio [Kaufman, 1989] and, therefore, supply and demand can be equated.

Although the *financial market* comprises numerous different markets, we will only focus on the *secondary market* and, more precisely, on securities issued by companies: equities, or share and company bonds. We are therefore not interested in securities issued by governments (bonds) or the currency market. This choice has been made because most of financial tools, as we will explain later in this chapter, are aimed at these kinds of securities.

Equities are, in some respects, the most important kind of securities that are dealt in the stock exchange, mainly because they form the basis of the capitalist society. The owners of ordinary shares are entitled to receive dividends when they are declared, and to be paid a proportion of the company's assets after payment of the creditors if the company is wound up [Dine, 1994].

The price movements of the shares quoted at the Stock exchanges can be explained in different ways, according to observation time period: long-term market moves are affected by fundamental factors like economics; in the medium term technical supply and demand factors can be decisive; while investor psychology can be the most powerful short-term influence. In all cases, the share price reflects the *perception* that the market has of the company. The price of equities will

[1] The trading of shares is therefore performed in the capital market, since shares do not have a fixed expiry date.

therefore be affected by the firm's own fortunes, but also by numerous other kind of information, such as the forecasts for an increase or a decrease in the interest rates and, in general, by all sorts of relevant economic news. Therefore, the important component is not the news itself, but *how* the news and data are perceived by the market.

The decision-making process that leads to investment decisions is the crucial aspect for the investors who need to support their decision with the largest possible amount of relevant information. This data can be either *quantitative*, such as historical price data, prices of raw materials, data on inflation, currency exchanges, or *qualitative*, for example a news report stating that "there are fears of an increase in the German interest rates". Another distinction is between *real-time* information and *historical information*, with the former extremely important for real-time decision-making processes.

It is important to point out that we here consider a situation in which the market is not perfect. In case of a perfect market situation[2] the operators would have free and complete access to the information needed which would allow them to perform correct decisions.

Investors have nowadays access to a huge and vast amount of information, provided by agency press news providers, historical archives, government agencies and private organizations which collect, organize, process and distribute different types of data. Investors are usually connected in real-time to the news providers, drastically reducing the lag between when news happens and when it is assimilated by the operator. The cost of this information is decreasing rapidly, making it more accessible to small or private investors.

This "information revolution", as in many other "information-based" sectors started in the late 70s, with the development of the first commercial fast computer

[2] In a perfect market operators have a perfect and costless knowledge of the market and are able to perform optimum decisions. In reality, this situation does not exist, since operators do not have access to all possible information. Moreover, the information is not freely available. The operators therefore base their decisions on imperfect knowledge which can lead to partial or incorrect solutions.

networks and continued in the 80s. The first target was to provide as much information as possible in real-time. Quantity was the target at that time. The quantity of the services and information available has been growing exponentially, leading to the current paradoxical situation, where financial operators suffer of "data overload". Operators have access to information, some of which irrelevant, which they cannot profitable use because of the time they have to spend in selecting and understanding the appropriate data for their needs. Operators try to select the most important information, leaving for further and deeper analysis the less important, for example analysis of price behaviour. In the 90s, therefore, the emphasis has shifted from the *quantity* of the information provided to the *quality.*

Quality can be improved by providing analysis and automatic selection of this large amount of information. The financial operator, by using this new quality information, can drastically reduce the time he needs to select the appropriate data and "summarize" it.

Financial tools are instruments used to perform such tasks and to produce new *inferred* data. We can define a "financial tool" as a tool that can be used to support the decision-making process of the investors, by summarizing, selecting or producing some analysis of the original source data or suggest some operative decisions. Financial tools are therefore instruments used to perform *economical analysis* of the stock market.

Many different kinds of financial tools have been developed. In the context of this work, we only concentrate on tools used in decision-making processes dealing with the trading of securities issued by companies and quoted in the stock exchange. We will not therefore consider tools regarding forecast analysis of other kinds of prices, profit predictions, economic forecasts etc.

Financial tools can be classified according to three different criteria:

- the kind of technique on which the tool is based;
- the kind of input/output of the system;
- the kind of task that the tool performs for the financial operator.

Depending on the *kind of technique on which it is based*, financial tools can be grouped into *conventional tools* and *innovative tools* [Costantino *et al.*, 1996c]. Conventional tools are based on mathematical, statistical and probabilistic algorithms. These techniques are extremely well-known, highly optimized and effective on particular sets of data and have been experimented for a long time. Conventional tools can be further classified in: *systems based on statistics*, *systems based on econometrics* and *systems based on heuristics* [Fornasini and Bertotti, 1989].

Systems based on statistics perform analysis and calculations on data consisting of historical time series of share prices. They can be either *descriptive* if they provide a simple and effective representation of the phenomena observed or *interpretative*, if they try to identify the variables able to explain the phenomena or the link between the explicative variables and the phenomena analysed.

Systems based on heuristics try to explain the economical and financial events without an underlying economical / econometrical or mathematical relation, but basing on "heuristics" and knowledge available within the financial community.

Systems based on econometrics try to identify the mathematical / economical relations and models which are able to interpretate and explain the evolution of the economical phenomena [Fornasini and Bertotti, 1989]. This kind of analysis is called *fundamentalistic analysis.*

Innovative tools (or tools based on *AI techniques*) have been introduced in the last 20 years and are mainly based on techniques borrowed from Artificial Intelligence, such as *Neural Networks* and *Expert systems.*

According to the *input* and *output* of the system financial tools can be classified as *quantitative* or *qualitative* tools [Costantino *et al.*, 1996c]. Quantitative tools are tools used to produced quantitative or qualitative information from quantitative data. Differently, qualitative tools are tools which process qualitative data and produce qualitative or quantitative information.

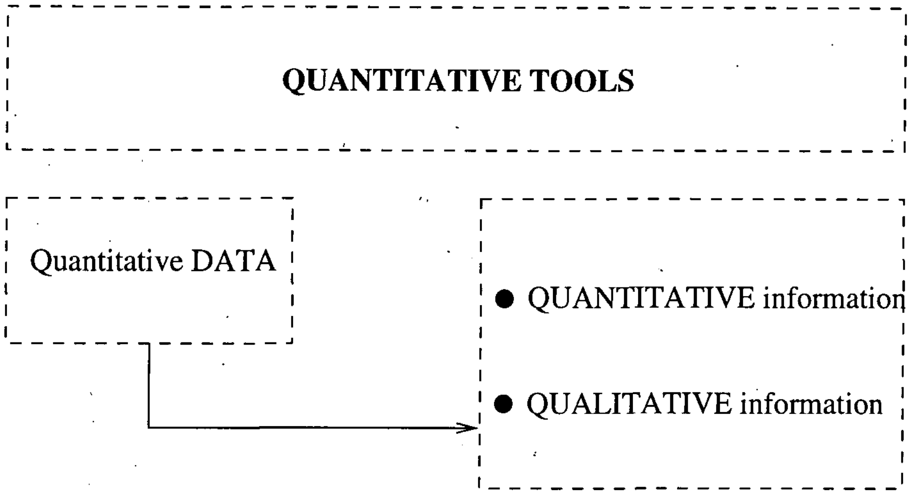

Figure 2.1: The kind of input and output of a quantitative tool

Finally, we can distinguish between **explanatory tools**, (e.g. historical price analysis), **forecasting tools**, (e.g. linear prediction, falls in the category of the tools based on conventional techniques, and neural networks for prediction of prices, which falls into the AI category) and **general support tools**. *Explanatory tools* can be defined as financial tools which are used for explaining the movements of shares. However, they cannot normally be used for the prediction of future prices. *Forecasting tools* are, on the contrary, tools used to predict future prices of the stock. Finally, *general support tools* are tools which are used as support for the analysis of the financial operator or to pre-process the information, but cannot be used for explaining movements of share prices or for predictions.

2.2 Quantitative tools

We can define *quantitative tools* as the tools that are used to produce quantitative or qualitative information from quantitative data (figure 2.1). For example, a quantitative tool can process the price-history of a particular share, taking into account any selected variable that is relevant and shows the main periodical cycles and trends. Other tools can be used to predict the future prices of the same shares or to suggest to the financial operator when is the best time to sell/buy a particular share.

	Explanatory	Forecasting
Conventional	Moving average MACD / divergence indicator Stochastic Index Trading Bands	Linear and n.l. Predictions Applied Logic Regression MA - MACD Trading Bands
Innovative	Expert Systems	Expert Systems Neural Networks

Figure 2.2: Different kinds of quantitative tools.

Quantitative tools have been historically based on conventional techniques. More recently, tools based on *innovative* techniques have been introduced. For example *Neural Networks* are mostly used in finance as quantitative tools (prediction of shares). *Expert Systems*, instead, have been instead applied for both quantitative and qualitative problems. Figure 2.2 shows tools based on different techniques.

One of the drawbacks of quantitative tools is that the information produced consists of "numbers" and has to be interpreted by the financial operators thus slowing-down their investment decision-making process. Even most of the tools based on the latest Artificial Intelligence techniques present similar problems. For example, the input and output of *neural networks* is strictly numeric [Walczak, 1995]. The output of such systems is often difficult or impossible to understand for humans. Therefore, before a tool can be used, the real world data must be translated into a numeric quantitative form and re-translated after the elaboration into a qualitative form. Such tasks are extremely time-consuming and, in most cases, cannot be automated.

2.2.1 Statistical and probabilistic techniques

In this section we will discuss some of the main conventional tools based on mathematics, statistics and probabilistic techniques, adapted and enhanced for the use in finance. The mathematical disciplines have been studied for hundreds of centuries and therefore some techniques and algorithms are extremely well known and easily applicable in the financial domain. Mathematics and more specificly statistics are extremely wide-spread in the financial domain and nowadays, they are still the most used tools within the financial community.

One of the main disadvantages of most of the indexes and techniques is that they do not take into account any "intelligent" mechanism for analyzing data - they use only statistics and mathematics. Therefore, they fail when the facts are unpredictable or "illogical". Differently, financial tools based on artificial intelligence have been designed following the assumption that prices in the stock exchange are not originated by mathematical or statistical equations. They are determined by how the people involved in the buying and selling process think in a particular moment and how they are influenced by facts. Therefore, trying to explain people's behaviour, which reflects into trading decisions using statistical and mathematical indexes can, often, lead to wrong analysis.

However, some of the quantitative indexes based on conventional techniques, such as the moving average or the average of the quotation of a share in a certain amount of time are extremely important for the operators' decision-making process and they are still among the most frequently used techniques in the financial community.

MA, MACD, stochastic index and trading bands

In this section we will briefly cover some of the most frequently used indexes in the financial community: the *moving average*, the *moving average convergence/divergence (MACD) indicator*, the *stochastic index* and the *trading bands.* These indexes are used to identify *trend reversals* - changes in the current global direction (upward or downward) of the price of a share. Knowing the exact time in which the trend changes is extremely important to maximize the value of the investment since it is usually the time in which an operator decides whether to buy or sell a particular share. We can therefore consider them to belong to the group of *forecasting* indexes. However, the basic moving average index is also often used as support for other techniques and, therefore, it might also be included in the category of *explanatory* tools.

Trends are typically generated by institutional investors, which represent about 70 per cent of the total trading of a stock [Brown and Bentley, 1995]. Theoretically, once a trend-direction is in place, it is not easily changeable in the opposite direction. The trend therefore continues in the same direction until there is either an *over-bought* situation - where buyers dry up, creating a downward correction for the trend, or an *over-sold* situation - for a downward trend, in which sellers dry up and an upward trend is taken.

Moving average and *MACD* are two techniques that help identify the point in which the trend is likely to change direction.

Moving average (MA) is one of the basic statistical indexes and is often the basis of other indexes. Basically, it is used to smooth out the fluctuations in the stock prices (over a significance number of observances), obtaining the underlying trend. The result is normally plotted on a graph, with intervals usually of 30 days. The way in which the MA is computed is very straightforward. Basically, to obtain a simple 30-day moving average, the total closing price of a stock over of 30 days is computed and divided by 30. Once the main average has been obtained, to compute the subsequent points is only necessary to add the most recent closing price to the total closing price of the stock, removing the oldest and divide it by 30.

From the operator's point of view, when a stock breaks consistently through its 30-day moving average (either upwards or downwards) the probability of a change in the trend is more likely and the operator must decide whether to buy or sell a particular share.

One of the main problems of the basic moving average is the fact that the indication can be misleading suggesting changes in trend which are only temporary or totally random. An improvement of the basic MA index is the *moving average convergence/divergence indicator (MACD).*

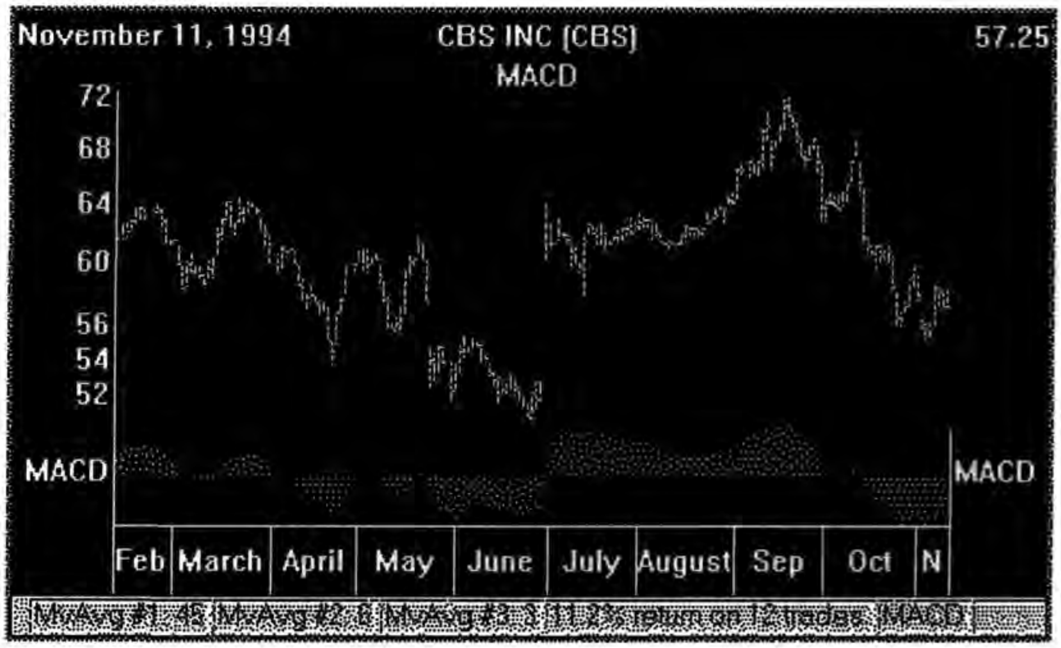

Figure 2.3: A MACD indicator histogram.

MACD

The target of the Moving Average Convergence / Divergence (MACD) indicator is to "measure the intensity and direction of the market's mood and confirm a trend reversal." [Brown and Bentley, 1995].

The MACD indicator uses three exponential moving average: a short or fast average, a long or slow average, and an exponential average of the difference between the short and long moving averages, which is used as a *signal line*. An exponential moving averages is similar to a normal moving average but it gives more weight to the most recent observations.

The MACD indicator is usually plotted on a graph. In figure 2.3, the example of a MACD histogram is shown. The blue bars represent the difference between the short (fast) and long (slow) exponential moving averages. The outer edge of the bars forms the *MACD line*. The red line (or *signal line*) that traverses the blue area is the exponential moving average of the difference between the short and long averages. It measures the convergence or divergence of the short and long moving averages.

Changes in directions of the trend are signaled by the convergence and divergence of these indexes. When the indexes converge going downwards, a "negative breakthrough" is signaled, and the operator should sell his stocks, while a "positive breakthrough" occurs when the indexes converge by an upwards direction.

More in detail, the MACD graph can be interpreted as follows:

- A **buy signal** is suggested when the market is in an over-sold situation and, thus, the MACD line crosses above the signal line. This event is called a *positive breakthrough.*
- A **sell signal** is suggested when the market is in an over-bought condition and, thus, the MACD line crosses the signal line downwards. This event is called a *negative breakthrough.*

In the lower part of figure 2.3 the MACD histogram is shown. The histogram is another way of representing the MACD indicator and buy/sell signals are given when the histogram crosses the zero line.

Stochastic index

The *stochastic index* is not as widespread as the *moving average* or the *MACD* and can be defined as the percentage of the difference between the low and high quotations of a stock. Similarly to the other two indexes, it is mainly used for identifying situations in which a particular stock is *over-sold* or *over-bought* and, therefore, the trend will be soon reversed.

Trading bands

Trading bands are often used to support indications given by the MACD index and are also considered to be a "forecasting" tool. It is in fact used mainly to support the indications coming from the MACD index. The technique is based on the assumption that a stock will flotate within a predictable range around its moving average. Two additional bands are thus drawn on the graph perfectly symmetrically and at a fixed and equivalent distance, in opposite directions, from the moving average. When the price of a stock breaks through the upper or lower band then, probably, there will be a change in the current trend direction.

Many other tools and technical indicators which are not described here can be used for making decisions regarding trends or in support of them - i.e. *trend-setters*, *resistance* and *support* lines, *basing period break-out*, etc.

Linear and non-linear prediction

Linear prediction, as the name suggests, is a tool which is used to predict future prices of a stock given a history of prices over a certain amount of time (time-serie). The term "linear" emphasize the fact that the equations used are linear [Hatamian, 1995]. The next price is calculated using a linear equation containing a certain number of variables (often called *memory*) corresponding to the most recent observations, thus:

$$tr_{(}n + 1) = \Sigma_{i=N-1}^{N} b_i * y_i.$$

The historical price data (time-serie) is used in the training of the system to obtain the best-matching equation which identifies (*fit*) the weights of the coefficients applied to the specific variables. The coefficients are determined using the *least-squares* method, thus finding the minimum of all possible sums (according to the different parameters) of the squares of the difference between the real observations and the predicted ones.

The chosen model will be the best-matched explanatory model of the past data and will be used to produce forecasts.

The accuracy of the predictions drastically decreases moving from the next-day prediction to the following ones. At a certain stage, determined by the number of variables taken into consideration in the process (memory), the new predicted observations will only depend on the observations already forecasted and the possibility of error will be extremely high.

Linear-predictions are therefore mainly used to predict "next-day" price movements, rather than long term predictions for which, different kind of tools can be used. Linear prediction is also often used to predict values which are not directly related to stocks - i.e. commodity prices, inflation etc.

Differently from *linear-prediction*, *non-linear prediction* is based on non-linear equations which can include more variables. The equation of the model is:

$$tr_(n + 1) = \Sigma_{i=N-n}^{N} b_i * y_i + \Sigma_{i,j+N-n}^{N} c_{(ij)} * y_{(i)} * y_{(j)}$$

The introduction of a new set of parameters will lead to an increase in the number of variables to be fitted to match the historical data (time serie) and, consequently, in the time and complexity of the calculations. The predictions produced by this model are much better than the simple linear model since it is able to capture some complex and more irregular components of the trend. However, the intrinsic statistical nature of the technique makes it impossible to predict a totally unexpected change in the behaviour of a particular stock. In fact, as we already pointed out earlier, the behaviour of the price of a particular stock is influenced by the decisions made by the operators who do not necessarily follow a particular statistical or mathematical reasoning for performing their decision-making process.

2.2.2 Artificial Intelligence techniques

The decision-making environment for businesses is becoming more and more complex on a daily basis. This is partly caused by the exponential increase of the global marketplace, which leads to a even greater increase of information to be processed by the financial operators [Walczak, 1995]. Financial companies like *Fidelity*, *Bloomberg*, *Dow Jones* and *Dreyfus* are offering all kinds of data and services, such as stock quotes (historical and real-time), electronic trading, research reports and analysis of companies, prices etc. These data are available on cheap and efficient networks such as, for example, the internet.

Financial operators must therefore process a larger amount of information in a shorter period of time. Decision making in this complex and global environment requires therefore a high-quality and appropriate knowledge.

As we have already seen in the previous sections, statistical and probabilistic techniques are relatively simple and efficient methods for analyzing and forecasting quantitative data. However, these techniques are unable to work in a highly complex and "irregular" market where decisions are taken not on the basis of previously observed patterns or cycles, but are mainly guided by emotive and psychological factors.

Towards the end of the 70s, the first financial tools based on new techniques borrowed from artificial intelligence were introduced. These techniques offered the possibility of explaining and predicting data which did not necessarily follow a certain pattern or cycle. They were able to infer knowledge from a collection of input data. Nowadays, most of actual AI financial tools are based on *neural networks*, which are *example-based* tools and *expert systems* which, instead are *rule-based* tools [Walczak, 1995].

Over the past 15 years, the financial community and financial institutions have become the primary driver of artificial intelligence and its longest-term user [Newquist, 1995]. This situation was not what the researchers expected when the first artificial intelligence techniques were studied. When the first companies specializing in AI systems appeared on the market, they were directly related to university labs and they were working for and within that environment. The second generation of AI companies (around 1985) understood that one of the biggest potential sectors was constituted by the financial community and they started studying specific applications and techniques.

Nowadays, the financial industry is the primary commercial adapter of artificial intelligence techniques worldwide. The financial companies are the first to try the use in real situations of new AI techniques and the quickest to substitute conventional techniques to the new ones. Several reasons can be thought but the two most obvious are: access to cash and direct and immediate impact of the new technologies on the results [Newquist, 1995]. Financial institutions, in fact, are extremely keen on investing and testing new AI techniques that can potentially result in an exponential and immediate increase in the profits. Using AI techniques, financial operators can quickly get better information from data which they already own and improvements can be quickly measured in terms of monetary returns. Given the fact that a financial tool will normally be used to support the decision-making process for buying/selling decisions, the performance of these decisions can be measured within a short period of time, if not in real-time. In comparison, an industrial company is not able to obtain such a clear overview of the impact of the new technology. This is because the introduction of the new technique and the results are not directly related but have to be related to the impact in the production, quality-control or diagnoses, faster customer services, predictive maintenance and, in general, the "stage" at which the new technology is introduced. The financial community is therefore one of the biggest real-user of Artificial Intelligence techniques, compared to other sectors, which are likely to make use of more known and conventional technologies.

The first artificial intelligence technique to be introduced in the financial market has been *expert systems*. Later, systems based on *neural networks* started to be used and nowadays new systems are introduced continuously, while others begin to be analyzed and tested, such as natural language processing, the subject of this thesis.

Expert Systems

An expert system can be defined as a *"knowledge-based system that emulates expert thought to solve significant problems in a particular domain of expertise"* [Sell, 1986].

The main characteristics of expert systems compared to *neural networks* is that they are *rule-based*. This means that the expert system contains a predefined set of knowledge which is used for all decisions. The system uses the predefined

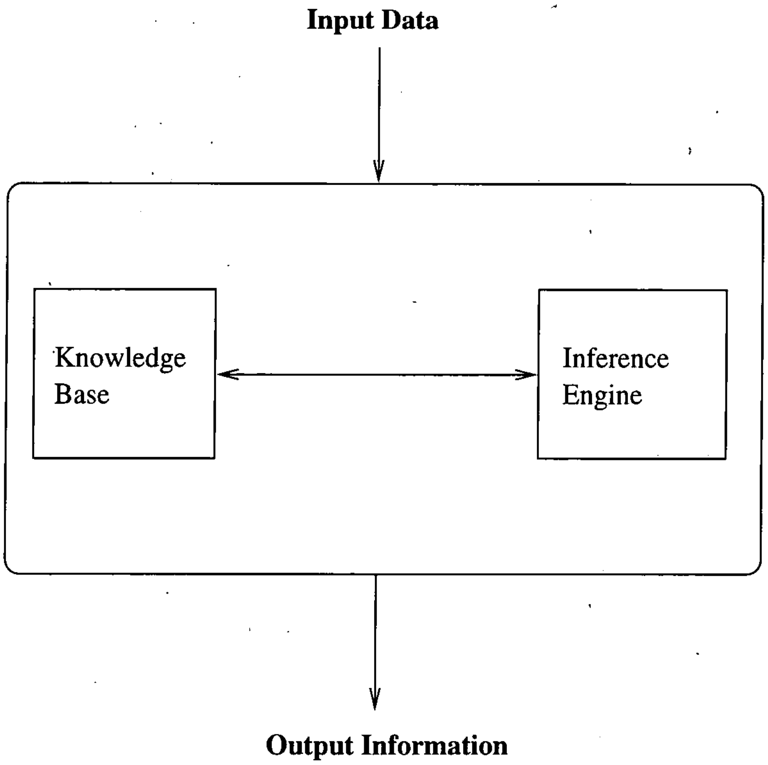

Figure 2.4: The main components of an expert system.

rules to produce results by using *inference rules* which are coded into the system. Depending on the kind of input and the rules used, expert systems can either be used as quantitative or qualitative tool.

A generic expert system will consists of two main modules: the *knowledge base* and the *inference engine.*

The knowledge base contains knowledge of the system regarding the specific domain or area for which it is designed to solve problems or make recommendations. For example, if the system has to work in the financial domain (*domain knowledge*), the *knowledge base* will include the specific rules that the system contains - i.e. decisions concerning shares. The knowledge base is coded into the system according to a specific notation which is usually found in the following forms:

- Rules
- Predicates
- Semantic nets

- Frames
- Objects

The *inference engine* processes and combines the facts related to the particular problem, case and question, using the part of the knowledge-base which is relevant. The selection of the appropriate data in the knowledge base is performed according to searching criteria. The way in which inference rules are written and applied to the information in the knowledge base vary greatly from system to system and can follow different paths. Among the others, two methods of reasoning are often employed *forward chaining* and *backward chaining* [Luconi *et al.*, 1994].

Several other modules are usually present in an expert system (e.g. meta-knowledge)[3].

The most important step for the development of financial tools based on expert systems is the acquisition of the domain specific knowledge, consisting of the methods that would be used by a domain expert for making appropriate decisions [Walczak, 1995]. This knowledge will normally consist of heuristics which, unfortunately, are extremely difficult to verbalize and the interview process to identify and collect these heuristics can last for weeks.

We will not describe in further detail the architecture of expert systems[4], but we will here concentrate on their applications in finance. Unfortunately, information about such systems is generally limited, since disclosure of successful approaches by the financial operators could lead to the loss of competitive advantage and large sums of money. As a general point, financial operators today tend to prefer neural-networks for real-time forecasting, while expert systems now tend to be used more in other financial fields, where the outcome of the system must be a clear decision - i.e. validating user's credit-card accesses. Expert systems are used in accounting [O'Keefe *et al.*, 1993], auditing [McCarthy *et al.*, 1992], decisions in insurance companies [Wright and Rowe, 1993], etc. In general, present expert

[3] For additional details on expert systems in general see [Walczak, 1995]

[4] Additional information can be found in [Zahedi, 1993], [Watkins and Eliot, 1993] or [Sell, 1986]

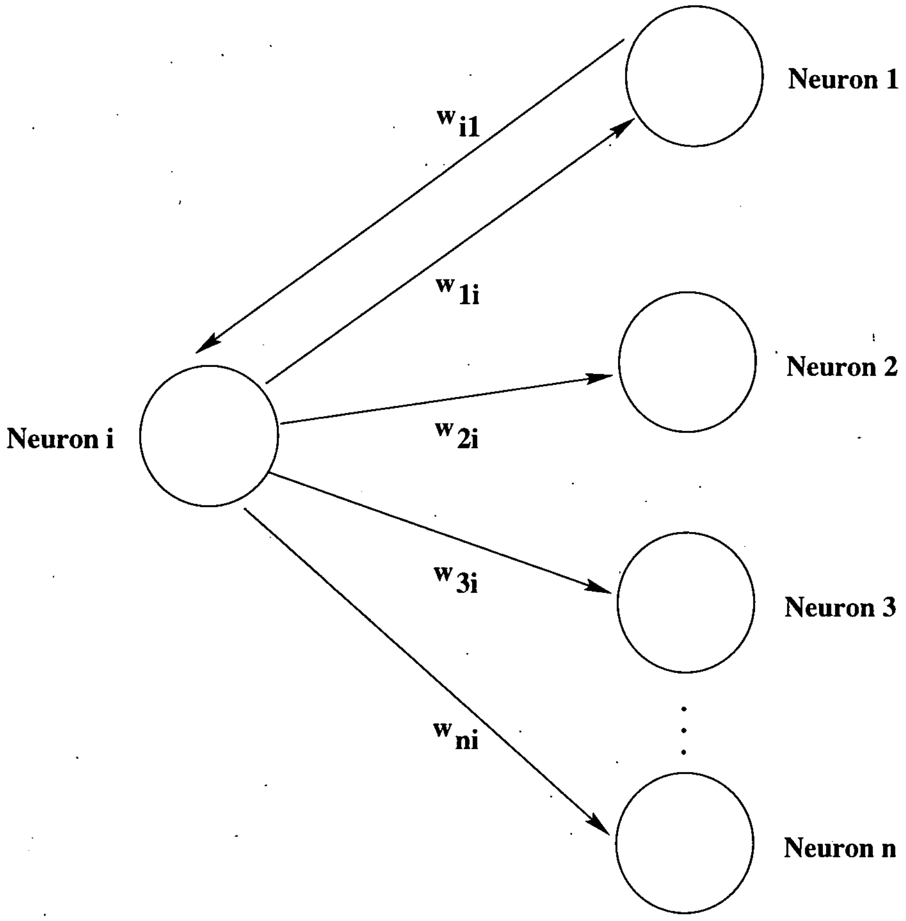

Figure 2.5: The basic structure of a neural network.

systems in finance are normally used to support the operator's decision, rather than as decision makers [Gilbert, 1995].

Neural Networks

Today neural networks are the most common system for the prediction of share prices excluding conventional techniques. Although few expert systems are still in use, neural networks have proven to be effective tools for supporting financial operator's trading decision, once they have been correctly trained. Neural networks are therefore *forecasting* tools, rather than *explanatory* tools. Neural Networks in finance are mostly used to produce quantitative information (e.g. expected prices of shares). However, they can be potentially used for qualitative analysis.

Neural networks have been created with the purpose of duplicating and simulating the components and functionalities of the brain.

The main component of a neural network is an element modeled after a neuron and called *neural*, *processing element* or just *element.* Neurons are connected to one another with a network of paths which carry the output of a neuron to the next one (figure 2.5). The fact that the communication on a single path is unidirectional does not prevent the creation of a two-way connection, since it is possible to create an additional link from the destination to the source neuron. Each neuron, although connected to many other neurons, produces a single output impulse each time, which is sent to other neurons (figure 2.5).

The most important parameter of the neural network is the *strength* or *weight* with which the neurons are linked. If two neurons, i and j are connected with two uni-directional links, the links will both have two separated weights w_{ij} and w_{ji}. The weights are extremely important, since they are the elements where the *domain-knowledge* is stored and represent the way in which the neural-network is able to learn new knowledge (through the *training* of the network). Associated to each neuron is also a *state* which is usually implemented as an extra weight and must therefore also be estimated [Refenes, 1996].

The neurons are connected using two different types of connection [Zahedi, 1993]:

- **excitatory**: in which the output of a neuron increases the potential action of the neuron connected;
- **inhibitory**: in which the connected node will reduce its activity by receiving the output of the source neuron.

In most neural networks, however, all connections are *excitatory.* The neurons are then grouped into *layers* or *slabs* which are inter connected. Each *layer* performs a specific operation - i.e. a *input layer* will consist of neurons which receive input from the external environment.

We will not further discuss the analytic representation and structure of a neural network, since it is not relevant in the context of this work[5]. The important aspect to notice here is that the neural-network, to be of any use, must *learn* from the external world and this process is called *training*. The knowledge, as we already observed, will be stored as *weights* given to the links between the neurons. The learning method is the most relevant distinguishing factor among neural networks and, in case of financial neural-networks, represents the most important aspect. The learning method can be *supervised*, if the output of the net is compared with results which are already known during the training of the net, or *unsupervised*. More important, it can be *off-line* or *on-line*. In the former case, the neural network is trained as a separate process and cannot be updated during its use. Vice-versa, an *on-line* neural-network can learn while it is being used and, in the case of financial tools, this possibility is extremely important. Training can be a long and expensive process and is a time-consuming operation. In case of financial forecasting, the operator will be usually required to input a relevant number of items (usually consisting of price of stocks) in the system and to periodically update the network, before this can be used in real-time prediction.

Two different sets of information are needed to perform the training process of the network. First, the designers need to acquire knowledge regarding the aspects of the domain (e.g. the share price of a particular company) that can influence the outcome of the decision-making process [Walczak, 1995]. Second, the designer must collect a huge amount of data that will be used to train the network according to the criteria identified in the first set of information. This data will typically consist of historical price databases regarding price shares connected to the specific criteria identified. An important point to notice is that the data to be collected must be free of *noise* or *errors* which would otherwise cause the production of non accurate solutions/forecasts. The training process can last from hours to weeks, depending on the amount of data that is used for the training. The training of neural networks is becoming a very important issue in the financial community and various techniques for improving the process are being studied [6].

[5] Additional information can be found in many sources, for example [Zahedi, 1993].

[6] For example: [Klimasauskas, 1995]

Trading systems based on neural-networks are nowadays extremely widespread. Neural Networks are commonly used for the prediction of shares prices. They have also been applied to currency price forecasts, future forecasts etc. [Azoff, 1994]. The performance of the system in terms of predictions compares well with those provided by conventional methods, for example regression [Refenes *et al.*, 1994].

Any financial institution has at least one system working in real-time which supports the decision-making process of the operators.

Neural Networks and expert systems can be integrated to overcome the limitations of each technique. Although most of the best systems developed by big financial operators are not publicly available, an enormous quantity of neural-networks systems are available even to the small investor. Many small AI firms specialize in Neural Network systems which are sold to end-users[7]. Neural Networks have been used for other purposes in finance including forecasting foreign exchange rates [Steurer, 1994], [Levitt, 1994], bonds prices forecasting [Kingdon, 1994] and for the prediction of corporate mergers [Sen and Oliver, 1994].

2.3 Integrating conventional and intelligent systems

As we have seen in the previous section, Artificial Intelligence is widely used in finance and is increasingly substituting conventional statistical and probabilistic techniques in many fields of application. Although in the future it might happen that tools based on AI techniques will substitute most of conventional techniques, many of the well-known and efficient conventional techniques will survive and, probably, be integrated with the new technologies. For example, the Moving Average indicator, described in section 2.2.1 will continue to be used as a basis for subsequent analysis.

[7] A brief list of AI companies specialising in Neural Networks products can be found in [Azoff, 1994].

An important issue, therefore, is studying how to integrate the *two different worlds* and the different kinds of artificial intelligence techniques.

We will here briefly analyze some potential points of interest. First of all, Expert Systems (and potentially Neural Networks) can be integrated with databases [Zahedi, 1993]. Expert systems, in fact, can help recognise patterns and errors that would be impossible to recognize for a conventional database and would be left as a task for the user.

Another possibility is to integrate expert systems and neural networks with statistic and probabilistic techniques. For example, we can imagine a tool that, given a particular set of input data (e.g. a historical price database of a share), will first process the data using conventional (and fast) techniques and, if the user is then interested, process the data using Expert Systems and Neural Networks for obtaining additional results, reporting the differences in the forecasts. Moreover, statistical and probabilistic techniques can be used in particular situations where AI techniques fail to produce a sensible result but an answer must be produced anyway. Statistical and probabilistic techniques can also be used to pre-process the data given as input to the Expert System / Neural Network. For example, the seasonal component of a stock's price time-serie can be eliminated using conventional techniques before further processing using a forecasting tool based on Neural Network.

Finally, Expert Systems and Neural Networks can be integrated between themselves using one or another according to specific situations. The two techniques, in fact, present advantages and disadvantages according to the particular situation in which they are used.

2.4 Qualitative tools

We can define *qualitative tools* as tools which process *qualitative* data and produce *qualitative* or *quantitative* information (figure 2.6) [Costantino *et al.*, 1996d].

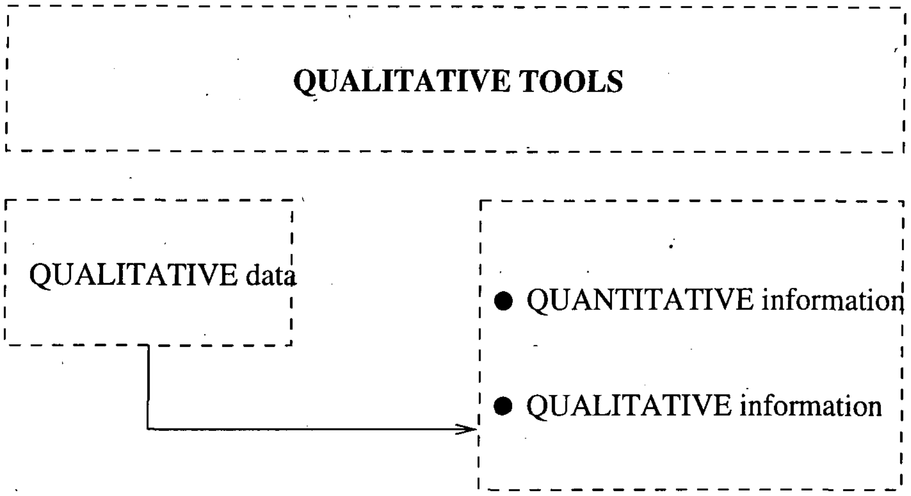

Figure 2.6: The kind of input and output of a qualitative tool

	Explanatory	**Forecasting**	**General Support**
Conventional			Information Retrieval
AI	Natural Language Processing Systems Expert Systems	Natural Language Processing Systems Expert Systems Neural Networks	Natural Language Processing

Figure 2.7: Different kinds of qualitative tools.

While *quantitative* data is easily identifiable in prices of stocks, historical time-series of share, bonds, inflation, interest rate and all sort of relevant numerical data *Qualitative* data is more difficult to define. Quantitative data can be easily used in mathematical or statistical equations, which does not normally apply to qualitative data.

Qualitative data, instead, is difficult to express in a numeric format. For example, data regarding *rumors*, *fears*, *broker's recommendations*, *takeovers* etc. are all qualitative data. A sentence such as:

"there are rumors of a possible takeover of Apple, the troubled computer manufacturer"

represents an information which is extremely relevant to the financial operators, since it will probably immediately cause a marked movement in the quotation of Apple's shares as well as those of the possible buyers. However, taking it into account in a mathematical/statistical equation would be extremely difficult. These kinds of news are extremely important because they affect the expectations of the operators regarding a particular share. The way in which the operators are influenced depends on how an operator perceives the information. Even if, in theory, it could be possible to produce a complete econometric model which takes into account all possible variables and expected behaviors of the players, the complexity of the financial world makes it impossible to produce such a model which would anyway be extremely expensive in terms of the necessary computing-power.

A similar situation can be found in macro-economics. The advanced econometric macroeconomics models employed by the central banks often fail to predict the development of the economical cycles, crisis and expansions. Only very few macroeconomic relations (e.g. interest-rate/investments) are actually widely used and effects of a change in one of the variable is easily predictable.

Qualitative data is much more difficult to process than quantitative. Therefore, while all sorts of quantitative financial tools are nowadays available on the market, very little progress has been done in the processing of qualitative information, which is usually left to the financial operators.

In the financial community, news, rumors and facts are among the most important factors that determine the operators behavior. Operators, in fact, are much more influenced by news than by analysts forecasts or historical price analysis of shares. When a news such as "*the inflation rate is expected to increase next month*" arrives, the consequences are immediately visible and operators base their decisions on their personal experience and on other's people behavior, rather than on expensive and complex forecasts produced by complicated neural-networks financial forecasting systems. Quantitative methods work fine and help the operators' decision-making process by suggesting a "normal" path of the prices but, in the end, what is important is the news and people's reaction to it.

The financial operators and information providers understood the importance of qualitative data as the key-point in the trading decision-making process long time ago. Therefore, the emphasis has been on providing as much relevant qualitative information as possible. Financial operators receive in real-time news regarding companies (announcements, rumors, profit forecasts etc.), macroeconomics (movements of inflation rate, unemployment etc.), politics (changes in general macro economic policy of the government, tax policies etc.). They also have access to huge quantities of past information.

The ideal situation would be a system which is able to process the qualitative information, take into account any possible factor and produce a response such as "buy/sell". Unfortunately, excluding conventional mathematical and statistical techniques which are likely to be impossible to be used for such analysis, current artificial intelligence techniques are not sophisticated enough to produce such an output and, therefore, the decisions are still mainly taken by the operators.

Current development of *qualitative tools* is therefore concerned with "reducing", summarizing or partitioning the news according to specific criteria, rather than inferring decisions from them. This is equivalent to performing simple analysis on quantitative data (e.g. the Moving-Average index), rather than producing a clear operative decision. Explanatory or forecasting qualitative tools have yet to be developed.

Expert Systems have already been analysed as quantitative tools in section 2.2.2. However, they can be successfully used for using qualitative input and qualitative output, such as suggesting buy/sell decisions. In the same way, Neural Networks can be used as qualitative tools. For example, they could be used to produce a direct buy/sell suggestions rather than expected prices of stocks.

One technique that can be considered strictly qualitative is *natural language processing*, which will be discussed in detail in the following chapter.

Chapter 3

Information retrieval and information extraction

3.1 Introduction

In this chapter we introduce the field of information extraction and the most relevant techniques, approaches and systems. Many of the statistical and probabilistic techniques used in information extraction are borrowed from information retrieval. We will therefore briefly analyse information retrieval focusing on the most important techniques which are applied to information extraction. We will later focus on information extraction and the most important systems that have been developed so far (specifically those developed for the MUC-5 and MUC-6 competitions).

3.2 Information Retrieval

Information retrieval is the extraction of relevant information from a large collection of texts. Typically, information retrieval techniques are used to retrieve relevant documents from a large collection of various kinds of documents (newspaper articles, legal documents etc.). The relevant documents can be identified according to specific criteria supplied by the user. These criteria are generally called a *query* which represents the user's requirements for the selection of the relevant documents. This first kind of information retrieval is sometimes called *adhoc* information retrieval (e.g. in the TREC conferences [Harman, 1994c]). Another aspect of information retrieval is the *routing* of a stream of incoming documents, which are discriminated between relevant and non-relevant documents given specific criteria. Some authors [Lewis and Jones, 1996] consider information retrieval as a global term covering any sort of extraction or identification of data in source article and *document retrieval* (DR) as the identification of relevant documents in a collection. We here prefer to consider *information retrieval* as the process used for the identification of relevant documents among a collection of documents, rather than as a global term.

An important aspect of the information retrieval area is efficiency and effectiveness. Efficiency is usually quantified in terms of the computer resources used by the program, such as memory and CPU time.

Efficiency is very difficult to measure as a machine-independent value. Moreover, it should be measured taking into consideration the effectiveness of obtaining the information requested.

Effectiveness is usually measured in terms of *precision* (accuracy) and *recall* (completeness). *Precision* can be thought of as the ratio of the number of relevant documents retrieved to the total number of documents retrieved.

$$Precision = \frac{number\ of\ relevant\ items\ retrieved}{total\ number\ of\ items\ retrieved}$$

Recall is the ratio between the number of relevant documents retrieved and the total number of relevant documents (both retrieved and not retrieved) [Rijsbergen, 1979]:

$$Recall = \frac{number\ of\ relevant\ items\ retrieved}{total\ number\ of\ relevant\ items\ in\ the\ collection}$$

Another measure, sometimes used, is *fallout*, which is defined as the ratio between the number of non relevant items retrieved and the total number of non relevant items in the collection [Harman, 1994c]:

$$Fallout = \frac{number\ of\ nonrelevant\ items\ retrieved}{total\ number\ of\ nonrelevant\ items\ in\ the\ collection}$$

The techniques that are available for extracting relevant information from a text are numerous. It is useful to classify them into the statistical and probabilistic approach, the linguistic approach and the combination of the two approaches.

While the statistical and probabilistic approach makes wide use of these techniques to find the relevant information in the text, with the linguistic approach the goal is to perform a deeper analysis of the source text involving the analysis of the meaning of the sentences in order to select the information that is needed. The third approach combines the two approaches, the techniques based on statistics and probabilistic and the linguistic.

3.2.1 Statistical and Probabilistic approaches to the Information Retrieval

The first approaches to information retrieval use keyword searches and statistical techniques in order to retrieve the relevant documents. These approaches achieve good results in terms of speed, but has many disadvantages mainly related to the fact that the meaning of the text is not understood by the information retrieval application.

The Boolean retrieval

A very simple retrieval method, that also forms the basis of many commercial retrieval services is the *Boolean retrieval.* This method, unfortunately, provides low performance in terms of *precision* and *recall.* The term "Boolean" is used because the query specifications are expressed as words or phrases combined using operators coming from the *Boolean logic.* In the *Boolean method* the documents retrieved are those which are able to match exactly the terms that form the query and thus, there is no distinction among the documents retrieved. The main problem of this method is that it does not allow any ranking of the documents of the collection. In fact, it is quite unlikely that every document will be relevant in the same way as all the others. Moreover, excluding documents from those retrieved because they do not exactly match the query can be reductive. *Boolean retrieval* is an *exact-matching* retrieval method, thus it only allows retrieval of documents that match exactly the query. This is different from *best matching* methods that retrieve documents that match the query in the best way.

The Vector-model

Another approach is based on the frequency of a word in a document and, thus, on the identification of key-terms in the document (this process is known as "indexing") which can be matched with the queries supplied by the user. In fact, a term appearing often in the text may be more important for the identification of relevant information than a term appearing rarely. On the other hand, if the same term occurs in many documents, it will probably be irrelevant for finding the relevant information. Therefore, the specificity of a given term as applied to a given document can be measured by a combination of its frequency of occurrence inside that particular document (the *term frequency* or *tf*) and an inverse function of the number of documents in the collection to which it is assigned (the *inverse document frequency* or *idf*). The *idf* factor can be computed as 1 divided by the document frequency. A possible weighting function for a generic term i appearing in the document j can be [Salton, 1986]:

$$w_{ij} = tf_{ij} \; . \; idf_i$$

The second factor shown in the formula above, the inverse document frequency of a term, can be obtained in advance from a collection analysis, while the term frequencies can be computed from the individual documents, as needed.

Using the weighting formula, the documents of the collection can be ranked in connection with the query of the user. The first documents will be those where the specific terms occur frequently in the document, and very rarely in the rest of the collection. Such terms will in fact distinguish the relevant documents from the non-relevant ones.

The main disadvantage of the simple word-based approach (sometimes also called *vector method* [Croft and Turtle, 1992] considered above is that single words are rarely specific enough for accurately discriminating documents, and their grouping is often accidental [Strzalkowski, 1994]. A better method is to identify groups of words that create meaningful phrases, thus considering not just words to look for relevant information inside a document but phrases. A good example [Strzalkowski, 1994] is the phrase *"Joint Venture"* that can be much more important in a financial article than either *"Joint"* or *"Venture"*. In large databases comprising hundreds of thousands of documents, the use of phrasal terms is not just desirable, but it becomes necessary. The terms that will be used for extracting information from a text are usually pre-processed through four main techniques: *removal of high frequency words*, *suffix stripping*, *detecting equivalent stems* and *addition of synonyms*. High frequency words are eliminated because they are supposed to be too common and, thus, not significant for the identification of relevant information in a document. This phase is normally implemented by comparing the input text with a "stop list" of words that are supposed to be common. Common words can be, for example, *about, into, cannot, our, etc.*. The advantages of the process are not only that non-significant words are removed and will not interfere with the searching process, but also that the size of the original text will be reduced consistently [Rijsbergen, 1979].. The second process, *suffix stripping*, is more complicated. The words involved in the searching process are checked and common suffix are removed from the word, obtaining the stem. For example, verbs are reduced to their stem (e.g. "moved" is reduced to "mov"). A standard approach to do this is to have a complete list of suffixes and to remove the longest possible one. Unfortunately, suffix-stripping can lead to errors in the searching process, and the suffix must be removed taking into account the context. Unfortunately, many words, even if they look equivalent are not and special algorithms must be used, as well as lists of irregular words. Moreover, the suffix-stripping algorithms are often different from one language to another. The most important advantage of the suffix-stripping algorithms consists in the fact that words sharing the same stems should represent the same concept and, thus, the number of words to be used in the search process can be reduced.

Another process that can be useful to improve the recall performance of information retrieval applications is to add to the query synonyms of the original terms or broader terms. The last step that can be useful in order to reach a better performance in the searching process is *normalisation.* This process is normally *lexicon based* and, thus, a dictionary is needed in order to construct the proper word. For example, the word "retrieval" is reduced to "retriev" by the word-stripping algorithm and reconducted to "retrieve" by the normalisation algorithm.

Techniques such as suffix removal, addition of synonyms, addition of related or broader terms can be seen as *recall* - enhancing strategies but, unfortunately the use of these techniques can also lead to a loss of *precision.* In fact, it is often possible to notice a trade-off between *recall* and *precision* and every operation that causes a broadening of the terms to search will generally lead to an increase of *recall*, while a narrowing of the terms will lead to an improvement of the *precision.* In fact it usually happens that the harder a system tries to extract all the relevant information (i.e., the more aggressively configured it is), the more likely it is to extract erroneous information [Sundheim, 1993].

The probabilistic approach

Another approach to information retrieval is based on the use of probability and, thus, the model is called the *probabilistic retrieval* model. Probabilistic information retrieval models are based on the *Probability Ranking Principle* [Belkin and Croft, 1992]. This states that a way to discriminate between the documents of a collection is to rank the documents in the order of their probability to match a certain query. Clearly, the ranking process takes into consideration the limited information available at the time it is made. The probability ranking principle, thus, assumes that it is possible to calculate the $P_{relevance/document}$ and, also, to estimate it accurately. The problem that arises trying to calculate the probability above is that the percentage of the relevant documents among all the others is unknown and, therefore, must be estimated. Following the *Probabilistic Retrieval* model, the most valuable documents for retrieval purposes are those whose probability of relevance to a query is largest [Salton, 1986] and, in this, the approach differs quite a lot from the simple word-matching search. The relevance property of a document is estimated by taking into consideration the relevance of the individual terms in the document. Various different probabilistic formulations have been proposed and they differ mainly in the way in which they estimate the probability of relevance of a document. A possible relevance weighting function can be:

$$tr_i = log\ (\tfrac{N-n_i}{n_i}) + constants$$

where N is the collection size and n_i represents the number of documents in the collection with term i [Salton, 1986].

The Term-Discrimination approach

A different approach to information retrieval is the *Term-Discrimination Model.* This model assumes that the most useful terms for finding relevant information in the collection of documents are those which can distinguish the documents of a collection from each other. Thus, *the value of a term should be measured by calculating the decrease in the "density" of the document collection that results when a given term is assigned to the collection* [Salton, 1986]. The "density" of the documents will be high when they are indexed by many of the same terms. With the Term-Discrimination approach the terms that occur often in all the documents of the collection will become the less useful to search for relevant information, while the terms that make it possible to distinguish between the documents will be preferred. In other words, the best words to be used should be those which appear neither too often in the documents, nor too rarely. The *Term-Discrimination Model*, thus, is based on the assumption that terms that are able to distinguish a document among the others document of the collection are those which are enough *specific* but not too much to be rare.

Other non linguistic approaches to information retrieval

In recent years, researchers moved towards new directions in information retrieval. The common thought is that better performance in information retrieval tasks can be obtained if the algorithms are able to "understand" in some way the meaning of the text in order to extract the relevant embedded information. In this view [Croft and Turtle, 1992], information retrieval is an *inference* or *evidential reasoning* process in which the goal is to estimate the probability that the information needed by the user is available given a document as "evidence". The Retrieval Method based on *Inference Networks* follows this approach [Croft and Turtle, 1992] and it is based on *Bayesian inference networks* [Belkin and Croft, 1992] which are directed, acyclic dependency graphs in which nodes represent prepositional variables or constants and edges represent dependence relations between propositions. The basic document retrieval inference network consists of two component networks: a document network and a query network. The document network represents the collection of the documents through a variety of representation schemes. The document network is built at the beginning for a particular document collection, and it does not change while the search process takes place. While the document network consists of many nodes, the query network consists of a single node, which represents the query supplied by the user, (that is the information needed). Moreover, while the document network is built when a particular collection of documents is given, the query network is built each time the user supplies a query and is modified during the search processing as existing queries are redefined or new queries are added to perform better search. A complete description of the algorithms can be found in [Croft and Turtle, 1992]. However, at the moment the *inference networks* method does not seem to be largely used in information retrieval applications.

3.2.2 The Linguistic approach to information retrieval

The Linguistic approach to information retrieval is based on the assumption that the statistical and probabilistic techniques are limited in the sense that they are not able to understand the meaning of the text. Other authors [Jones, 1992] refer to the linguistic approach as a *meaning-oriented* approach. The linguistic approach is based on the idea that a technique that could deeply understand the text could be much better than statistical and probabilistic techniques. Once the meaning the text would have been understood, the user could retrieve information by simply providing queries in natural language to the system, obtaining the information needed in the desired form. In other words [Stanfill, 1992], an information retrieval application based on the Linguistic Approach should be able to process the collection of documents by a natural-language understanding system, and to extract the meaning of the documents. The user's requests should be processed by the same natural-language understanding system and the information could be easily identified since the system has already understood the meaning of the text.

Unfortunately, the systems that have been built until now are still not able to cope with the free-text of a general domain and, usually, work only in limited domain. In addition, linguistic approaches tend to be slower than systems based on statistics and probability and require very high computing power. Current research in information retrieval is therefore oriented towards the improvement of techniques based on statistics and probability through the use of linguistics methodologies and these approaches are usually called *knowledge-based* approaches.

3.2.3 Knowledge-based approaches (weak methods)

Knowledge-based approaches make use of natural language processing techniques to improve the performance of statistical or probabilistic methods in information retrieval. In the group of these techniques it is also useful to include the *weak methods* [Jacobs, 1992], since they make use of NLP techniques but not in order to completely understand the meaning of the text. Many of the advanced approaches to information retrieval can be included in this group. For example some of the applications used for TREC (the second Text REtrieval Conference) (see section 3.3) are based on statistical and probabilistic techniques supported by NLP methodologies.

The use of natural language processing techniques combined with methods based on statistics and probality can follow different paths. One possibility is [Hobbs *et al.*, 1992] to ignore certain information that is present in the text, focusing only on part of the information. In the development of TACITUS [Hobbs *et al.*, 1992] the choice was to use a statistical keyword filter to firstly select the information that would be later processed. In this way, it should be possible to reach a good level of speed, without incurring in the typical problems of parsers. Normal approaches to Information Retrieval are able to convert natural language queries into a generic query like, for example:

I'm interested in algorithms for parallel computers.

into

$$f(x) = C_1 * parallel + C_2 * algorithms + C_3 * computers.$$

DR-Link [Liddy and Myaeng, 1994], for example, is able to construct a logical query from a natural language query supplied by the user by using a sub-language grammar.

An interesting approach that shows a deeper use of NLP techniques in information retrieval is proposed by [Strzalkowski, 1994]. The system proposed is still based on a pure statistical core but is assisted by various natural language modules such as a query translator for the information requested by the user and the improvement of indexing by using natural language generated compound terms. The authors emphasise the fact that the system should be a careful compromise between purely statistical non-linguistic approaches and those requiring deep expensive semantic analysis. The first step taken by the system is to process the database with a parser. Furthermore, some sentences are extracted from the parse-tree and used as compound indexing terms in additional to single-word terms. The parsing algorithm initially attempts to generate a complete analysis for each sentence. However, unlike many other parsers, after a certain amount of time spent trying to parse the full sentence, the parser enters the *skip-and-fit mode* in which tries to "fit" the parse. In this mode, the parser attempts to reduce incomplete constituents [Strzalkowski, 1994], usually, skipping portions of input and starting again the process with less information. The skipped fragments are later analysed by a simple phrasal parser that looks for noun phrases and relative clauses and then connects this to the main parse structure coming from the information successfully parsed. The system also processes user-queries by parsing them and recognising all the indexing terms. After the final query is constructed, the collection of documents is searched. Another system that makes use of a parser for performing the analysis of a text is the Clarit system [Evans and Lefferts, 1994].

The Conquest system [Nelson, 1994] makes deeper use of the NLP technology and is based on a dictionary including a semantic network for both indexing and query analysis. The dictionary is composed by a list of words each of which presents multiple meanings. Moreover, each meaning contains syntactic information and a dictionary definition. The semantic network is composed of nodes which correspond to meanings of words and are linked to all other related nodes. The system uses the information in the semantic network to increase the performance of indexing. The natural language queries provided by the user are also processed by the system. Queries, however are not understood in a real sense. ConQuest, in fact, attempts to understand only the meaning of single words and the importance of them. The final query used to retrieve the relevant documents from the collection using statistical techniques is built by looking up the meanings and the related terms of the words in the semantic network. ConQuest also includes modules for suffix-stripping and normalising.

3.3 The TREC competitions

A particularly important group of systems performing information retrieval tasks are those that competed in the TREC 1-4 competitions [Harman, 1994c], [Harman, 1994b], [Harman, 1995]. The systems that entered the competitions had to perform specific tasks for each of the conferences, which were evaluated using the TREC evaluation metrics.

...

(desc} Description:

Document will identify a type of natural language processing technology which is being developed or marketed in the U.S.

...

(narr} Narrative:

A relevant document will identify a company or institution developing or marketing a natural language processing technology, identify the technology, and identify one or more features of the company's products.

(con} Concept(s}:

1. natural language processing
2. translation, language, dictionary, font
3. software applications

...

Figure 3.1: A TREC-2 topic.

3.3.1 Tasks

The target of the TREC (Text REtrieval Conferences) is to encourage research in information retrieval using large (several gigabytes) data collections [Harman, 1994a]. The first relevant TREC competition is TREC-2 (The Second Text REtrieval Conference) [Harman, 1994c], which formed the basis for the following TREC competitions. TREC-2 comprised two different types of information retrieval: *ad-hoc* and *routing.*

Another important point of the competitions was the construction of the queries, which could be done either automatically, manually or semi-automatically for a particular topic. A topic consisted in information regarding the kinds of documents to be extracted from the collection. An example of TREC-2 topic can be seen in the example shown in figure 3.1

In the case of automatic queries construction, a TREC-2 system was supposed to extract the relevant information from the topic and to produce the query without human intervention. There were several collections of document for training and evaluating the systems, as well as associated topics for the extraction of proper queries.

The TREC-3 competition was held one year after the TREC-2 competition. The tasks of the competition were rather similar to those of the TREC-2 competition and included a "routing" and an "adhoc" task. In the routing task the questions asked to the systems were the same all times while the data changes. Differently, in the adhoc task new questions were asked against a static set of data [Harman, 1994b]. In addition to the English language used in the TREC-2 competition, TREC-3 introduced a new language, Spanish, for testing the portability of the existing information retrieval systems.

The topics for the *automatic*, *manual* and *interactive* construction of queries in TREC-3 were different from those used in TREC-2. The topic were in fact much shorter and missing of the complex structure of the TREC-2 topics. The list of relevant concepts was also eliminated and the topics included a mini-knowledge about a topic such as a real user of the systems might posses. Moreover, the topics were written by the same group of users.

The TREC-4 competition [Harman, 1995] was held in December 1995. The main tasks of the competition were similar to those performed in TREC-3 with the addition of specific "*focussed*" tasks. The five specific tasks, called *tracks* were: a multilingual track, an interactive track, a database merging track, a "confusion" track and a filtering track. The multilingual track was designed to evaluate the performance and the difficulties in porting the systems towards new languages, Spanish in this particular case. The results show that the developers were able to quickly adapt their systems and the performance were similar to those for the English language. The interactive track was designed to study the possibility of having better interactions with the system during the retrieval of the information, rather than a predefined batch.

The database merging track was designed to check the performance of the systems on 10 different sub databases to improve the flexibility of the systems.

The "confusion track" regarded the processing of news which were partially corrupted or imprecise. The task was intended to test the systems on possible real data, such as from OCR programs, speech recognition systems etc. The TREC-4 conference was therefore intended to improve the system's flexibility and portability across new domains, rather than improving them in standardised tasks.

The TREC-5 competition [Harman, 1996b] was held in November 1996. The main tasks of the competition were similar to those performed in the TREC-4 competition with the addition of a new set of documents and topics to support the Chinese language [Harman, 1996a, A. Smeaton, 1996]. In addition, new document corpus and new topics were used for the Spanish tasks focusing on European Spanish texts.

3.3.2 Techniques

The main techniques employed in the TREC competitions have already been described in the previous sections. These include statistical, probabilistic, linguistical and knowledge-based approaches to information retrieval. We will not here describe in detail each of the specific systems that participated in the TREC competitions.

3.3.3 Evaluation metrics

An important aspect of the TREC conferences is the evaluation of the results. The TREC-2 evaluation metrics were mainly based on the classical information retrieval measures: precision and recall. The evaluation measures used in TREC-2 are particularly relevant because of the high number of systems that competed and, also, for the high number of documents tested. The evaluation of the subsequent TREC-3 and TREC-4 competitions was also based on the TREC-2 evaluation metrics. The first choice that had to be made to evaluate the performance of the TREC-2 systems was the number and kind of documents to be used for the evaluation of the systems. One solution was the evaluation of all the documents. However, this possibility was not chosen because of the large amount of documents. Another possibility was to extract a sample of articles from the collection and use those for the evaluation of the systems. The solution chosen for TREC-2 was to evaluate the systems on the first 1000 articles for each topic.

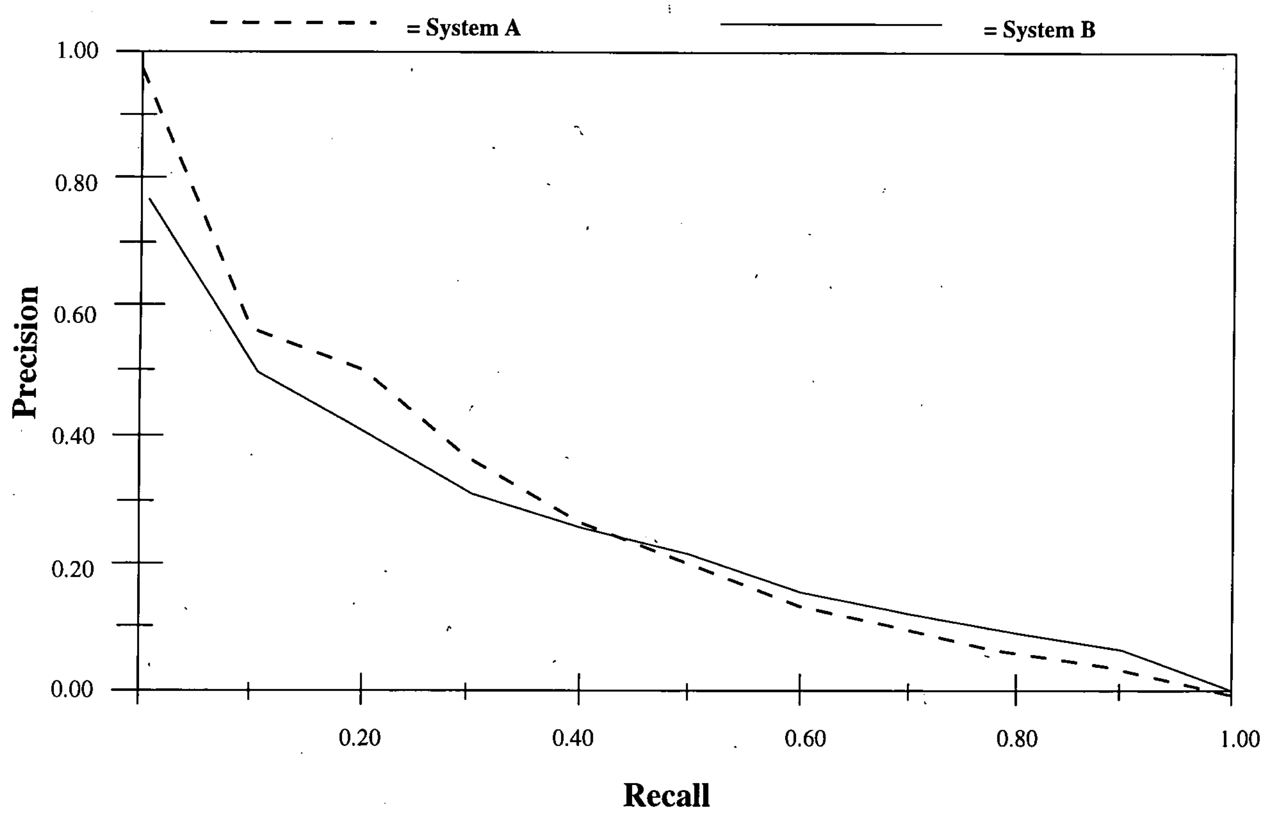

Figure 3.2: The recall/precision curve.

The recall/precision and recall/fallout curves

The main measures used in TREC-2 were the standard precision, recall and fallout. However, these measures were combined in order to obtain the recall/precision and recall/fallout curves.

The recall/precision curve is plotted as follows: the x axis plots the recall values at fixed levels of recall where the y axis plots the average precision values at given recall values. The curves have been plotted in TREC-2 over 50 topics. The use of the curves assumes a ranked output from a system. Systems which do not rank documents were not tested in the TREC-2 competition.

In figure 3.2 the recall/precision curves of two systems are shown. System A shows higher precision at the low recall end of the graph and, therefore, is more accurate, while system B shows higher precision at the high recall end of the graph and, thus, will give a more complete set of relevant documents.

The recall/fallout curve is another way to show the results and, normally,

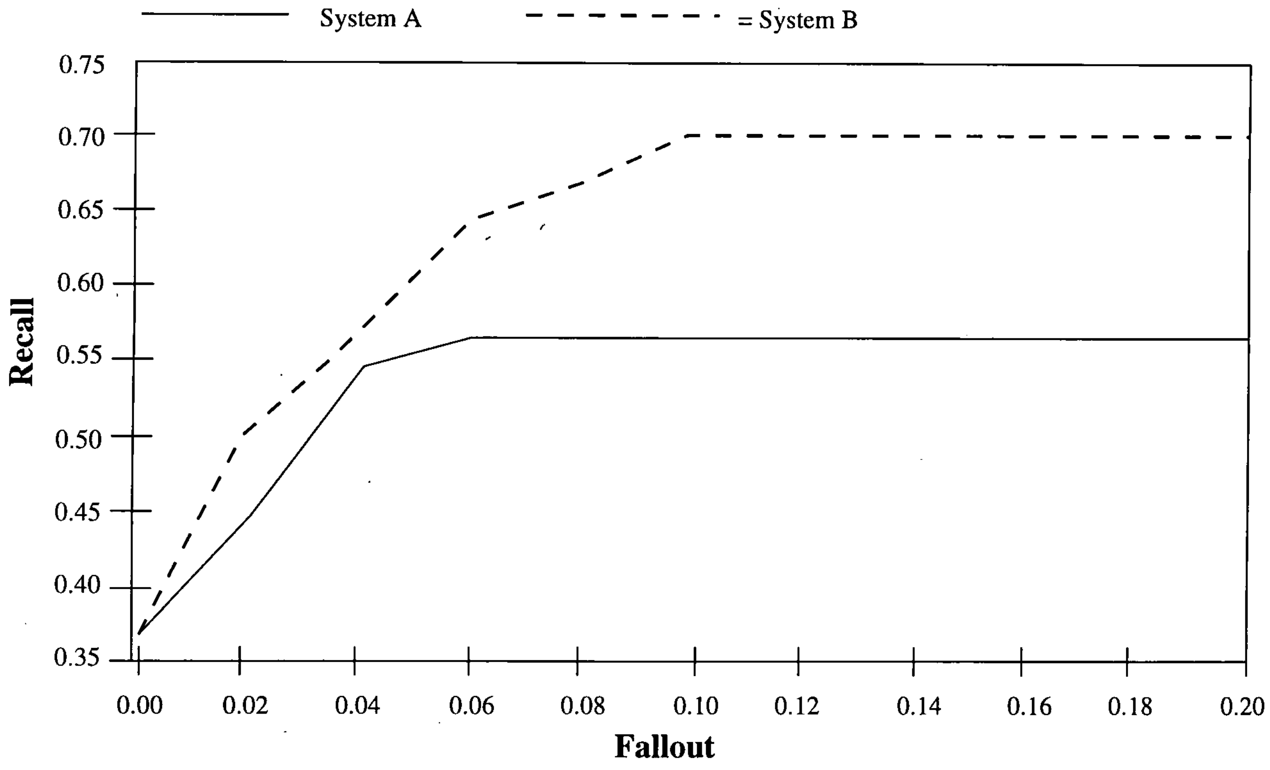

Figure 3.3: The recall/fallout curve.

shows the same order of performance as does the recall/precision curve. The recall/precision curves show the retrieval result as an user would read them while the recall/fallout curve is more useful to understand the ability of a system to retrieve non-relevant material. In particular, the fallout measure shows the "discrimination power" of a particular systems on a large document collection. For example, system A in figure 3.3 has a fallout of 0.02 at a recall of about 0.48; this means that this system has found almost 50 per cent of the relevant documents, while retrieving 2 per cent of the non-relevant documents.

Single-value measures

In addition to the recall/precision and recall/fallout curves, in TREC-2 other measures were used.

The first one is called *non-interpolated average precision* and represents the area under an ideal recall/precision curve. The index is computed firstly by computing the precision average for each topic. This second average is obtained as follows:

$$Precision\ average\ for\ each\ topic = \frac{precision\ after\ each\ relevant\ document\ retrieved}{total\ number\ of\ relevant\ documents\ retrieved}$$

Finally, the *non-interpolated average precision* index is obtained averaging the precision average indexes obtained for each topic on the total topics of the set of documents.

The second "single-value" measure is an average of the precision for each topic after 100 documents have been retrieved for that topic.

The TREC competitions have been particularly useful because they involved the evaluation of information retrieval systems in the identification of relevant documents in very large collections of source documents. These collections included several gigabytes of source documents and, therefore, the speed of the information retrieval engines had to be taken into account by the participants.

3.4 Information Extraction

The goal of *information extraction* is to extract specific kinds of information from a document [Riloff and Lehnert, 1994]. In other words, the input of the system is a stream of text while the output is some representation of the information to be extracted from the texts [Hobbs and Israel, 1994]. The main difference between information retrieval and information extraction is that while the main objective of information retrieval is to identify relevant documents among a generic collection of documents, information extraction tries to identify relevant information inside documents and produce a representation of the information. Thus, information extraction systems must not only be able to judge the relevance of a particular document in the collection but, also, identify relevant information inside the documents. Finally, the output of the two systems is rather different. The output of a classical information retrieval system will simply be the collection of relevant documents retrieved, while the output of an information extraction system can be of different types. One kind of output can be thought of simply as the fragment of relevant text found in the documents of the collection (see figure 3.4).

Source: DowVision, a service of Dow Jones.
Date: Jan 19,1995 Time: 9:34 am

Banker Trust New York Corp. (BT) said its 1994 earnings were affected by persistently difficult market conditions, which hurt trading revenue.

The company said trading revenue was 49 dlrs million in the latest fourth quarter, down from 449 dlrs million in the year-ago quarter, while trading-related net interest revenue fell to 50 dlrs million in the latest fourth quarter from 187 dlrs million a year earlier.

Banker Trust said many of its proprietary trading businesses, principally fixed income instruments, recorder lower revenues during the latest fourth quarter as market conditions remained generally unsettled. The company said trading results also declined significantly in the emerging markets of Asia and Latin America, and said the volume of traditional risk management products slowed.

The company said revenue increased from its client-related businesses that provide financing, advisory and transaction processing services.

Bankers Trust said it reclassified 423 dlrs million of leveraged derivative contracts as receivable in the loan account and placed them on a cash basis during the last fourth quarter. Of the amount, the company said 72 dlrs million was subsequently charged off to the allowance for credit losses. About half of the remainder related to transactions with Procter and Gamble Co. (PG).

With the transfers and charge-offs, the company said it has taken action on the leveraged derivative transactions that likely will not perform according to the contract and has charged off the balances deemed to be uncollectible.

Bankers Trust said net charge-offs for the latest fourth quarter were 85 dlrs million, compared with 184 dlrs million a year ago.

The system could identify the relevant information in the sentences:

"Banker Trust New York Corp. (BT) said its 1994 earnings were effected by persistently difficult market conditions, which hurt trading revenue. The company said trading revenue was 49 dlrs million in the latest fourth quarter, down from 449 dlrs million in the year-ago quarter, while trading-related net interest revenue fell to 50 dlrs million in the latest fourth quarter from 187 dlrs million a year earlier."

Figure 3.4: Information Extraction using fragments from the original document.

A car bomb exploded outside the Cabinet Office in Whitehall last night, 100 yards from 10 Downing Street. Nobody was injured in the explosion which happened just after 9 pm on the corner of Downing Street and Whitehall. Police evacuated the area. First reports suggested that the bomb went off in a black taxi after the driver had been forced to drive to Whitehall. The taxi was later reported to be burning fiercely. (THE DAILY TELEGRAPH 31/10/92)

Template produced:

Incident: The bomb explosion outside the Cabinet Office and outside 10 Downing Street in a black taxi.

Where: Outside the Cabinet Office and outside 10 Downing Street in the black taxi that a driver drove to Whitehall.

When: last night (30 October). When a forceful person forced a driver to drive t Whitehall a black taxi that fiercely burned.

Responsible: Unknown.

Target: Cabinet Office. 10 Downing Street.

Damage: Human: nobody. Thing: the black taxi that a driver drove to Whitehall.

Source: Telegraph.

Source-date: 31 October 1992.

Certainty: facts.

Other relevant information: Police evacuated Downing Street.

Figure 3.5: Information extraction using templates.

Alternatively, the output of an information extraction system may consist of *templates* (figure 3.5). A *template* is as a structure with a predefined set of slots, one for each type of information to be extracted from the text [Riloff and Lehnert, 1994].

Clearly, the way the template is presented to the end user is crucial for the effective usefulness of an information extraction system. In [Hobbs and Israel, 1994] three different but interacting considerations are indicated for an effective template:

- the template as representational device;
- the template as generated from input;

- the template as input to further processing, by humans or programs or both.

The design of templates has to take into consideration many different aspects, often in competition between themselves [Onyshkevych, 1993]:

descriptive adequacy: the template should present all the relevant information for a particular task or application. The relevant information should include all supporting information such as measurement units etc.;

clarity: the information included in the template should not be ambiguous, for both the human end-user of the template or for further processing of the information by computer applications;

determinacy: there should be only one way of representing the extracted information inside a template;

perspicuity: *"the degree to which the design is conceptually clear to the human analysts who will input or edit information in the template or work with the results"* [Onyshkevych, 1993];

monotonicity: adding new information to a template already filled, there should be no new objects in the template and none of the existing slots should be removed;

reusability: templates can be used in other domains and, therefore, must be designed to be easily converted.

3.4.1 The scripts / frames systems

The first attempts to understand the meaning of a text and to produce summaries or templates from it were mainly based on the concepts of *frames* or *scripts*. A frame can be defined [Tait, 1982] as *"networks of nodes and relations the highest level of which are things which are always true about the situation, the lowest level being slots which are filled by the details of a particular instance of the situation."* In a similar way, a script can be defined [Tait, 1982] as *"a predetermined, stereotyped, sequence of actions that define a well-known situation."* The systems based on scripts and frames were thus based on a collection of pre-defined stories and their task was therefore to identify which one was suitable for the text analysed. Various attempts have been made using the "script - frame" approach. The most important systems based on these approaches are here analysed. However, the analysis will not be very deep, because more advanced techniques have been introduced in recent years.

SAM

SAM [Schank and Riesbeck, 1981] is a system based on *scripts*. Each script provides pre-stored expectations about what will be read, based upon what has already been seen. The main problem of SAM, as well as for other systems based on scripts (or frames) is that it cannot be used for analysing texts for which it does not have a pre-stored script. SAM, in facr, does not deeply understand the meaning of the text and bases its knowledge on the scripts. Moreover, SAM, unlike other programs based on script or frames, is not prepared to deal with unknown information in a text. This means that, if a particular text does not exactly match any script, the program will fail to classify and predict its contents and, therefore, will not be able to produce any output for it. The main module in SAM is the *script applier* whose task is to introduce the largest script for a particular story. Once the script has been applied, predictions are made for what has to be the subsequent input. If the following text does not match any more the predictions contained in the script, the script applier will look for another script. Clearly, the main problem arises when the script applier is not able to match any script to the text.

PAM

A different approach is followed by PAM [Schank and Riesbeck, 1981] which identifies two main problems of the script / frames approach. The first is that it is not always possible to identify a single frame which classifies the entire text. The second is that the selection of an appropriate frame for a specific text can be ambiguous. Some texts, in fact, can potentially fit in more than one frame. Moreover, the authors of PAM consider the fact that it is very unlikely that one single story (text) is based on a single goal and it is more likely that in a single story different goals will appear together. PAM is thus based on the analysis of the goals that can be found in a particular story [Schank and Riesbeck, 1981]:

goal subsumption: a situation in which many recurrences of a goal are planned at the same time;

goal conflict: a story can include goals in conflict between themselves;

goal competition: different goals can be in competition.

The behaviour of PAM is therefore directed to the understanding of the goals of a particular text and, thus, predict the subsequent text.

FRUMP

Another script-based system is FRUMP by De Jong [DeJong, 1982]. The program, like others, tries to match a particular sketch script to the text being analysed and tries to make predictions about the further contents of the text. The main characteristic of this program, is that it just skips the parts of the text that do not satisfy its predictions.

Scrabble

Another script/frame based system here considered is *Scrabble* by Tait [Tait, 1982]. One observation made by Tait in his analysis is that a system should not skip over sections of text which do not follow the pre-loaded scripts. Scrabble is composed by five main modules: an English semantic parser, a combined stereotype management module, a stereotype application module, a text representation summariser and an English generator. The interesting modules in this context are the stereotype management module, the stereotype application module and the English summariser.

The *stereotype management module* analyses the text coming from the parser and suggests a possible stereotype suitable for the text. It is important to notice that, unlike other systems, more instances (scripts) can be activated at the same time. Thus, the stereotype management module must also be able to decide whether an instance has to be deactivated or a new instance activated. An instance that is not used in a particular situation but has not been removed yet is called "suspended".

The *stereotype application module* waits for the incoming text and checks if the predictions available in the currently activated instances, supplied by the *stereotype management module* are satisfied.

The task of the *summary generator* is to produce the summary of the original text and supply it to the English generator for the final output of the program. The summary generator receives three kinds of input. The first is the list of all activated and suspended stereotype-instances. The second is a list of all parsed sentences that were not expected by the activated instances (this second input is given because Scrabble does not skip unexpected part of a text like FRUMP). Finally, the third input that is supplied to the summary generator is a *"data-structure which represents the textual interrelationships of the propositions in the input text which gave rise to the elements of the other two"* [Tait, 1982]. The first step taken by the summary generator is to attempt to choose between related instances used to analyse the text by further processing them. The summary generator attempts, in fact, to reduce the number of instances. This means that some instances may be killed or suspended for a particular text. The process in this step differs from the equivalent one made in the *stereotype management module*, for the fact that the assumption made in the summary generator algorithm are much stronger that in the stereotype management module, so that less instances are analysed. Once the relevant instances have been chosen, the elements that will be later included in the summary can be also identified. However, the contents of the summary is not yet known and, therefore, each stereotype-instance is processed to produce the data-structure coming from the input and this is combined with the part of the text that was unexpected and, thus, impossible to process. The approach suggested by [Tait, 1982] seems to be better than others in the sense that it also considers the parts of the text that were unexpected and does not just skip them.

ATRANS

The first commercially successful program based on the script/frames approach was ATRANS, from Cognitive Systems [Hederman, 1992]. The task that ATRANS (Automatic Funds TRANSfer Telex Reader) performs is to extract information from Telex messages [Lytinen and Gershman, 1986]. More precisely, the messages that ATRANS processes are requests for money transfers that banks send to each other. ATRANS first extracts the relevant information and then uses it to produce an output that can be later used for an automatic execution of the transfer. The form of the output can be viewed as a template containing information such as the amount of money to be transferred, the date, the beneficiary and so on. The Telex messages given as input to ATRANS can vary considerably due to the fact that a money transfer can be sent from any part of the world. In fact, the English form of the messages will vary in spelling, sentence construction, standard abbreviations, amounts, date conventions and so on. ATRANS is able to extract correctly 80 per cent of the relevant information in the Telex messages, while about 15 per cent of relevant information is lost and 5 per cent is identified incorrectly. The evaluation, however, was not carried out within a formal framework and was influenced by the very restricted domain.

ATRANS is based on the script - frames approach. This means that, as well as other systems based on this approach, it will first try to fit an appropriate script to identify the relevant information in the text. There are various script available in the system, according to the different kind of transfers that can be processed such as, multiple intermediary banks, different methods of payments, more then one beneficiary and so on.

The architecture of ATRANS is similar to that of other script - based systems. ATRANS consists of four parts: the *message classifier*, the *text analyser*, the *message interpreter* and the *output formatter.*

The *message classification* module is similar to *script - applier* modules found in other systems and determines the type of message currently processed, choosing the appropriate script from those available in the system. However, more than one script can be applied to the same Telex. This mainly happens in the case of multiple transfers and the *message classification* module identifies the common portions of the transfer (using one script) and composes several single transfer messages.

The output of the *message classification* module is given as input to the *text analyser* module which is defined as the "heart of the system" by the authors. This module uses the chosen script and the dictionaries available and tries to identify the frames being referred in the text following the predictions of the script, such as, for example, payment, test and cover.

The *analyser* does not try to verify the information extracted or check its consistency. This task is performed by the *message interpreter* module which *"verifies and consolidates the extracted information items".* However, the output of the *message interpreter* is still in an internal representation form.

The conversion from the internal representation to the final output form is made by the *output generator* module. The form varies according to the particular user of the system (e.g. the Swift and Chips banking networks).

Systems based on the script/frame approach were the first attempt to produce summaries and templates from a text. However, many other approaches have been developed in subsequent years and, thus, they will not analysed here in more detail.

3.5 The MUC competitions

A particular group of interesting systems are those developed for the MUC 1-6 (Message Understanding Conference) tasks, [Grishman and Sterling, 1989], [DAR, 1991], [DAR, 1992], [DAR, 1993], [DAR, 1995]. The target of the MUC competitions was to improve the technology of information extraction systems performing specific tasks set for each of the MUC conferences.

The procedure was equivalent for all MUC competitions. The first step was the definition of the tasks to be performed by the participating systems. Once the tasks had been defined, a set of training documents, the associated keys and an automatic scoring program were made available to the participants. The participants could use the set of training documents for developing and training the systems and carrying out an initial unofficial evaluation of the systems. After a specific amount of time allowed for the development and training of the systems, the actual evaluation was carried out. A final set of evaluation documents was released to the participants which were asked to submit the results using this set of documents. The final evaluation of the results was carried out matching the results produced by the systems against a set of pre-defined keys for the set of evaluation documents. In the MUC-6 competition one of the evaluation tasks was released to the participants only one month prior to the final evaluation to emphasise the portability of the systems.

We will briefly analyse each of the tasks of the MUC conferences and in particular the MUC-5 and MUC-6 conferences, the main techniques employed by the systems that participated and the measures used for the evaluation.

3.5.1 The MUC Tasks

The first two MUC competitions, known as MUCK-I and MUCK-II [Grishman and Sterling, 1989] represented initial experiments related to the evaluation of systems performing rather simple tasks on source texts. However, the evaluation set was extremely limited (just 5 texts for the MUCK-II competition) and, therefore, the results themselves are not particularly useful. The MUC-3 evaluation, instead, was significantly more relevant, both in terms of tasks and systems which entered the competitions. However, the most important MUC competitions in terms of kind of tasks, number and quality of the systems that entered the competition and results were the number 5 [DAR, 1993] and number 6 [DAR, 1995] which will be here described in more detail.

TST1-MUC3-0080
BOGOTA, 3 APR 90 (INRA VISION TELEVISION CADENA 1) - [REPORT] [JORGE ALONSO SIERRA VALENCIA] [TEXT] LIBERAL SENATOR FEDERICO ESTRADA VELEZ WAS KIDNAPPED ON 3 APRIL AT THE CORNER OF 60TH AND 48TH STREETS IN WESTERN MEDELLIN, ONLY 100 METERS FROM A METROPOLITAN POLICE CAI [IMMEDIATE ATTENTION CENTER]. THE ANTIOQUIA DEPARTMENT LIBERAL PARTY LEADER HAD LEFT HIS HOUSE WITHOUT ANY BODIGUARDS ONLY MINUTES EARLIER. AS HE WAITED FOR THE TRAFFIC LIGHT TO CHANGE, THREE HEAVILY ARMED MEN FORCED HIM TO GET OUT OF THIS CAR AND GET INTO A BLUE RENAULT.
HOURS LATER, THROUGH ANONYMOUS TELEPHONE CALLS TO THE METROPOLITAN POLICE AND TO THE MEDIA, THE EXTRADITABLES CLAIMED RESPONSIBILITY FOR THE KIDNAPPING. IN THE CALLS, THEY ANNOUNCED THAT THEY WILL RELEASE THE SENATOR WITH A NEW MESSAGE FOR THE NATIONAL GOVERNMENT.
LAST WEEK, FEDERICO ESTRADA VELEZ HAD REJECTED TALKS BETWEEN THE GOVERNMENT AND THE DRUG TRAFFICKERS.

Figure 3.6: A MUC-3 document in the terrorist domain.

The MUC-3 Tasks

The MUC-3 task was to extract information about terrorist attacks from articles from newspapers, TV and radio news, speech and interview transcripts, rebel communications etc. The systems had to extract the relevant information and represent it into a predefined template. Differently from MUCK-I and MUCK-II, the MUC-3 competition required the systems to discriminate between relevant and irrelevant information (information filtering). In figure 3.6 an example MUC-3 article is shown, while in figure 3.7 an example MUC-3 template produced from the same article is shown.

The systems that entered the competition were based on various techniques such as: statistical, key-word, finite-state analysis, deep natural language processing etc. The specific systems will not be here analysed, since most of them participated in the subsequent editions of the MUC competitions.

0. MESSAGE ID: **TST1-MUC3-0080**
1. TEMPLATE ID: **1**
2. DATE OF INCIDENT: **03 APR 90**
3. TYPE OF INCIDENT: **KIDNAPPING**
4. CATEGORY OF INCIDENT: **TERRORIST ACT**
5. PERPETRATOR: ID OF INDIV(S): “THREE HEAVILY ARMED MEN”
6. PERPETRATOR: ID OF ORG(S): “THE EXTRADITABLES” / “EXTRADITABLES”
7. PERPETRATOR: CONFIDENCE: CLAIMED OR ADMITTED: “THE EXTRADITABLES” / “EXTRADITABLES”
8. PHYSICAL TARGET: ID(S): * 9. PHYSICAL TARGET: TOTAL NUM: * 10: PHYSICAL TARGET: TYPE(S) * 11: HUMAN TARGET: ID(S): “FEDERICO ESTRADA VELEZ” (“LIBERAL SENATOR” / “ANTIOQUIA DEPARTMENT LIBERAL PARTY LEADER” / “SENATOR” / “LIBERAL PARTY LEADER” / “PARTY LEADER”)
12. HUMAN TARGET: TOTAL NUM: 1
13. HUMAN TARGET: TYPE(S): GOVERNMENT OFFICIAL / POLITICAL FIGURE: “FEDERICO ESTRADA VELEZ”
14. TARGET: FOREIGN NATION(S): -
15. INSTRUMENT: TYPE(S) *
16. LOCATION OF INCIDENT: COLOMBIA: MEDELLIN (CITY)
17. EFFECT ON PHYSICAL TARGET(S): *
18. EFFECT ON HUMAN TARGET(S): -

Figure 3.7: A MUC-3 terrorist template.

The MUC-4 Tasks

The MUC-4 competition [DAR, 1992] was held one year after the MUC-3 competition. The tasks for the MUC-4 competition were rather similar to the MUC-3 ones. Few differences can be found in the templates, where some slots which included two pieces information were splitted into two separated slots. For example, the MUC-3 slot (**TYPE OF INCIDENT**) became two MUC-4 slots (**INCIDENT: TYPE** and **INCIDENT: STAGE OF EXECUTION**). Similarly, other slots were separated and few other changes were carried out on the template definition.

The main change in MUC-4 regarded the evaluation metrics [Chinchor, 1992] which, for the first time, included the *F-measure* for a combined evaluation of *precision* and *recall.*

A greater number of systems entered the MUC-4 competition. We will not describe these systems and the techniques on which they were based, since most of them entered the subsequent MUC-5 competition.

The MUC-5 Tasks

The MUC-5 competition comprised two domains (joint ventures and microelectronics) and two languages, English and Japanese, thus obtaining four language/domain pairs. Each pair was associated to about 1200 to 1600 articles. In the case of the joint venture domain for the English language (the only domain here considered) the articles were extracted from more than 200 sources, including the Wall Street Journal, the Jiji Press, the New York Times, the Financial Times, the Kyodo Services and a variety of other technical publications in fields such as communications, airline transportation etc. The articles were mainly extracted using statistical and probabilistic information retrieval techniques. However, manual filtering techniques were also used. A number of irrelevant documents, about 5 per cent of the total, were also included to test the system's ability to discriminate between relevant and irrelevant documents in the collection. Each of the documents of the collection was associated with a hand-made template to be used for the subsequent evaluation by

Bridgestone sports co. said Friday it has set up a joint venture in Taiwan with a local concern and a Japanese trading house to produce golf clubs to be shipped to Japan.
The joint venture, Bridgestone sports Taiwan co., capitalised at 20 million new Taiwan dollars, will start production in January 1990 with production of 20,000 iron and "metal wood" clubs a month. The monthly output will be later raised to 50,000 units, bridgestone sports official said.
The new company, based in Kaohsiung, southern Taiwan, is owned 75 pct by Bridgestone sports, 15 pct by union precision casting co. of Taiwan and the remainder by Taga co., a company active in trading with Taiwan, the officials said.

...

With the establishment of the Taiwan unit, the Japanese sports goods maker plans to increase production of luxury clubs in Japan.

Figure 3.8: A MUC-5 document in the joint-venture domain.

the scoring-program. The developers had access to numerous sources of different kinds of data, such as the English language Gazetteer, provided to regularise geographic location information, lists of currency names and abbreviations, national adjectives, lists of countries and used abbreviations, the hierarchical classification of all the industry or business type in the U.S.

The joint venture domain of MUC-5 is relevant in the context of this work because it can be considered a subset of the financial domain. However, the joint-venture domain was extremely limited because it considered only a very limited partition of the financial domain. The MUC-5 systems were in fact built to skip over any kind of information not regarding the joint-venture domain and, even if the overall results of the competition were in absolute terms significant, the restricted domain in which they were achieved has to be taken into account. In figure 3.8 a typical joint venture MUC-5 article is shown.

A MUC-5 template was supposed to be able to identify the joint-venture, the participants, the capital of the new company and all the other relevant information related to the joint-venture (figure 3.9). The MUC-5 task was not therefore limited for the kind of template regarding the joint-venture which, in fact, was enough to explain all the important information related to it, but for the extremely restricted partition of the financial domain considered: the joint-venture domain.

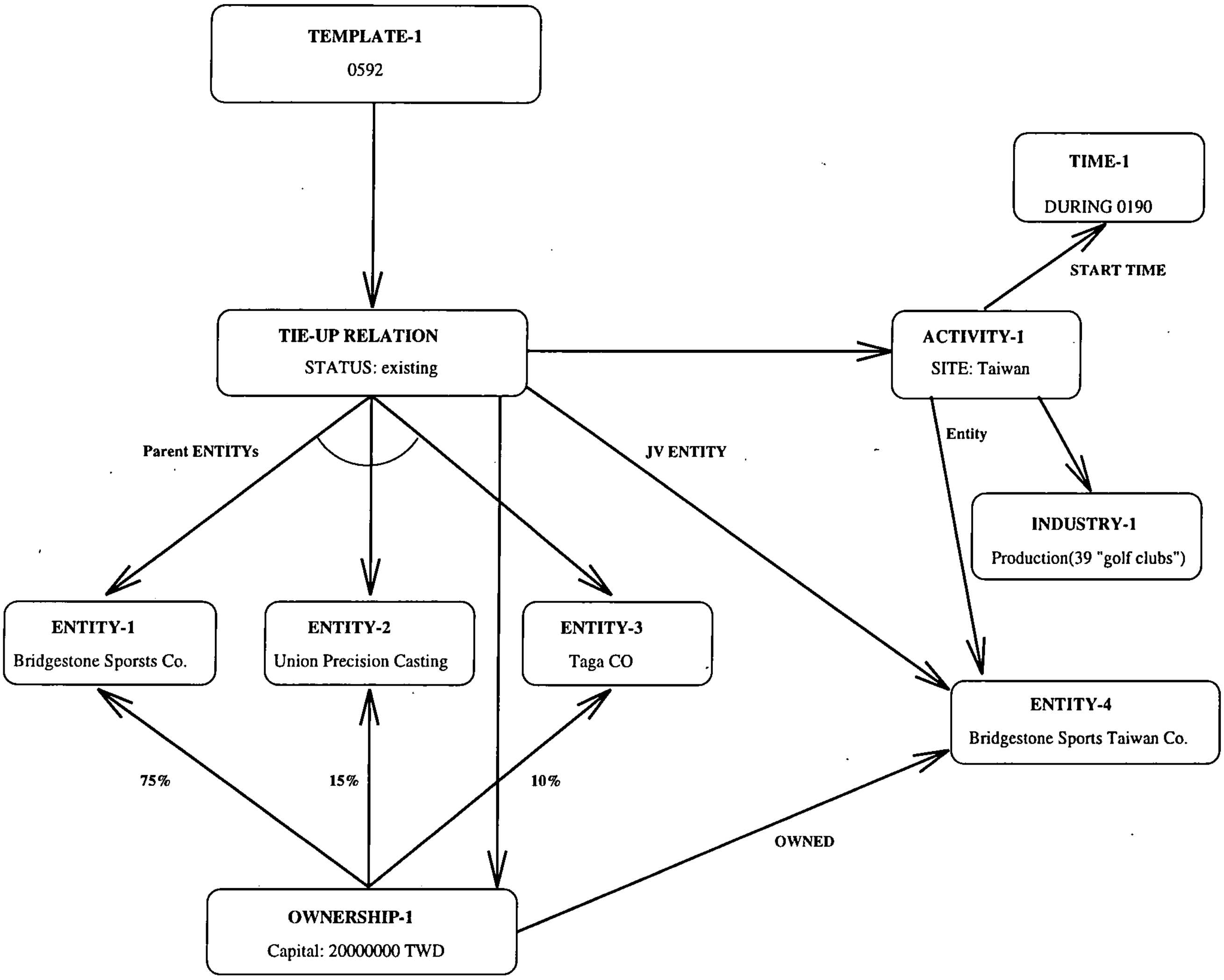

Figure 3.9: The MUC-5 joint-venture template schema.

The MUC-6 Tasks

Differently from MUC-5, the MUC-6 competition [DAR, 1995] consisted of four different tasks:

- named entity recognition: this task involved the recognition of entity names (for people and organizations), temporal expressions, place names and numeric values such as money and percentages. The task cannot be defined a proper information extraction task. However, it is usually a necessary step to produce a template or a summary of the original text;
- coreference: this task involved the identification of coreference relations. Different kind of links had to be indentified. For example, pronouns like "it" had to be linked to their corresponding entity such as the name of a company.
- information extraction "mini-MUC" (template filling): this task involved the extraction of information about a specified class of events and the filling of a template for each instance of such an event. The information extraction task included the identification of people, organizations and artifacts. An example of MUC-6 organisation template can be found in figure 3.10[1]
- scenario-templates. These templates had to be created following the guidelines contained in a "scenario" which was released only one month before the final evaluation of the systems. A sample training scenario was released together with the training data which, however, was substantially different from the final one. The late release of the scenario forced the systems to be more *flexible* and increase their *portability* towards new domains. Differently from the MUC-5 or the "mini-MUC" templates, the scenario templates included references to other templates. The final evaluation scenario-template, called *management scenario* concerned changes in the management of companies and is shown in figure 3.11.

[1] The text of the article can be found in figure 3.8.

```
(ORGANIZATION-0592-1} :=
ORG-NAME: “BRIDGESTONE SPORTS CO.”
ORG-ALIAS: “BRIDGESTONE SPORTS” “BRIDGESTON SPORTS”
ORG-DESCRIPTOR: “SPORTS GOODS MAKER”
ORG-TYPE: COMPANY
ORG-NATIONALITY: JAPAN
(ORGANIZATION-0592-2} :=
ORG-NAME: “UNION PRECISION CASTING CO.”
ORG-ALIAS: “UNION PRECISION CASTING”
ORG-DESCRIPTOR: “A LOCAL CONCERN” “CONCERN”
ORG-TYPE COMPANY
ORG-LOCALE “TAIWAN” COUNTRY
COMMENT: “uninformative descriptor”
(ORGANIZATION-0592-3} ‘:=
ORG-NAME: “TAGA CO.”
ORG-DESCRIPTOR: “TRADING HOUSE”
ORG-TYPE: COMPANY
ORG-NATIONALITY: JAPAN
(ORGANIZATION-0592-4} :=
ORG-NAME: “BRIDGESTONE SPORTS TAIWAN CO.”
ORG-TYPE: COMPANY
ORG-DESCRIPTOR: “A JOINT VENTURE”
ORG-LOCALE “KAOHSIUNG” CITY “KAOHSIUNG” PROVINCE
ORG-COUNTRY: TAIWAN
COMMENT: “A JOINT VENTURE is the most substantive descriptor”
(ARTIFACT-0592-1} :=
ART-DESCRIPTOR : “GOLF CLUBS”
COMMENT: “ART-TYPE not specifiable without rest of task”
```

Figure 3.10: A MUC-6 organisation template.

```
<TEMPLATE> : =
   DOC_NR:                   "NUMBER"
   CONTENT:                  <SUCESSION_EVENT>

<SUCCESSION_EVENT>:=
   SUCCESSION_ORG:           <ORGANIZATION>
   POST:                     "POSITION TITLE"|"no title"
   IN_AND_OUT:               <IN_AND_OUT>
   VACANCY_REASON:           {DEPART_WORKFORCE, REASSIGNMENT, NEW_POST_CREATED,
  OTH_UNK}

<IN_AND_OUT>:=
   IO_PERSON:                <PERSON>
   NEW_STATUS:               {IN, IN_ACTING, OUT, OUT_ACTING}
   ON_THE_JOB:               {YES, NO, UNCLEAR}
   OTHER_ORG:                <ORGANIZATION>
   REL_OTHER_ORG:            {SAME_ORG, RELATED_ORG, OUTSIDE_ORG}

<ORGANIZATION>:=
   ORG_NAME:                 "NAME"
   ORG_ALIAS:                "ALIAS"
   ORG_DESCRIPTOR:           "DESCRIPTOR"
   ORG_TYPE:                 {GOVERNMENT, COMPANY, OTHER}
   ORG_LOCALE:               LOCALE-STRING
   ORG_COUNTRY:              NORMALIZED-COUNTRY-or-REGION

<PERSON>:=
   PER_NAME:                 "NAME"
   PER_ALIAS:                "ALIAS"
   PER_TITLE:                "TITLE"
```

Figure 3.11: The MUC-6 management scenario template

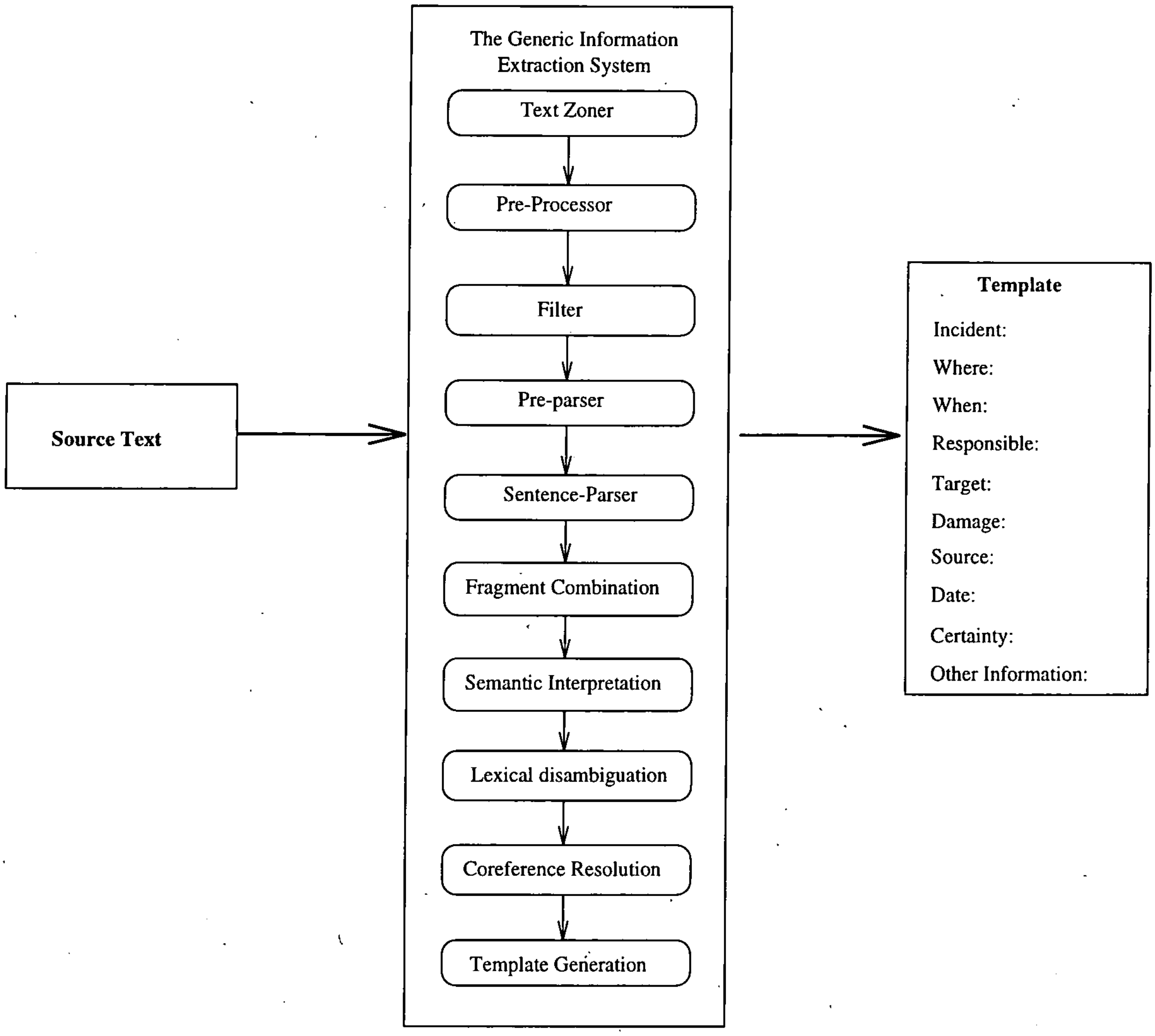

Figure 3.12: The generic information extraction system.

3.5.2 The main techniques employed

The systems developed for the MUC tasks show many improvements compared to the systems based on the frames / scripts approach.

Hobbs [Hobbs, 1993] described the architecture of a a generic MUC-5 information extraction system in order to better compare and analyse the systems that entered the competition. The structure such generic system is composed of ten modules performing different tasks (figure 3.12).

The first module of Hobb's generic information extraction system for the MUC task is the *text zoner*. The task of the text zoner is usually to parse the text, obtaining text segments. The way in which the different systems perform this operation can vary quite significantly.

The output of the *text zoner* module is then supplied to the *pre-processor* whose task is to identify the different sentences and to produce a sequence of lexical items from each sentence identified. A lexical item is a word together with its related information from the lexicon. Normally, the module makes the lexical attributes in the lexicon available to subsequent processing. Other tasks of this module can be the recognition of *multiwords*, the recognition and normalisation of times, dates, locations and so on and, finally, to handle unknown words, either by ignoring them or trying to guess them using morphological algorithms. Finally, spelling correction is often performed in this module.

The third module is typically the *filter*. This module tries to identify sentences that are likely to be irrelevant, normally using superficial techniques. The goal is to produce a smaller text that can be further quickly processed. The techniques used to perform this operation are generally of two kinds. The first simply looks for particular patterns of words that signal the relevance of the sentence. The second is to generate a statistical profile to weight the sentence which will be considered as relevant only if the measure exceeds a specific threshold.

The task of the *pre-parser* module is to recognise certain small-scale structures that are very common in sentences and, thus, simplify the subsequent task of the sentence parser. The depth of this analysis can vary between the different systems. However, usually the recognition of the "small-scale structures" is carried out by a finite-state module.

The output of the pre-parser is further processed by the *sentence parser* module which takes the sequence of lexical terms or phrases generated by the pre-parser and attempts to produce a parse tree for the entire sentence. Recently, many developers are abandoning full-sentence parsing in information extraction applications. In fact, some systems process only fragments of an entire sentence, usually because their grammar has a limited coverage. Moreover, the parsers used are often domain-dependent.

Since parsers used in information extraction tasks are not usually able to process a complete sentence, a *fragment combination* module is normally required. The task of this module is to combine the parse tree fragments generated by the sentence parser. The task can be performed on the original fragments or on a logical representation form in which they are translated.

The *semantic interpretation* module translates the parse tree or parse tree fragments into a semantic structure, logical form or event frame. Many system also perform lexical disambiguation at this level. Some systems do not perform full-sentence parsing and, therefore, group words into phrases and translate them into logical forms.

The *lexical disambiguation* task can be performed in various parts of the system and not necessarily after the semantic interpretation. The goal of lexical disambiguation is to translate *"a semantic structure with general or ambiguous predicates into a semantic structure with specific, unambiguous predicates"* [Hobbs, 1993].

The penultimate module is the *coreference resolution* which tries to convert a semantic structure in which there may be separate nodes for the same entity into a semantic structure in which nodes belonging to the same entity are merged together.

Finally, the *template generation* module tries to convert the semantic structures generated by the other modules into the final template. Information that passes the threshold of interest is placed into the template, while the others is dropped.

The MUC-5 Systems

The systems that competed in MUC-5 were 17. The best four system were: GE-CMU Shogun, BBN Plum, SRI Fastus and NUBA from the University of Manitoba.

About half of the MUC-5 systems were based on deep natural language techniques. The remaining systems emphasised either *finite-state* techniques[2] or *fragment parsing* techniques.

[2] A description of the most used finite-state techniques can be found in [Noble, 1988].

The four best systems in MUC-5 emphasised either *finite-state* analysis techniques or *fragment parsing.* Shogun and Fastus were mainly based on *finite-states* analysis techniques, while NUBA and Plum emphasised the use of *fragment parsing.* The authors of these systems justify the choice explaining that NLP techniques are still not able to fully understand the meaning of the text and that advanced statistical and finite-state analysis techniques can provide better results in terms of precision and recall. Another advantage of the systems based on finite-state analysis techniques is that they are particularly fast in the analysis of the source texts. Although the NLP systems' performance had been worse, the developers claim that these systems can be easily ported towards other domains.

In MUC-5 there were mainly seven systems based on deep natural language techniques: DBG, Alembic, Veniex, Proteus, Salomon, Link and Sussex. The best system of this group was the one developed by the University of Sussex, which was classified 7th. However, the recall of this system was about half of Shogun's, while precision was not too far from the average of the best 6 systems.

The best four systems will now be briefly analysed and compared according to the *generic information extraction system* described by Hobbs [Hobbs, 1993].

GE-CMU Shogun system

SHOGUN was the best performing system in the MUC-5 competition, with a recall of 57% and a precision of 49%. The System is based on the *finite-state approximation* approach [Jacobs *et al.*, 1993]. The assumption made by the authors is that the *finite-state patterns* will be usually able to process all the input that a general grammar would cover, but in a more tolerant way. Rules for compiling different knowledge sources into the finite state model are also included. The main difference between SHOGUN and other information extraction systems is in the use of the *finite-state analyser* instead of techniques based on parsing. SHOGUN differs from other approaches because it no longer has any purely syntactic component, and uses instead finite state rules. SHOGUN is composed of three main modules: the *pre-processing* module, the *finite-state sentence analysis* module and

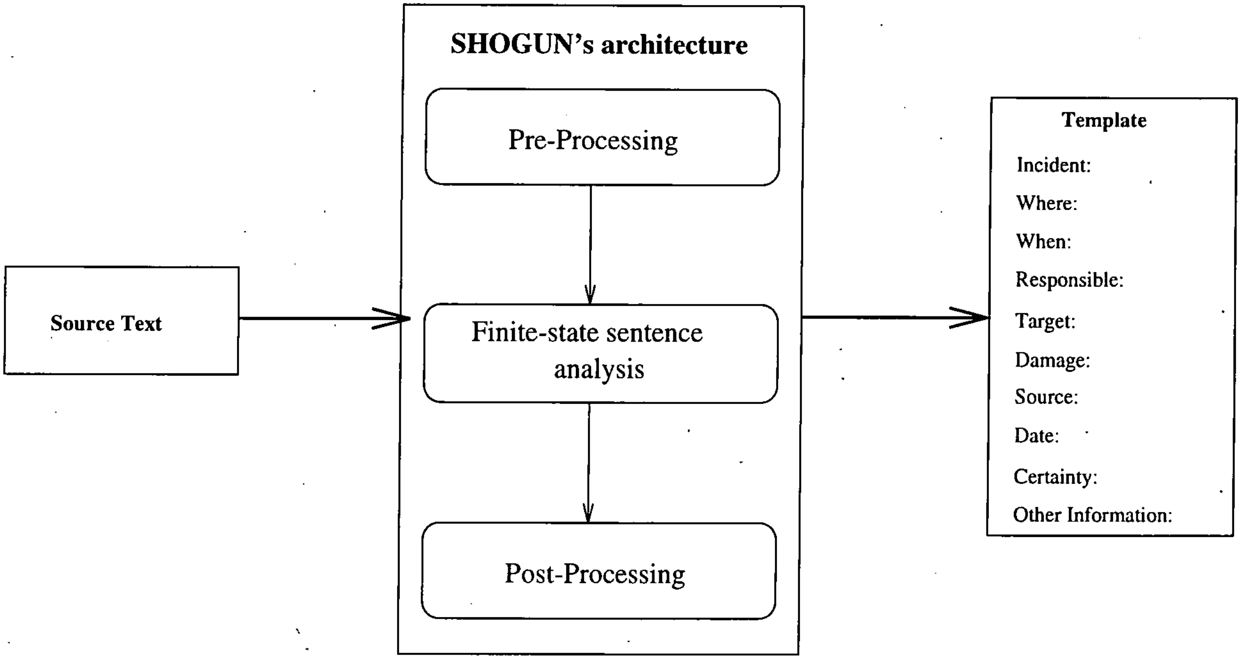

Figure 3.13: SHOGUN's architecture.

the *post-processing* module as shown in figure 3.13

The *pre-processing* module identifies names, dates, locations and other special phrases. Unknown company names are also identified and learned by the system. Referring to "The Generic Information Extraction System" [Hobbs, 1993], the functions that SHOGUN's *pre-processing* module performs are equivalent to the *text-zoner*, the *preprocessor*, the *filter* and the *preparser*.

The input of the *pre-processing* module is supplied to the second module: the *finite-state sentence analysis* (Linguistic analysis), which also represents the most original feature of SHOGUN. According to "The Generic Information Extraction System" [Hobbs, 1993] the tasks performed in this module are equivalent to the *parser* and the *lexical disambiguation* modules.

The *finite-state sentence analysis* module is based on the same knowledge base notation as the pre-processing module but it also includes syntactic and lexical information to perform sentence-level interpretation. The module annotates (through the pattern matcher) the text, as well as the pre-processing module. However, the annotation produced at this stage is very close to the information that will be used to produce the final template.

The third module is the *post-processor*. According to "The Generic Information Extraction System" [Hobbs, 1993], the tasks performed here are equivalent to the *fragment combiner*, the *semantic intepreter*, *discourse processing* and the *template generator*. The set of annotations produced by the *Finite-State Analysis* module is processed through semantic interpretation, top-down analysis using TRUMPET (*TRansportable Understanding Mechanism Package Expectation Tool* [Jacobs and Rauf, 1990]) and discourse processing. TRUMPET processes the pieces of semantic interpretation, trying to map them onto the final template. TRUMPET uses domain constraints, reference resolution and heuristics for merging and splitting the different information from multiple sentences and paragraphs. To produce the final template, SHOGUN must analyse and resolve all the references to objects and events. This task is performed by the *discourse processing* module.

BBN PLUM system

The second best system in MUC-5 was BBN's PLUM (Probabilistic Language Understanding Model) system [The PLUM System Group, 1993]. The recall of this system was 38 per cent, while the precision was 59 per cent. The main original features that can be found in PLUM are: *statistical language modelling*, *learning algorithms* and *partial understanding*.

The first feature is the use of statistical modelling to guide processing. The authors underline that the use of statistical methodologies allows the achievement of good results in terms of portability, robustness and trainability. The second feature is the use of learning algorithms. These kinds of algorithms were used in PLUM to obtain the knowledge bases used by PLUM's processing modules and to train the probabilistic models. The authors stress that the use of learning algorithms can improve the portability of the system towards other domains.

The third feature is partial understanding. PLUM, as well as other systems, does not fully understand the meaning of the text. It is designed to take advantage of the information that is fully understood, without failing when complete information is missing. Thus, full grammatical analysis and full semantic interpretation

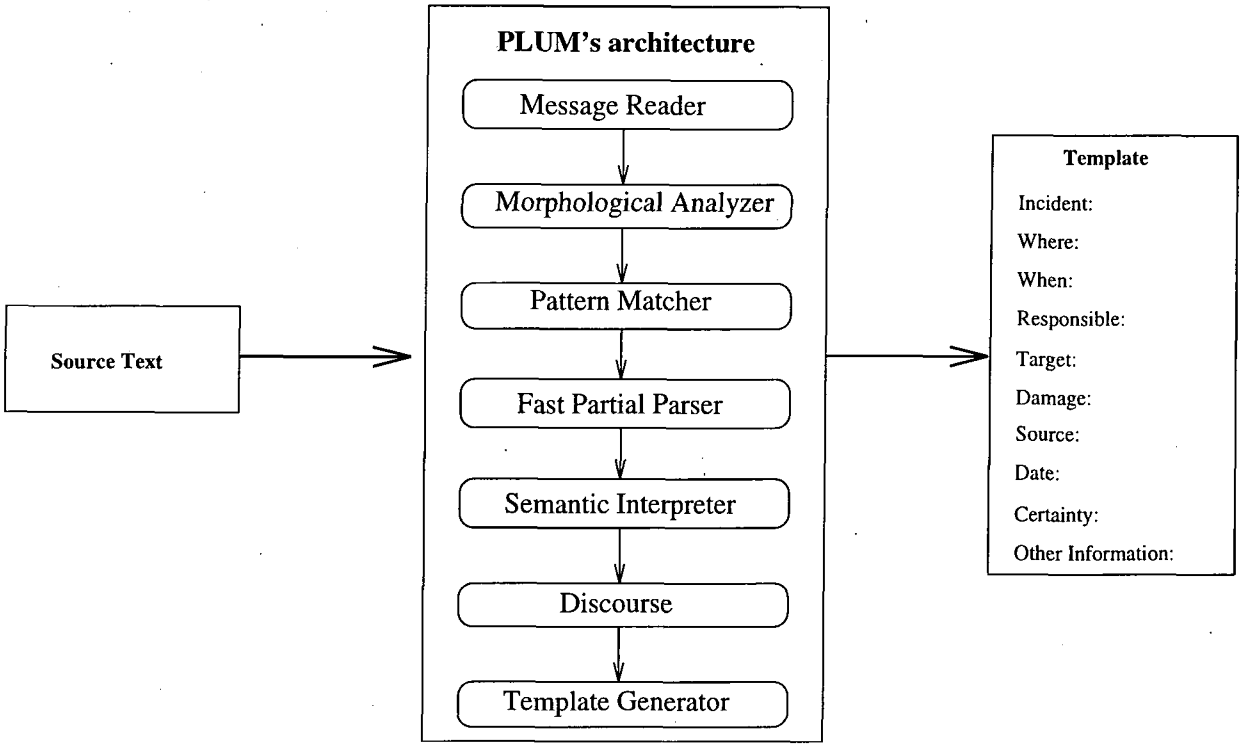

Figure 3.14: PLUM's architecture.

are not performed by the system.

PLUM's architecture is based on seven different modules: the message reader, the morphological analyser, the pattern matcher, the parser, the semantic interpreter, the discourse and the template generator, as shown in figure 3.14.

The first module, the message reader, reads the input given to the system (a file containing one or more messages) and determines the message boundaries, identifies the message header information and determines paragraph and sentence boundaries. The task performed by this modules is equivalent to the one performed by the *text zoner* module of "The Generic Information Extraction System" [Hobbs, 1993].

The next step taken by BBN is morphological analysis. Each word is classified and its parts-of-speech (e.g., proper noun, verb, adjective, etc.) identified. The assignment of part-of-speech to known words is done using a bi-gram probability model and frequency models, while probabilities based on word endings are used to assign part of speech to highly ambiguous or unknown words.

The output of the morphological analyser is supplied as input to the next module: the pattern matcher. The task performed by this module is to apply patterns to the input to identify relevant groups of words such as company names, organisation names and person names.

The fourth module is the parser. The parser produces one or more non-overlapping parse fragments. However, the parser will sometimes not be able to fully process a sentence and, therefore, will defer any decision regarding ambiguities. In case permanent ambiguities arise, the parser will leave the sentence starting to process a new one.

The output of the parser, a set of parse-fragments, is further processed by the semantic interpreter which consists of two sub-components: a *fragment interpreter* and a *pattern - based sentence interpreter.* The *fragment interpreter* applies semantic rules to each of the parse fragments produced by the parser. The semantic forms identified are entities, events, and states of affairs. Each of these forms can be classified into known, unknown and referential. Entities correspond to people, things, places and time intervals. Entities are related to each other through events and states of affairs. The *fragment interpreter* is able to prevent the generation of errors, improving robustness. Moreover, the *fragment interpreter* is not often able to process all the parse fragments. However, the system is designed to deal with *partial understanding.* The second sub-module of the *semantic interpreter* is the *sentence interpreter.* The task of this sub-module is to add extra "long-distance" relations between semantic entities in different fragments of the same sentence.

The sixth module is the discourse processing. The task performed by the module is to create a meaning for the whole message based on the meaning of the single sentences. However, the semantic information given as input may be insufficient to obtain a complete understanding. Thus, the discourse component must be able to infer "long-distance" relations that were not previously identified by the semantic interpreter and resolve any reference in the text. The output of the discourse processing module is a list of "Discourse Domain Objects (DDOs)". Creating the DDO, the discourse processing module fills the empty slots with the information supplied by the semantic interpreter and with those inferred. Moreover, the module tries to merge different DDOs to check whether a new DDO is only an update of an old one. The output of the discourse processing will not be further processed by "linguistic" modules, it will only be adjusted according to the required application-specific layout.

PLUM's last module is the template generator. The module produces the final output of the application. The generation of the output will be based on specific requirements of the application, instead of on linguistic processing.

SRI FASTUS system

The third classified system in MUC-5 was FASTUS (Finite State Automata-based Text Understanding System] from SRI [Appelt *et al.*, 1993]. Fastus belongs, as well as Shogun, to the group of systems that emphasise *finite-state* analysis. The authors underline that their system performs *information extraction* tasks, not *text understanding.* In the view of the authors [Appelt *et al.*, 1993], *information extraction* is a much simpler and more tractable task, characterised by specific information to be extracted from the text. *Text understanding*, in contrast, is extremely difficult, and presents a number of problems that have not been yet solved. Fastus was thus developed without using deep natural language techniques and is based on *finite-state* analysis techniques. There are seven main modules in the system: the *tokenizer*, *the preprocessor*, the *parser*, the *phrase combiner*, the *pattern recognizer*, the *merger* and the *post processor* as shown in figure 3.15.

The task performed by the first module, the *tokenizer*, is quite straightforward: it reads the ASCII input and performs the following tasks: groups characters into "words", computes value of numeric tokens, detects abbreviations, determines sentence boundaries and normalises corporate prefixes and suffixes.

The second module in Fastus is the *preprocessor.* The preprocessor accepts the tokens produced by the tokenizer and produces an output constituted by *lexical terms* which can be defined as a token or a sequence of tokens that have an entry in

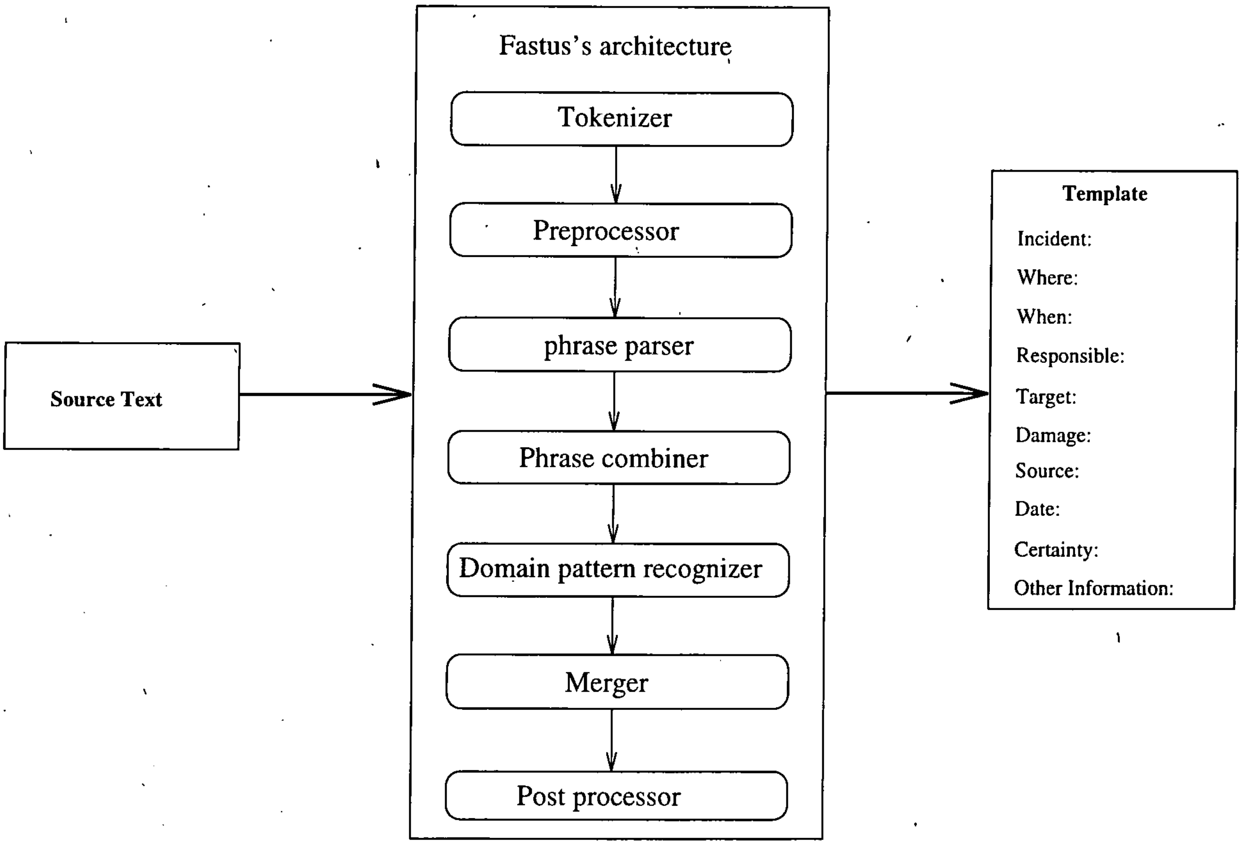

Figure 3.15: FASTUS's architecture

Fastus's lexicon. The lexical terms produced by the preprocessor are obtained from the following main steps: groups of words comprising multiword lexical items are collected together, company names, composed company names or possible company names are identified, names of people are identified.

The third module is the *phrase parser* which accepts the lexical item list produced by the *preprocessor* and produces a list of phrases. The parser identifies the head of each phrase and if the head corresponding to an information corresponds to an object in the template then an object of the appropriate type will be associated with the phrase.

The next step is performed by the *phrase combiner* and consists in combining the phrases supplied by the *phrase parser* to obtain, where possible, larger structures. This includes merging adjacent locations noun groups, conjunctions of company names and others.

The fifth module is the *pattern recognizer*, which constitutes the core of the Fastus system. The output of the pattern recognizer is represented by raw templates. The module tries to match the following main patterns:

- (company-name) ++ "joint venture" (company-name)
- (company-name) "joint venture" (company-name) ++
- (company-name) * "joint venture" (company-name) *

The first pattern will match sentences in which the parent companies (two or more) are listed before the company resulting from the joint venture. The second pattern matches the opposite case, in which the resulting company is indicated before the (two or more) parent companies. Finally, the last pattern will match the remaining case in which the name of the resulting company is not indicated. The domain pattern recognizer operates in different phases and patterns recognised are later discarded if they are not consistent with the new data.

Since the output of the *domain pattern recognizer* consists of many templates, the *merger* tries to merge them. Firstly all the templates found in a sentence are merged, subsequently, the remaining templates are merged with any templates from previous sentences.

Finally, the *postprocessor* normalises the raw templates and produces an output that meets the required characteristics by expanding dates, extracting company names from the original text and so on.

University of Manitoba NUBA system

The last MUC-5 system here considered is University of Manitoba's NUBA which was classified fourth after Shogun, Plum and Fastus. NUBA belongs to the group of systems, as well as Plum, that emphasise fragment parsing and is based on five main components: the *lexical analyser*, the *parser*, the *semantic analyser*, the *plan recognizer* and the *template filler* [Dekang, 1993]. However, not all the modules were implemented in the MUC-5 version of NUBA.

The task of the first module, the *lexical analyser*, is to recognise the sentence boundaries and to create a set of lexical items from the original sentence. A lexical term is composed by two elements, the *surface-string* which represents a set of subsequent words extracted from the sentence, and the associated *attribute-vector* which can be either *syntactic* or *semantic.* The recognition of sentence boundaries is performed by a LEX program. The next step taken by the *lexical analyser* is to map the words and phrases of each sentence into a set of *lexical items* using a lexicon. The lexicon used by NUBA consists of keys (which may be composed by more than one word) and a list of functions which can be either the meaning of the word or a list of phrases of which the word is the head word [Dekang, 1993]. The physical organisation of the lexicon is based on a hashing technique. Once the list of *lexical items* has been obtained, the next step is to apply a set of *lexical rules* on to them. The tasks performed by these rules are: corporate name recognition, irrelevant sentence filtering, negation handling, city name recognition, determination of location of entities.

The list of lexical items is supplied to the *parser*, which is the second of NUBA's modules. For MUC-5, however, the parser was not yet connected to the system.

NUBA's domain knowledge is stored in a semantic network used for the semantic interpretation defined by the authors as the process for *"finding the best explanation of how the content words in the sentence are related to one another in terms of semantic relationships in the network"* [Dekang, 1993]. In this view, each lexical item (with the associated attribute vector) corresponds to a node of the semantic network and the goal of the semantic interpretation is to find a generalised subtree in the network able to connect the lexical items. This subtree is called a scenario. More than one scenario can be found at the same time. The scenario preferred will be the one that is able to explain the largest number of lexical items with the minimum number of links. The best scenarios are later processed by the *discourse analysis* module, which tries, whenever possible, to unify different scenarios.

The last module implemented in the MUC-5 version of NUBA is the *template generator* module. The task of the module is quite straightforward: it processes the scenarios and produces one or more templates for each of them.

The MUC-6 Systems

The MUC-6 competition consisted of four main tasks: *named-entity*, *coreference* *template-element* and *scenario template*. Each group participating to the competition could choose to enter one or more tasks. We will here describe the best systems that entered the *template-element* and the *scenario templates* tasks. Although the *named-entity* and *coreference* tasks can be considered belonging to the field of information extraction, we will not discuss these results in detail. The reason is that these tasks do not imply the generation of *templates* of the original text, but they are extremely useful for supporting other kinds of information extraction applications[3]. The three best performing systems in the *template-element* task were:

Hasten	SRA
Alembic	MITRE
NLToolset	Sterling Software

while, for the *scenario template* task were:

NYU	New York University
PLUM	BBN Systems and Technologies
NameTag and Hasten	SRA

The Hasten System

The MUC-6 system developed by SRA uses the combination of two sub-systems: *NameTag*, a commercial software for recognising proper names and other key phrases in the source text and *Hasten*, a text extraction system. The key part

[3] For example, an information extraction application must be able to correctly identify names of companies, persons etc.

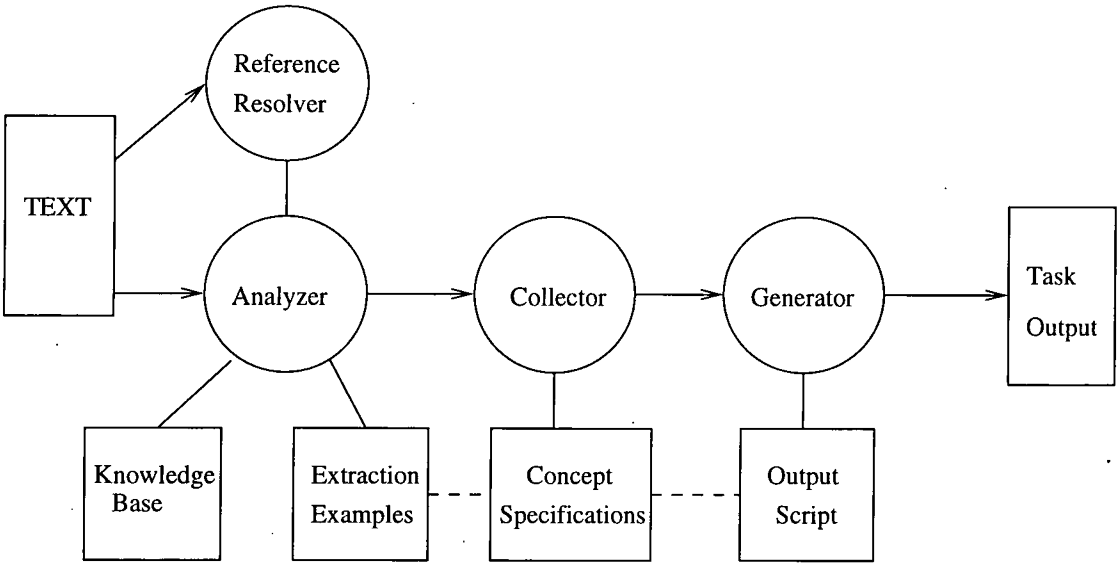

Figure 3.16: The Hasten MUC-6 system

of the combined system is *Hasten* which is built around the concept of *extraction examples*: the user provides source articles and annotates the text with what to extract and *Hasten* will use the information to analyze subsequent text [Krupka, 1995]. The system, in fact, computes the similarity between the annotations provided by the user from the sample articles and the key phrases currently being analyzed. The more examples the user enters, the more accurate should be *Hasten*'s extraction from the real data. The system consists of four main modules (see figure 3.16): the *analyzer*, the *reference resolver*, *the collector* and the *generator*.

The *analyzer* extracts semantic information from the text using a set of predefined examples regarding the text by matching the new information against the similar one available in the examples. The module makes use of the *reference resolver* to link the references to their referents [Krupka, 1995]. The subsequent step is the *collector* which collects and merges the semantic information extracted by the *analyzer* according to the *concept specifications*. Finally, the *generator* produces the output according to an arbitrary output format using an output script.

The system basically matches the new input against the collection of examples available in the system. This process is done taking in account specific parameters and produces a set of similarity values for each new sentence which are used to decide the relevance of the new input. If the similarity value overcomes a certain

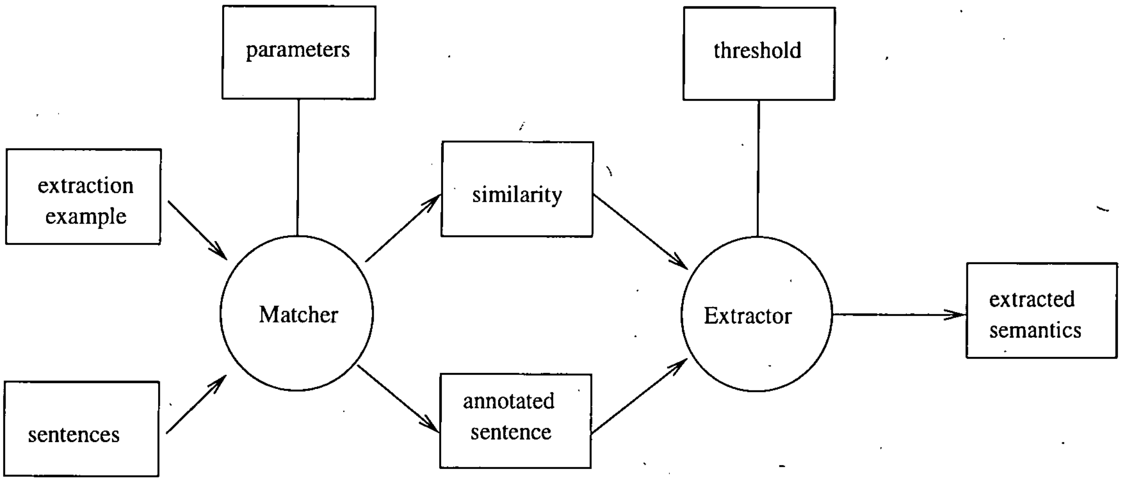

Figure 3.17: The extraction by example in the Hasten System.

threshold value, the information is considered to be relevant and will be further processed. In figure 3.17 the matching process for each of the input sentences of a source article is shown.

The *Hasten* System shows some improvements in respect to other systems based on *finite-state* analysis techniques and, in particular, those used in MUC-5. The system, in fact, allows the developers to specify the patterns by inputting directly the examples, rather than having to input the patterns using specific notations. The system, however, is still based on pure *finite-state* analysis techniques and, therefore, the user must supply each of the possible patterns encountered in the specific domain. Moving the system towards new domains can therefore be expensive, since new examples have to be provided. For example, the scenario template task required a total of 132 *egraphs* (examples) which compares with only few semantic structures of the LOLITA MUC-6 scenario template.

The Alembic System

The Alembic system [Aberdeen *et al.*, 1995] did not compete in the *scenario template* evaluation, but successfully participated in the *template element.* The system is based on an evolution of the *Alembic* system which competed in MUC-5 [Aberdeen *et al.*, 1993]. Not surprisingly, its evolution consisted in the introduction of simpler *syntactic* analysis modules, rather than performing deeper natural language

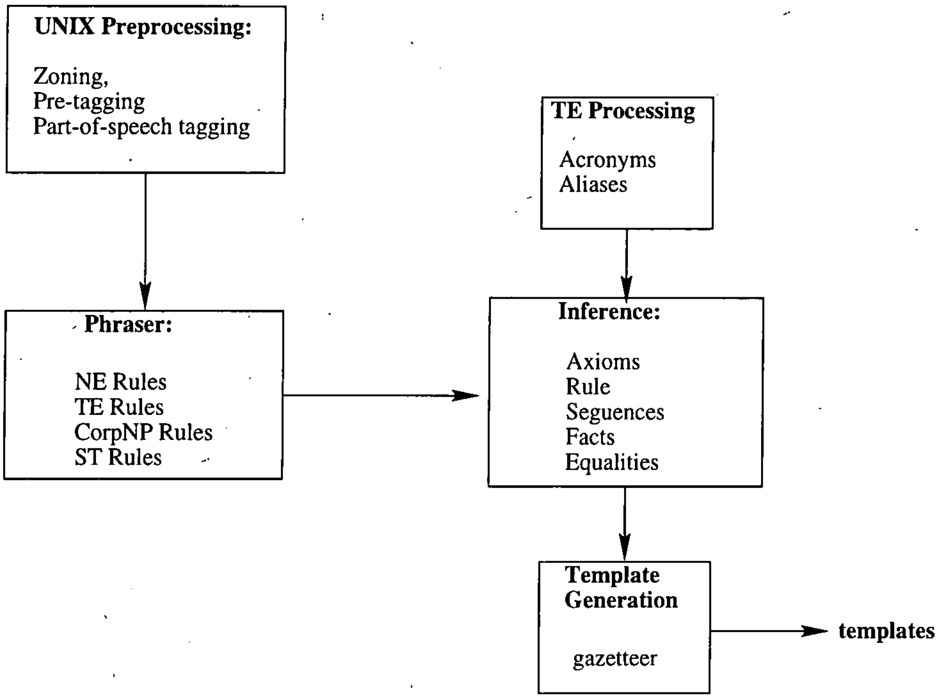

Figure 3.18: The Alembic System.

understanding as its predecessor. The new system comprises three main modules (figure 3.18): the *tagger*, the *phraser* and the *inference*. The *tagger* is implemented in C and is primarily responsible for part-of-speech tagging enriching the source text using SGML tags. The algorithm is based on an improvement of the work regarding rule sequences by Brill [Brill, 1994].

The central part of the system is the *phraser* which performs the syntactic analysis in the system. It is basically similar to many other systems based on *finite-state* analysis techniques with the difference of being driven by rule sequences. The rule sequences are applied against all the phrases of the current sentence. The phraser step is finished when all rules have been applied.

The third main module of the system is the *inference* which performs phrase interpretation, the so-called *equality reasoning* and infers new facts from the source text using *forward* inference rules.

Differently from its predecessor, the new alembic system is more similar to other conventional systems based on *finite-state* analysis techniques. Therefore, the system does not perform semantic analysis of the source text nor analysis of words meanings.

The NLToolset System

The NLToolset System, developed by Sterling Software in collaboration with Lockheed-Martin was originally designed for working in the counter-narcotics domain and was converted for the MUC-6 competition [Lee, 1995].

The system is based on *finite-state* analysis techniques and does not perform any semantic analysis of the input text. The core of the system is a pattern-matcher which is repeatedly used in the processing of the source text. The pattern-matcher is employed in the *reduction* modules. The text is repeatedly matched to each of the system's patterns obtaining the relevant information. The *extraction* module subsequently uses the results of the *reduction* modules to generate *expectations* which are used to fill the templates. Finally, a *merging* modules is used to merge expectations which refer to the same type (person, organization, etc.). The way in which template elements and scenario templates are generated by the system is slightly different (figure 3.19) but is conceptually equivalent.

The NYU System

The System developed by the New York University [Grishman, 1995a] is a system based on *finite-state* analysis techniques. Differently from the systems developed at New York University for the previous MUC competitions [Grishman and Sterling, 1993], the MUC-6 system does not perform any semantic analysis of the input text. The system comprises seven main stages: tokenization and dictionary look-up, four stages of pattern-matching, reference resolution and output generation.

The tokenization and dictionary look-up module is responsible for the identification of each word of the input text and makes use of various databases: a country database, a company dictionary, a government agency dictionary, a dictionary of common first names and a dictionary of scenario-specific terms specific for the MUC-6 competition. Once the words have been identified in the dictionary, a

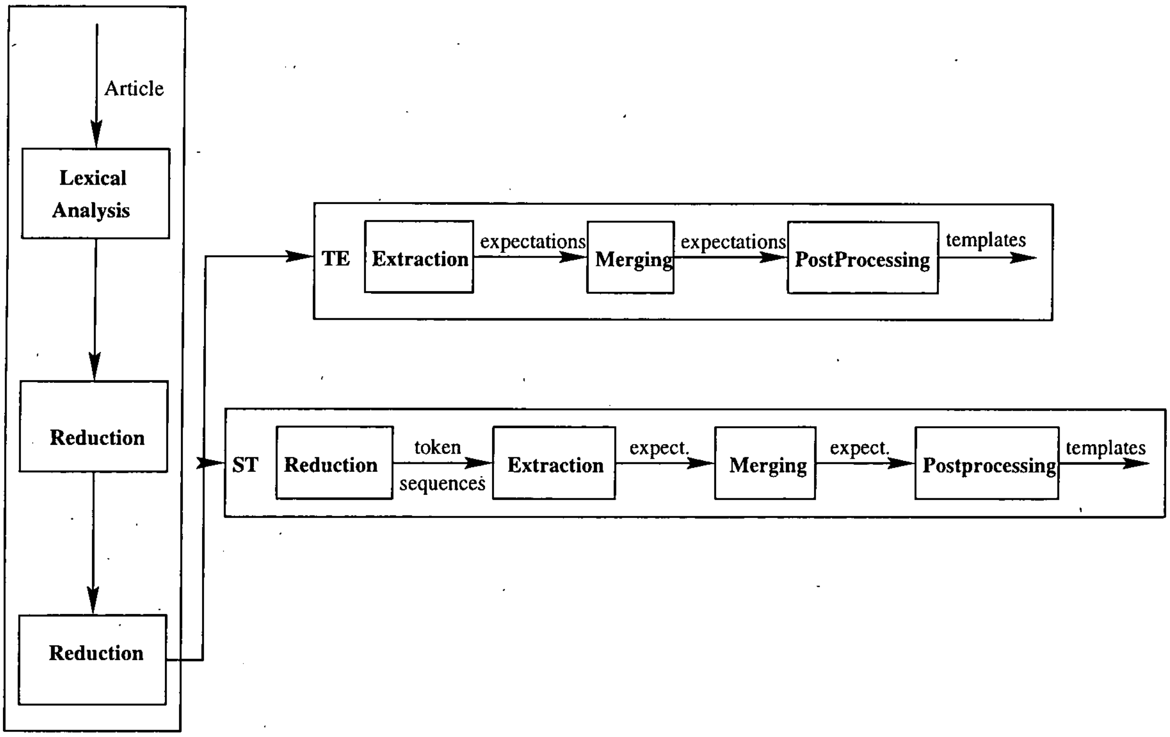

Figure 3.19: The NLToolset System.

post tagger is used to identify the most likely part of speech for each word.

The output of the tokenization module is later processed by the pattern-matching module which recognises four different forms of patterns: names, nouns, verbs and semantic patterns. Semantic patterns are scenario specific patterns, such as:

- start-job(person,position)
- add-job(person,position).

The output of the pattern-matching module is later analysed by the reference resolution module which unifies and integrates them in case they refer to the same object or event.

Finally, the system generates the output according to the output required for the specific task.

The NYU system is therefore a pattern-matching system without any form of "understanding" of the input text. The system was the best system in the scenario template competition.

The PLUM System

The MUC-6 plum system [The PLUM System Group, 1995] is directly derived from the PLUM system that participated in the MUC-5 competition [The PLUM System Group, 1993] (see section 3.5.2). The system comprises seven modules. Their functionalities correspond to those described in section 3.5.2 and are shown in figure 3.14.

Although the system presents several improvements and has been adapted for producing the MUC-6 scenario templates, it is still based on the same key points as its direct predecessor: the use of statistical models, partial understanding and learning algorithms.

The LOLITA system

The LOLITA system successfully participated in the MUC-6 competition. The system and the information extraction module will be analysed in detail in chapter 4 and in chapter 5 respectively.

3.5.3 The evaluation of the MUC results

The evaluation measures used in the MUC competition are rather important, because they have been used for the evaluation of a relevant number of systems using a relevant amount of data.

The MUC-5 measures were based on several basic scoring categories. These data were computed matching the templates produced by the systems to hand-made templates by a scoring program and were:

correct: if the key (in the hand-made template) matched exactly the response of the system;

partial: if the key matched the response, although not exactly;

incorrect: if no match was identified between the key and the response;

missing: if the key had a fill and the response didn't;

spurious: if the response had a fill which was not present in the key;

The primary performance measure in MUC-5 was the *error per response fill* which was calculated as the number of wrong responses divided by the total and it was chosen because it gives an indication to the developers about the source of errors. Three secondary measures were used: undergeneration, overgeneration and substitution. These three measures represent different elements of the overall error and can be used to view particular aspects of the error. The primary measure (error per response fill) and secondary measures (undergeneration, overgeneration and substitution) were mainly designed to satisfy the developers' need. However, another different measure was used, mainly because of its independence from the system, the *richness-normalised error*, which was designed for measuring errors related to the amount of information to be extracted from the texts [Chinchor, 1993]. This error was calculated by dividing the number of errors per word by the number of key fills per word. The error can be subdivided in minimum and maximum error.

MUC-5 used as unofficial secondary metrics, mainly to assure continuity with the precedent editions, the precision (accuracy) and recall (completeness) measures, which were slightly different from the one normally used in information retrieval. Recall was defined as the percentage of possible correct answers, while precision was defined as the percentage of actual correct answers given [Chinchor, 1993]. The last measure used in MUC-5 was the F-measure which represents a way to combine the precision and recall measures into a unique value and was first introduced by van Rijsbergen [Rijsbergen, 1979]. The F-measure, as combination of precision and recall, gives a values that falls between them. The β parameter in the F-measure represents the relative importance given to recall over precision and in the case recall and precision are of equal weight, β assumes value 1.0. The F-measure presents a higher value if precision and recall are more at the center of the recall-precision graph than if they are at the extremes of it. For example, if a system has precision and recall both of 50 per cent, the F-measure will be higher than a system that has recall of 20 per cent and precision of 80 per cent. This is also because the aim of the formula is to direct developers towards an improvement of both recall and precision. The measures explained above are shown in figure 3.20.

$$
\begin{aligned}
error\ per\ response\ fill &= \frac{incorrect{+}partial.0.5{+}missing{+}spurious}{correct{+}partial{+}incorrect{+}missing{+}spurious} \\
undergeneration &= \frac{missing}{correct{+}partial{+}incorrect{+}missing} \\
overgeneration &= \frac{spurious}{correct{+}partial{+}incorrect{+}spurious} \\
substitution &= \frac{incorrect{+}partial/2}{correct{+}partial{+}incorrect} \\
F\ measure &= \frac{(\beta^2{+}1.0).P.R}{(\beta^2.P){+}R} \\
recall &= \frac{correct + (partial\ .\ 0.5)}{possible} \\
precision &= \frac{correct + (partial\ .\ 0.5)}{number\ of\ actual\ answers}
\end{aligned}
$$

Figure 3.20: The MUC-5 evaluation measures

Within the MUC-5 evaluation the performance of humans against machines were compared [Sundheim, 1993]. The result of the comparison is that machine performances are still far below the humans in terms of precision and recall. In fact, four human analysts were able to extract up to 79 per cent of the information (recall) and, of all the information extracted, 82 per cent was relevant (precision). The best three performing systems in MUC-5 were able to extract in average 53 per cent of the relevant information and about 57 per cent of the information extraction was relevant. On the other side, machines notably outperformed humans in terms of speed. In fact, while a human took an average of about 37 minutes to produce a complete template, the three best systems were able to produce the template in an average of 143 seconds.

The evaluation of the MUC-6 systems was based on the same measures as MUC-5. The scoring of the named-entity task was slightly different from the template-element (mini-MUC) and the similar (from a scoring point of view), scenario templates [Chinchor and Dungca, 1995]. The evaluation indexes were based on the results of a scoring program which was available to the participants for the training of the systems. Although the scorer, as the MUC-5 scorer, evaluated also partial fills, the assumption was that the template should have been filled with exact copies of the source text.

The MUC competitions provided a general evaluation framework for a relevant number of systems performing information extraction on a high number of source texts, and contributed to the improvement of the research in the field of information extraction. However, one main criticisms can be made. It is in fact questionable whether the MUC tasks are useful for the end-user of an information extraction system. Tasks such as the MUC-6 named-entity recognition or coreference would probably not be directly useful for a potential user of an information extraction system, although they could be incorporated in more general systems. Callaghan [Callaghan, forthcoming 1997] argues that there is some artificiality in the MUC tasks and suggests that the scoring should be weighted according to how useful the answers are.

3.6 Information extraction in the financial domain

The goal of information extraction applied to finance is, as within other domains, to extract relevant information from a text producing an output consisting of a template of the original text. In regards to the kind of "lexicon" used in financial news articles, the assumption that financial news are based on specialistic knowledge and, thus, that they represent a restricted domain is not quite true. For example, if we consider national and international politics as belonging to or influential on the financial domain, it becomes clear that the dimensions of the lexicon in use will increase. Financial news can also include economic and statistical forecasts, as well as announcements made by political and economical authorities etc.

A distinction that is possible to make between systems in the financial domain is in the kind of articles used.

A first of systems can operate on news from on-line services like Dow Jones and

Bankers Trust Earnings -4-: Latest 4Q Trading Revs 49M dlrs Date: Jan 19, 1995 Time: 9:34 am

Bankers Trust New York Corp. (BT) said its 1994 earnings were affected by persistently difficult market conditions, which hurt trading revenue.

The company said trading revenue was 49 dlrs million in the latest fourth quarter, down from 449 dlrs million in the year-ago quarter, while trading-related net interest revenue fell to 50 dlrs million in the latest fourth quarter from 187 dlrs million a year earlier.

Bankers Trust said many of its proprietary trading businesses, principally fixed income instruments, recorded lower revenues during the latest fourth quarter as market conditions remained generally unsettled. The company said trading results also declined significantly in the emerging markets of Asia and Latin America, and said the volume of traditional risk management products slowed.

The company said revenue increased from its client-related businesses that provide financing, advisory and transaction processing services.

Bankers Trust said it reclassified 423 dlrs million of leveraged derivative contracts as receivables in the loan account and placed them on a cash basis during the latest fourth quarter. Of the amount, the company said 72 dlrs million was subsequently charged off to the allowance for credit losses. About half of the remainder related to transactions with Procter and Gamble Co. (PG).

With the transfers and charge-offs, the company said it has taken action on the leveraged derivative transactions that likely will not perform according to the contract and has charged off the balances deemed to be uncollectible.

Bankers Trust said net charge-offs for the latest fourth quarter were 85 dlrs million, compared with 184 dlrs million a year ago.

Company:
BANKERS TRUST NEW YORK CORP.
Industry:
MAJOR MONEY-CENTER BANKS
BANKS (UMBRELLA CODE)
Subject:
BANKING WIRE
EARNINGS
INTERNATIONAL NEWS WIRE
INTERNATIONAL ECONOMIC NEWS AND ANALYSIS
INTERNATIONAL EQUITY REPORT
WORLD EQUITY INDEX
WORLD EQUITIES REPORT
Market:
FINANCIAL
Geographic:
NORTH AMERICA
NEW YORK
UNITED STATES - REGIONAL NEWS CODE FOR WIRES

Figure 3.21: A typical article from an on-line news provider

Bloomberg. These news are similar to news of agency press in the sense that they mainly report facts. The text will therefore rarely include deep economical analysis on the facts or interpretation by experts in the particular field. The main aim of information extraction system in this case will be to extract basic information in real-time. In fact, people interested in this source of news do not usually have a considerable amount of time to read the news and, therefore, are quite interested in having a short summary or template instead of the whole article. Many on-line services already provide a short template at the end of the article, for example *Dow Vision*[4] (figure 3.21). However, the template is usually very limited in the kind of information extracted and it is mainly produced using simple key-word or statistical techniques. The templates produced are, often, unable to correctly represent the whole content of the original article which cannot therefore be substituted with the templates.

Another kind of systems considers a different type of articles. These articles are usually newspapers or magazine articles, such as *The Financial Times*, *The Economist*, *The Wall Street Journal* and so on. This second group of articles differs from the agency press articles because they usually include analysis, forecasts and interpretations and are not only simple "reports of facts". In figure 3.22 the same news reported from a newspaper (*The Financial Times*) and from an on-line news provider (*Dow Jones*) is shown.

Very few financial information extraction systems have been realised in the past. Moreover, very few information extraction systems that were able to work in the financial domain have been commercially successful. One of these was ATRANS, which has been described earlier in this work. ATRANS, however, was dealing with an extremely restricted subset of the financial domain (telexes regarding money transfers). Major vendors of on-line information, such as *Bloomberg* and *Dow-Jones* have developed tools able to recognise simple patterns and information (for example I.B.M. = IBM) [Church and Rau, 1995]. Other systems are JASPER [Cowie and Lehnert, 1996], which was designed to extract information from reports

[4] http://dowvision.wais.net

A typical article from "The Financial Times"
Disney to buy Capital Cities
Tuesday August 1 1995
By Tony Jackson in New York

Walt Disney is to pay 19.1bn dollars for Capital Cities/ABC, owner of the ABC television network, creating the world's largest entertainment company. The deal is the second biggest takeover after that of RJR Nabisco by Kohlberg Kravis Roberts for about 25bn dollars in 1988.

In an agreed deal, Disney will pay one share plus 65 dollars for each Capital Cities/ABC share, valuing the latter at 124 dollars, at yesterday's prices. The combined company will be called simply Walt Disney, and Mr Thomas Murphy, chairman of Capital Cities/ABC, will join the Disney board.

Mr Michael Eisner, Disney's chairman, claimed the synergies between the two companies were tremendous". He said: "Disney's intellectual property will appear on ABC's networks, and Disney's distribution systems will syndicate ABC's programmes."

Mr Warren Buffett, the billionaire portfolio investor whose company Berkshire Hathaway owns 12.9 per cent of Capital Cities/ABC, described the deal as "the marriage of the number one content company in the world with the number one distribution company".

Mr Buffett, who was instrumental in the merger of Capital Cities and ABC in 1986, bought shares in the company at the time for 17.25 dollars each, less than one seventh of the bid price. His holding is now worth 2.5bn dollars. He said yesterday: "I do not have blanket enthusiasm for all mergers. But this deal makes more sense than any deal I've ever seen, with the possible exception of Capital Cities and ABC."

The ABC television network rivals NBC for the top position among US networks, with about a 17 per cent market share. Capital Cities/ABC, which had revenues last year of 6.4bn dollars , also owns eight television stations, America's largest network of radio stations and an 80 per cent share of the leading cable television channel ESPN. It also publishes newspapers, books and magazines.

Walt Disney had been rumoured for some time to be interested in buying a TV network, but was thought to want CBS or NBC, either of which would cost 5bn dollars or less. It is much larger than Capital Cities/ABC, with a market value of 31bn dollars.

Mr Eisner, who worked as a programming executive with ABC throughout the 1970s, said discussions with ABC had gone on intermittently since he joined Disney in 1984. However, the deal had eventually been put together very quickly, he said. He had encountered Mr Buffett by chance while in Sun Valley, Idaho. When he raised the possibility of a merger, Mr Buffett took him to see Mr Murphy. "Eight days later, here we are," Mr Eisner said.

Mr Eisner said the takeover should not affect the joint venture struck last November between ABC and DreamWorks, the partnership between the Hollywood trio of Mr Steven Spielberg, Mr Jeffrey Katzenberg and Mr David Geffen. The venture aims to provide television programmes. He said: "We want the best shows on this network that it's possible to get. If DreamWorks can come up [with them], I'm thrilled." Mr Katzenberg resigned as Disney's production chief last summer after a highly public row over promotion.

The deal allows Capital Cities/ ABC shareholders to take all their payment in either cash or stock, subject to availability. Mr Buffett said "the odds are extremely high that we will have a very large amount of Disney stock".

The same topic from an on-line news service (Dow-Vision)
Disney, Capital Cities -2-: Details On Merger DIS CCB
Source: Dow Jones News Service via DowVision Date: Jul 31, 1995 Time: 8:04 am

BURBANK, Calif. -DJ- Walt Disney Co. (DIS) and Capital Cities ABC Inc. (CCB) agreed to merge in a transaction valued at about 19 billion dollars at current share prices.

In a joint press release, the companies said Capital Cities/ABC shareholders will have the right to receive one Disney common share and 65 dollars in cash for each Cap Cities common share. The transaction has been approved by both companies' boards, the companies said.

Figure 3.22: The difference between articles from newspapers and on-line agencies.

on corporate earnings, a very limited financial domain, and SCISOR, which considered the extraction of information regarding takeovers and mergers [Rau, 1987] from source text, but full templates were not produced. Another group of information extraction systems in the financial domain, although not commercialised, were the systems developed for the MUC-5 competition [DAR, 1993], earlier described in this work. However, they were only able to create templates regarding Joint Ventures, thus, working in an extremely restricted subset of the financial domain.

3.6.1 Automatic extraction of templates from source texts

The templates could be potentially defined by automatic programs, rather than by the user or during the design of the system.

Collier [Collier, 1994] [Collier, 1996] analyses the possibility of extracting automatic structures (templates) from source texts. The system is able to process a number of input texts and recognise the significant similarities between them, identifying a possible template for the extraction of the most important information. The main characteristic of the system is that a relevant number of articles must be pre-processed in order to identify a sensible template structure. Moreover, although the system could automatically identify the relevant information in the source articles, the information judged relevant by the system could be not satisfactory for the user of the system who might be interested to skip a particular kind of source articles.

3.7 User-definable template interfaces

Most of information extraction systems developed in the past have been designed and tested within government agencies and the scientific community and very few real applications have been commercially successful. The emphasis has been on the improvement of the performance of the systems in terms of precision and recall (e.g. the MUC evaluation metrics). However, little progress has been done in making the systems user-friendly.

One of the main criticism that can be made to many of the existing information extraction systems is that the users can hardly configure the systems to produce other kinds of results (templates) which differ from those already available in the system. In other words, the *templates* are often *coded* within the system and the user cannot modify the existing templates or add new ones without having to intervene directly on the system's code. If for scientific competitions such as the MUC conferences this can be acceptable, for real applications such as a financial information extraction system, this problem is extremely relevant.

The lack of flexibility of the current information systems has been also identified in the TIPSTER phase II project document [Grishman, 1995b], in which it is hoped that future systems will allow the definition of custom templates by the end-user. The document also defines specific standard objects and classes for the development of standardized components within a customizable information extraction system. The TIPSTER phase II document defines three different classes of objects for a customizable information extraction system:

- **ExtractionNeed**. This class should contain the input definition of the user, consisting of a formal specification (e.g. the template and slot names) and a narrative description describing the slot fill rules (e.g. the MUC-5 slot fill rules). This should be then translated by the system obtaining the CustomisedExtractionSystem.
- **CustomizedExtractionSystem**. This class should contain the system-specific procedures for extracting the user-defined templates from the source texts. These procedures should be created employing specific operations available in *CustomisedExtractionSystem.*
- **TemplateObjectLibrary**. This class should contain the system-specific rules for general concepts which might be used in the user's definitions of the templates such as *person*, *company* etc.

The architecture proposed in the TIPSTER phase II document is particularly interesting, but presents two main limitation:

- the document defines the class of objects *ExtractionNeed*, but does not exactly determine which kind of definitions should be entered by the user. The authors propose the MUC-5 slot "fill-rules", but it is not clear how the ambiguities of such definitions would be resolved or how the user should refer to information already defined in other slots. Processing a simple MUC-6 fill rule such as the following one for the ORG_NAME slot:

  ```
  The proper name of the organization, including any corporate
  designators.
  ```

  would present several ambiguities, which require deep understanding of the source text;

- it is unclear *how* the system would be able to *translate* the input definitions into specific templates in the class *CustomisedExtractionSystem.* It is proposed that this class should contain the operation "*Customise (ExtractionNeed): CustomisedExtractionSystem*" which should be an interactive process with the user, but it is unclear *how* this could be implemented;

- it is unclear how the user could refer in the fill-rule definitions to the class *TemplateObjectLibrary*, which should contain objects commonly used.

However, the architecture proposed in the TIPSTER phase II project is a first step towards the development of customizable information extraction systems.

Tabula Rasa [Ogden *et al.*, 1994] is an early prototype for supporting the analyst's process in defining new templates. The user is assisted in the definition of new templates using a menu-based approach. The *Hasten* system, which successfully participated in the MUC-6 competition [Krupka, 1995], is an example of partially-customizable information extraction system. The user-definable interface is based on *example-patterns* corresponding to relevant fragments of source texts which can be entered by the user and will be used for producing the templates.

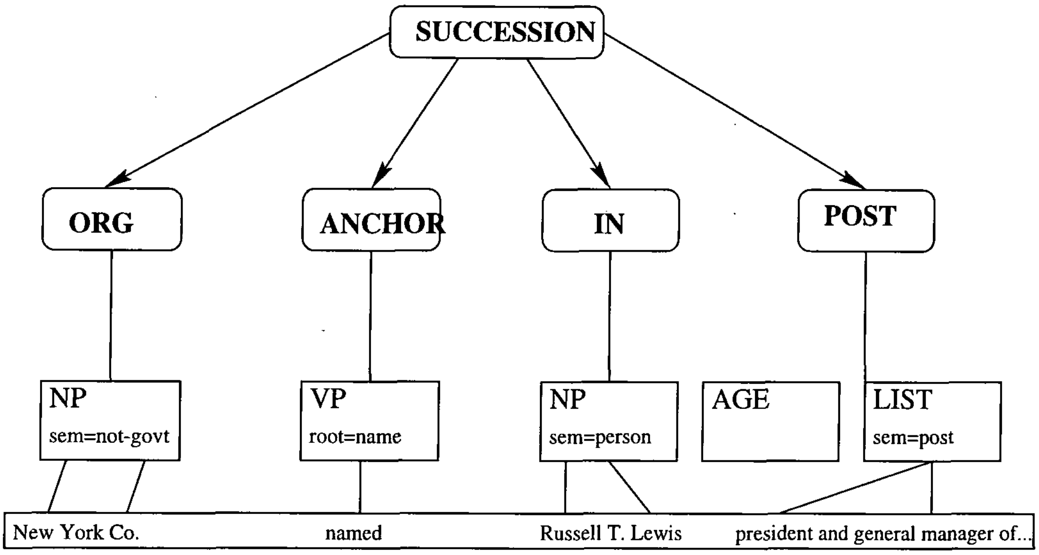

Figure 3.23: A Hasten pattern (egraph).

Although the interface presents the advantage of allowing the user's definition of the slots, few problems arise in the definition of a totally new template:

- the **template definition** is still coded in the system. The user is allowed to enter slot definitions for the templates already coded in the system, but the definition of new templates must be done by modifying the system's code.
- the user is required to enter a considerable amount of example patterns for the definition of a specific slot. The problem is mainly caused by the fact that the system is based on *finite-state* analysis which intrinsicly require a considerable amount of patterns. In figure 3.23 an example of Hasten pattern (egraph) for the MUC-6 management succession template is shown. The pattern is however unable to capture any other configuration of the same event and, in fact, the total number of egraphs which have been used for the MUC-6 management succession template is 132 [Krupka, 1995].

Differently from information extraction, in the field of *information retrieval* the issue of making the system as user-friendly as possible has already been tackled. For example, the systems that participated in the TREC competitions were able to automatically construct a user query from paragraphs in natural language.

3.8 Conclusions

In this chapter we introduced the field of information retrieval and information extraction and the most relevant techniques, approaches and systems. We particularly focused on the Message Understanding Conferences which provided a common evaluation framework for information extraction which allows to compare different systems performing the same tasks. Such evaluations have shown that the best MUC systems are based on *finite-state* analysis techniques or statistical and probabilistic techniques.

Chapter 4

The LOLITA System

4.1 Introduction

The LOLITA (Large-scale, Object-based, Linguistic Interactor, Translator and Analyser) System has been under development at the University of Durham since 1986. Following the principles of Natural Language Engineering (see section 1.3), the system has been designed as a *general-purpose* base. This means that different kinds of applications can be easily built around the original core which provides two main facilities: analysis, which converts text to a logical representation of its meaning, and generation, which expresses information represented in this logical form as text [Morgan *et al.*, 1995]. Currently, around 20 researchers work on various aspects of the system, as part of the Durham's Laboratory for Natural Language Engineering. Various kinds of applications have been built around the original system's core and include: dialogue, query, translation, database front-end, information extraction, telephone enquiry system, Chinese tutoring. One of the advantage of using a common general purpose NL core is that any improvement in the core are immediately reflected in the applications.

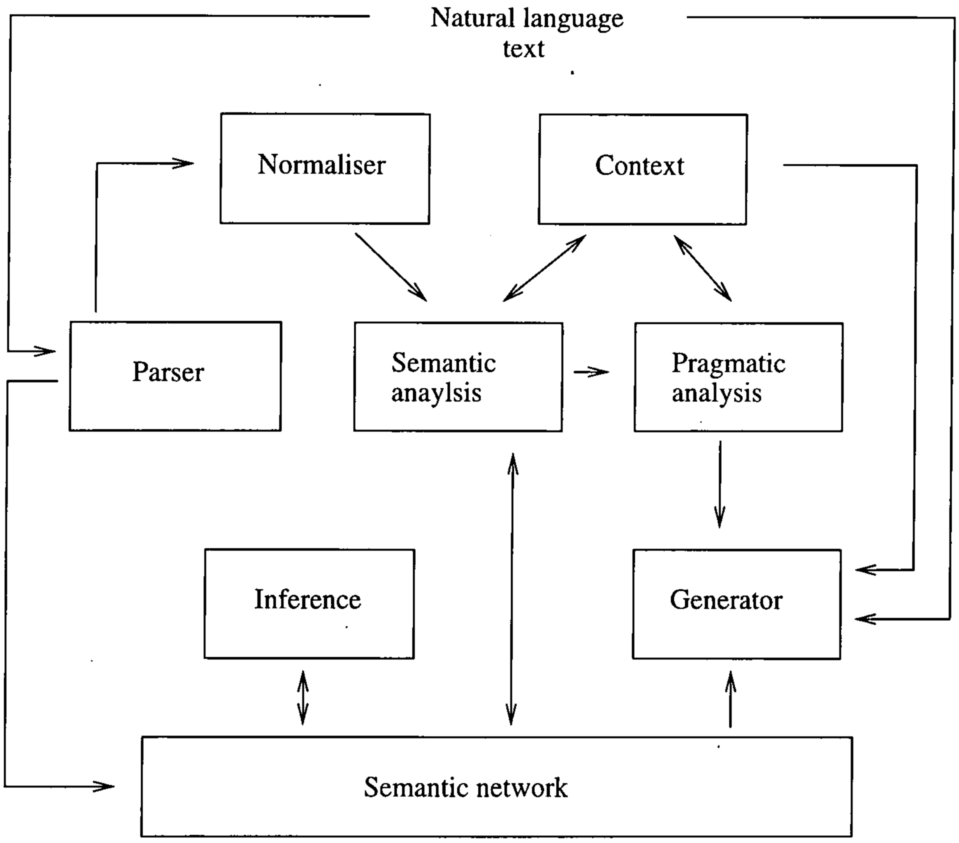

Figure 4.1: The LOLITA core

4.2 Architecture of the system

This section outlines the architecture of the LOLITA System and its important components which are shown in figure 4.1. The input is processed through a sequence of hierarchical modules and the analysis is stored in the semantic network which represents the *knowledge-base* of the system.

4.2.1 The Semantic Network

The Semantic Network (*SemNet*) represents the core of the system, as it affects all other part of the system and application and currently comprises around 100,000 nodes including also WordNet [Miller, 1990]. The representation is based on a directed hyper-graph. Nodes represent entities (e.g. *the company*) and events (e.g. *the company made losses*). The LOLITA *events* differ from the general concept of *event* and include, for example, statements such as "*John likes Mary*". Entities and events are connected in the semantic network with arcs which represent relationships. The knowledge is stored in the network by using control variables. Control variables are the essential information stored at each node, there are about 50 different control variables the most important of which are:

- **Rank.** This control gives the nodes quantification and can assume the following values: individual, (*the loss Company XY made in the first quarter of '94*), named individual (*Ferrari S.p.A.*), universal (*every loss*), prototype, bounded existential, framed universal or class.

- **Type.** This control values is very similar to grammatical qualifications with few exceptions and additions [Costantino *et al.*, 1996b]. For example the *relation* type mainly represents verbs, *attribute* represents adjectives and *entity* represents nouns [Smith, 1996]. A node can have the following type controls: entity, relation, typeless, event, fact, greeting, procedure, determiner, punctuation, attribute, mode, preposition, pronoun, conjunction or sub-conjunction.

- **Family.** This control groups nodes into semantic and pragmatic categories (families) to which they belong and can assume the following values: living, vegetal, animal, human, inanimate man-made, inanimate, animal or human, generic, inanimate organic, abstract, concrete, not human, temporal and location, organisation, and human-organisation. Family controls are also used to discriminate among the meanings of verbs. For example, "drive a car" and "drive sheep" will have different meanings of the verb "to drive" because *car* belongs to the family *inanimate man-made*, where *sheep* to the family *animal* [Smith, 1996]. The *family* control forms a *hierarchy* of nodes and allows fast searches in the semantic network.

These mechanisms allow the network to contain an elaborate knowledge-base (i.e. encyclopedic "world" knowledge, linguistic knowledge) which can be expanded by using most of the LOLITA applications which have been built so far and, in particular, the query application (see section 4.3.2). In figure 4.2 an example of

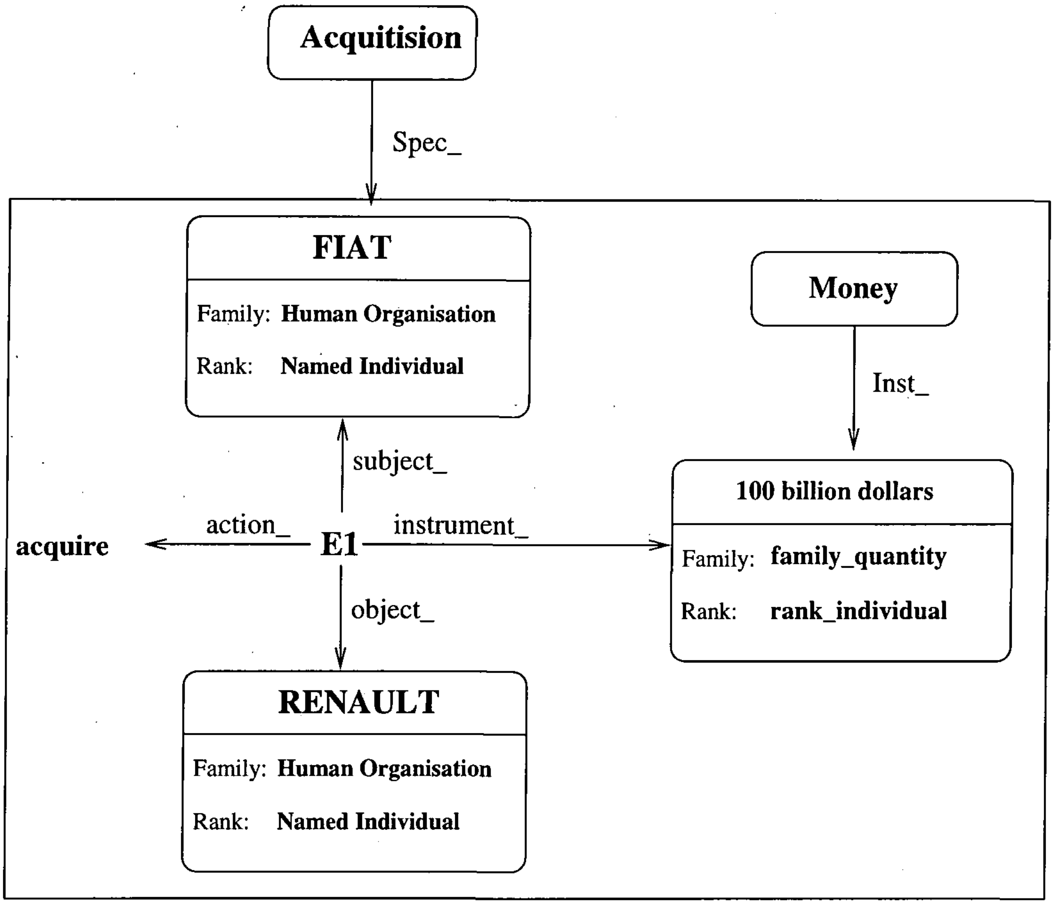

Figure 4.2: Representation for a typical financial event in the semantic network.

a takeover event corresponding to the sentence "*FIAT announced it acquired RENAULT with 10 billion dollars*" is shown, while figure 4.3 shows the representation of the two events corresponding to the fragment *10 billion dollars*[1].

Prototypical events

Prototypical information is used within the *LOLITA System* to restrict the kind of entities that can be used within a particular event according to the appropriate action.

An example is the "ownership" prototype which can be represented with the sentence *"human owners own things"*. This prototype restricts the subject of the verb *to own* to *humans* (which will belong to the set of humans who are also *owners*)

[1] The link to the concept of *acquisition* has been generated because the *acquisition-prototype* had been defined (see section 7.3.1).

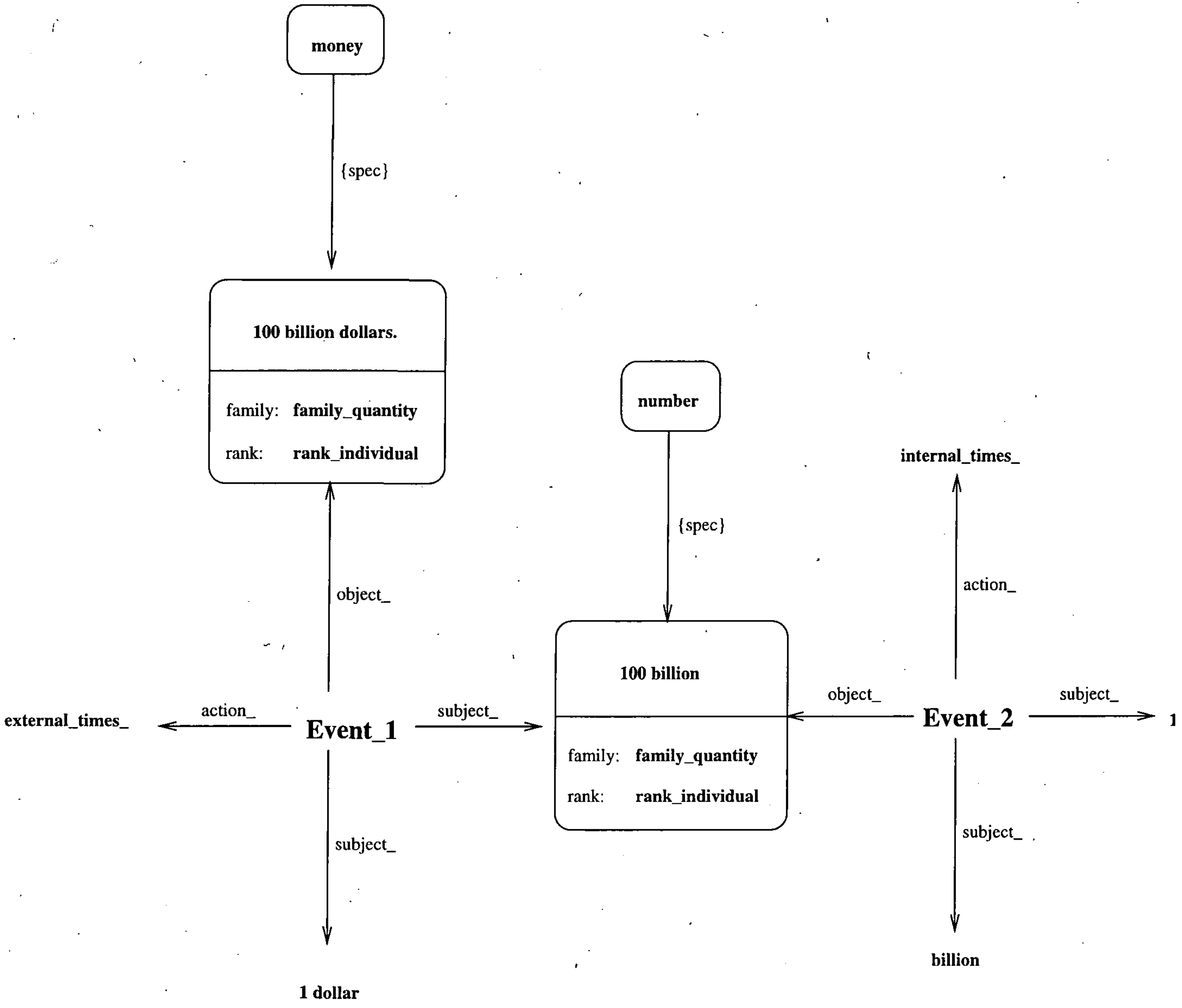

Figure 4.3: The SemNet representation for *100 billion dollars*

and the object to *non-humans.* This information will be used by the pragmatic analysis to accept or reject new events and for the disambiguation of meanings. In a sentence such as *he owns a car*, the pragmatic analysis will assume *he* as being human according the the *ownership* prototype.

4.2.2 Syntactic analysis

The syntactic analysis is the first step taken by the system for the processing of the input text and includes morphology, parsing and normalization. At the end of this stage the system has not performed any disambiguation or anaphora resolution.

Morphology

The first stage of the syntactic analysis is to build a surface representation of the input sentences. Words are separated according to the spaces in the input sentences and punctuation is used to separate the sentences into grammatical units. Contractions are expanded (e.g. "I'll" expanded to "I will") as well as monetary and numeric expressions (e.g. "10 $ million" to "million dollars"). Specific idiomatic phrases are transformed (e.g. "in charge of"). Misspelt words are recovered where possible and unknown ones are guessed [Parker, 1994]. The morphological process extracts the roots of the input words which are linked to appropriate lexical and semantic nodes, which will be used during the rest of the analysis. A *feature system* is used to preserve information such as number and case and word class (Noun, Verb) [Morgan *et al.*, 1995] and some semantic-based Features. Words are finally labelled with all possible categories for that word.

Parsing

The task of the parser is to group the words of a sentence into a symbolic structure representing their grammatical relationships. The parser produces a forest of parse trees normalised and ordered according to penalties. There are four stages in parsing:

- the **pre-parser** which identifies and processes monetary expressions (e.g. "10 million dollars"), which is implemented using a low-ambiguity simple grammar and a parser which attempts to find the largest non-overlapping sequences which match the grammar [Morgan *et al.*, 1995]; The simple grammar is a simplified grammar which is used to identify and convert into single units non-ambiguous phrases. Basically, the simple grammar treats parts of a sentence which can generally only be interpreted in a specific way. Treating them with specific rules rather than allowing the LOLITA parser to fully process them, reduces the number of possibilities which are considered by the parser and the possibility that they are wrongly interpreted. Phrases such as "100 million dollars" do not present any ambiguity and, therefore, it is possible to convert them in the simple grammar, as opposed to allowing the general cases.

  At present, the simple grammar covers mainly monetary values, identifying the key parts and marking them with the appropriate labels which will be then used by the Semantic Analysis module. For example, the parsing of a phrase such as:

  "*100 million canadian dollars*" will produce the following parse tree:

```
sen_phrase
 money_phrase
   quantity_currency
     quantity_plus
       quantity 100
       literal MILLION  * 2
     currency
       literal CANADIAN
       literal DOLLAR  * 4
```

The simple grammar includes all the possible currencies that can be recognised (dollars, pounds, marks etc.), all the possible quantifications (thousands, million etc.), and places (canadian, american etc.). It then uses these information to mark each word with the appropriate labels;

- the **parser** which is based on the Tomita algorithm [Tomita, 1986] and uses a large and highly ambiguous grammar written in a context-free style. The result of the parser is a "parse forest" which consists of all possible parses for the input sentence which, given the complexity of the grammar, can be extremely large;
- **decoding of the parse forest**. The forest is processed with the purpose of identify the lowest-cost trees. Such analysis is performed according to feature consistency and hand-assigned likelihoods of certain grammatical constructions [Morgan *et al.*, 1995];
- **selection of best parse tree**. The lowest-cost tree is ordered according to heuristics based on the tree form. Subsequent analysis (e.g. semantics) is performed on this parse-tree;
- **normalisation**. This stage applies several transformations to the parse tree with the goal of reducing the number of cases required in the semantic analysis. These transformations are performed preserving the meanings of the sentences. For example "RENAULT was bought by RENAULT" to "FIAT bought RENAULT".

In figure 4.4 the parse tree for the sentence *"FIAT bought RENAULT with 10 billion dollars"* is shown.

4.2.3 Analysis of meaning

Once the input sentences have been transformed into an ordered parse tree forest, the best tree must be interpreted and the semantic network updated with the new sentence: **FIAT bought RENAULT with 10 million dollars**

```
sen
 full_propernoun
   propernoun_not_comp FIAT [New]
 auxphrase_advprepph
   transvp
     verb BUY [Past] * 4
     full_propernoun
       propernoun_not_comp RENAULT [New]
   prepp
     prepNormMode WITH
     money_phrase
       quantity_currency
         quantity_plus
           quantity 10
           literal BILLION  * 3
         literal DOLLAR  * 4
```

Figure 4.4: Example of parsing

information by creating or updating SemNet nodes. Two modules are employed for analysing the meaning of the parse tree: *semantic* and *pragmatic* analysis.

Semantic analysis

The semantic analysis module maps the information available in the parse tree onto nodes in the semantic network. Existing nodes in the network are checked and eventually updated or linked to newly created ones. Each object or event in the parse tree must be mapped onto an appropriate existing or new node. In figure 4.5 the output of a part of the semantic analysis for the parse tree in figure 4.4 corresponding to the sentence "FIAT bought RENAULT with 10 billion dollars" is shown. The nodes corresponding to the object *FIAT*, *RENAULT*, *10 billion dollars* and to the event *FIAT bought RENAULT with 10 million dollars* have been created or already existed in the net.

The first step taken by the semantic analysis is to make references absolute. For example, the reference "I" must be referred to the person, company etc. who is the source of the information, while "you" to LOLITA. Temporal references

Sentence: **FIAT bought RENAULT with 10 billion dollars**

```
    * event: 96247 *
 universal_:
   event - 7688 - rank: universal  (happen_)  - suspended_
 subject_:
   FIAT - 96234 - rank: named individual - family: human organisation
                - suspended_
 action_:
   buy - 11034 -
 object_:
   RENAULT - 96238 - rank: named individual - family: human organisation
                   - suspended_
 instrument_:
   money - 96245 - rank: individual - family: state - suspended_
 time_:
   past_ - 20991 -
 date:
   12 May 1996
**********************************
 event:
  FIAT bought RENAULT by million 10 dollars.
```

Figure 4.5: Example of semantic analysis.

Fragment of semantic analysis for the sentence:
RENAULT rejected the price offered by FIAT because it was too low

```
  * event: 96233 *
universal_:
  event - 7688 - rank: universal  (happen_)  - suspended_
cause_of:
  event - 96255 - rank: individual  (reject)  - suspended_
subject_:
  price - 96247 - rank: individual - family: communication - suspended_
action_:
  be - 6458 -
mode_:
  too - 17446 - rank: universal - family: concrete
  low - 23894 -
time_:
  past_ - 20991 -
date:
  13 May 1996
status_:
  suspended_ - 29025 -
```

Figure 4.6: Example of semantic disambiguation

must be made absolute. For example, in the sentence *"the takeover offer will expire on Monday"*, *"this month"* must be referred to the date corresponding to the Monday after the date in which the sentence was processed. The step taken by the semantic analysis module is the disambiguation of the parse tree choosing among the possible interpretations. In the sentence *"RENAULT rejected the price offered by FIAT because it was too low"*, the attribute *"low"* will be attached to *"price"* rather than *"FIAT"* because of the fact that *"low"* cannot be an attribute of a company (human_organisation) but it is legal for a price (see figure 4.6). The semantic analysis module makes use of the *context*, which consists of the latest sentences analysed.

Pragmatic analysis

The task of the pragmatic analysis module is to check whether the new semantic information (consisting of new and updated nodes) produced by the semantic anal-

The sentence "FIAT acquire Roberto" is marked with "low belief"

```
  * event: 96242 *
universal_:
  event - 7688 - rank: universal  (happen_)
subject_:
  FIAT - 96236 - rank: named individual - family: human organisation - suspended_
action_:
  buy - 11034 -
object_:
  roberto - 19845 - rank: named individual - family: propername human
time_:
  past_ - 20991 -
date:
  13 May 1995
status_:
  suspended_ - 29025 -
belief_:
  low - 9302 - rank: universal - family: inanimate manmade
```

Figure 4.7: Example of pragmatic analysis.

ysis module is consistent with the one already available in the semantic network. The pragmatic analysis will therefore check and adjust the new information to better fit into the existing network. Prototypical information are used at this stage. For example, the sentence "FIAT acquired Roberto" (Figure 4.7) will be rejected by the pragmatic analysis module because of the fact that the object of an acquisition cannot be a human (Roberto), as defined by the acquisition prototype (see section 7.3.1), and will therefore be marked with a *low* level of belief. The level of belief is determined using *source controls* which implies looking at the source of the information, rather than at the information itself [Bokma and Garigliano, 1992].

Another task of the pragmatic analysis is to disambiguate among possible meanings of objects. This is done using a set of heuristics which are firstly applied to disambiguate the action of the event. Once the action is known, they are applied to the other unknown nodes of the event. The pragmatic analysis module takes into consideration the current *context* together with the *topic* of the text (the latter is supplied to the system in advance) if available. For example, the meaning of the verb *to buy* in a financial article will normally be:

Fact in the Semantic Network:

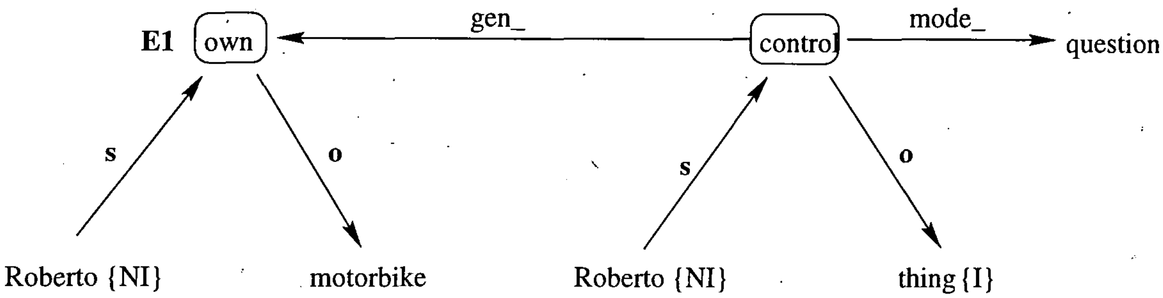

Figure 4.8: Example of question handled by the inference engine.

```
11034: To buy. (=> To acquire, To purchase)
```

rather than, for example:

```
25117: To buy. (=> To accept, To believe, To trust, to gather)
```

Using the *prototypical information*, the *topic* and *heuristical rules*, the pragmatic analysis module will choose the first meaning which is appropriate for the event.

4.2.4 Inference

The inference engine is used to *infer* information which is not directly available in the semantic network. Various kinds of inference are available within the LOLITA system such as *analogy*[Long and Garigliano, 1994], *deduction* and *induction.*

One of the main tasks carried out by the inference engine is to retrieve specific information from the semantic network. The inference functions available in the LOLITA System are, in fact, able to *answer* questions by returning events from the semantic network which *match* the question. For example, the question "*What does Roberto control?*" will return the event "*Roberto owns a motorbike*", since *control* is a generalisation of *own* (see figure 4.8).

The inference functions search in the semantic network for events with subject *Roberto*, action a specialization of *control* and object a concept which shares the same properties as *thing.*

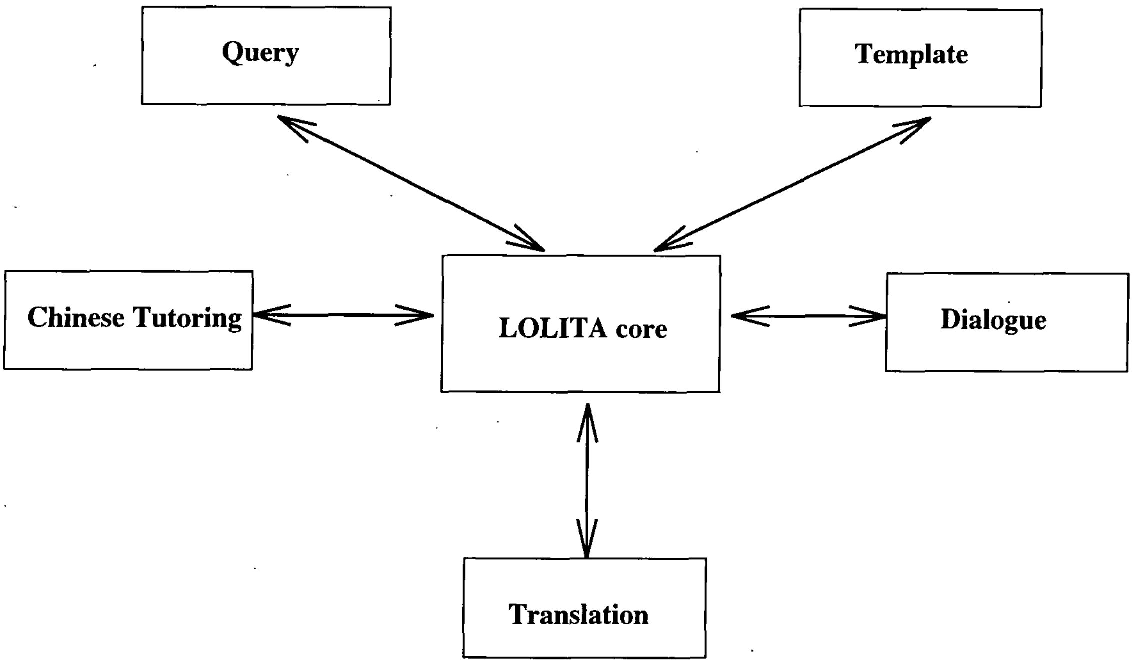

Figure 4.9: The LOLITA applications.

4.2.5 Generation

The LOLITA generator is used to rebuild surface language expressions from SemNet [Smith, 1996, Smith *et al.*, 1994]. The generator is widely used by most of LOLITA applications when an output in natural language is required, for example query, dialogue, translation and information extraction. The generator is able to produce NL sentences from any fragment or concept of the semantic network according to various kinds of controls, such as rhythm, length and colour which will affect the way in which the final NL sentences are produced.

4.3 LOLITA Applications

Various kinds of applications have been built on the LOLITA core (figure 4.9). The applications make use of the various modules of the LOLITA core and can access the semantic network directly or using the *inference engine.*

4.3.1 Analysis of Text

The basic task that the LOLITA System can perform is the simple analysis of text to build a representation of its meaning. The meaning of the text is then stored into the semantic network and is available for further processing. Rather than an application, this is a base operation that can be used for subsequent analysis.

4.3.2 Query

This application allows the user to interact directly with the system updating the existing knowledge and questioning the system on the information already available.

4.3.3 Translation

The Italian translation application emphasises the flexibility of the LOLITA core and of the Semantic Network. Few Italians rules have been added to the LOLITA grammar and enable the system to process basic Italian sentences. The information is stored in a language-independent form in the semantic network which is translated into English by the generator [Morgan *et al.*, 1994]. A Spanish version of the generator [Fernandez, 1995] allows LOLITA to produce basic sentences in Spanish, which allows English to Spanish machine translation.

4.3.4 Database front-end

An application for providing a NL front-end interface to conventional database programs is currently being developed [Garigliano *et al.*, 1995]. The queries in NL are analysed by the LOLITA core and stored in the semantic network. The database front-end application retrieves the appropriate information from the net and translates it into SQL [Hursch and Hursch, 1988] queries.

4.3.5 Chinese tutoring

The Chinese tutoring prototype application has been developed to help students learning Chinese to overcome transfer errors caused by mother tongue influence [Wang and Garigliano, 1992, Wand and Garigliano, 1995, Wang, 1994]. The tutoring system is able to ask the user to perform English to Chinese translation and check the input provided by the student which is parsed and diagnosed for transfer errors. The LOLITA core has been updated providing new chinese rules to the grammar and storing chinese data in the semantic network.

4.3.6 Extraction of Objects

An application for the identification of classes and objects for the development of applications is currently being developed [Mich and Garigliano, 1994] [Mich, 1996]. The project focuses on using the LOLITA System for supporting object oriented analysis and in particular of the early phases of natural language requirements analysis using the Object Modeling Technique [Rumbaugh *et al.*, 1991].

4.3.7 Dialogue

A dialogue application has been developed. The application is loosely based on Schank's script theory [Schank and Abelson, 1997] and has been implemented and interface with the LOLITA System and its generator [Jones, 1994]. The application is based on the concept of *Dialogue Structure Models* (DSMs), a schema which holds information about a stereotypical dialogue. The DSMs can not only be used to help understanding the input (making assumptions on what to expect), but also in the continuation of the dialogue, by constraining the generation of NL sentences by the system. Genetic algorithms have been applied to improve the dialogue [Nettleton and Garigliano, 1995, Nettleton, 1995].

4.3.8 Information Extraction

One of the applications developed on the LOLITA core is information extraction. Early stages of the development involved the design of the *incident* and *terrorist* template [Garigliano *et al.*, 1993] and the summary template. More recently the LOLITA system successfully participated in the MUC-6 competition [Morgan *et al.*, 1995].

Chapter 5

Information extraction in the LOLITA System

5.1 Introduction

This chapter discusses in detail the information extraction capabilities of the LOLITA System. The LOLITA System, as general purpose natural language processing system, provides the ideal framework for the development of information extraction applications in different domains.

5.2 Architecture of the information extraction application

The way in which the information extraction application works is straightforward. The source articles are firstly processed by the LOLITA system and stored in the semantic network. The task of the template application is then to search for the information needed in the semantic network accessing it directly or using the *inference rules.* Finally, the output is produced either using the *generator* or referring to fragments of the source document (see figure 5.1). The information extraction module interacts with the system also by setting the appropriate *topic*

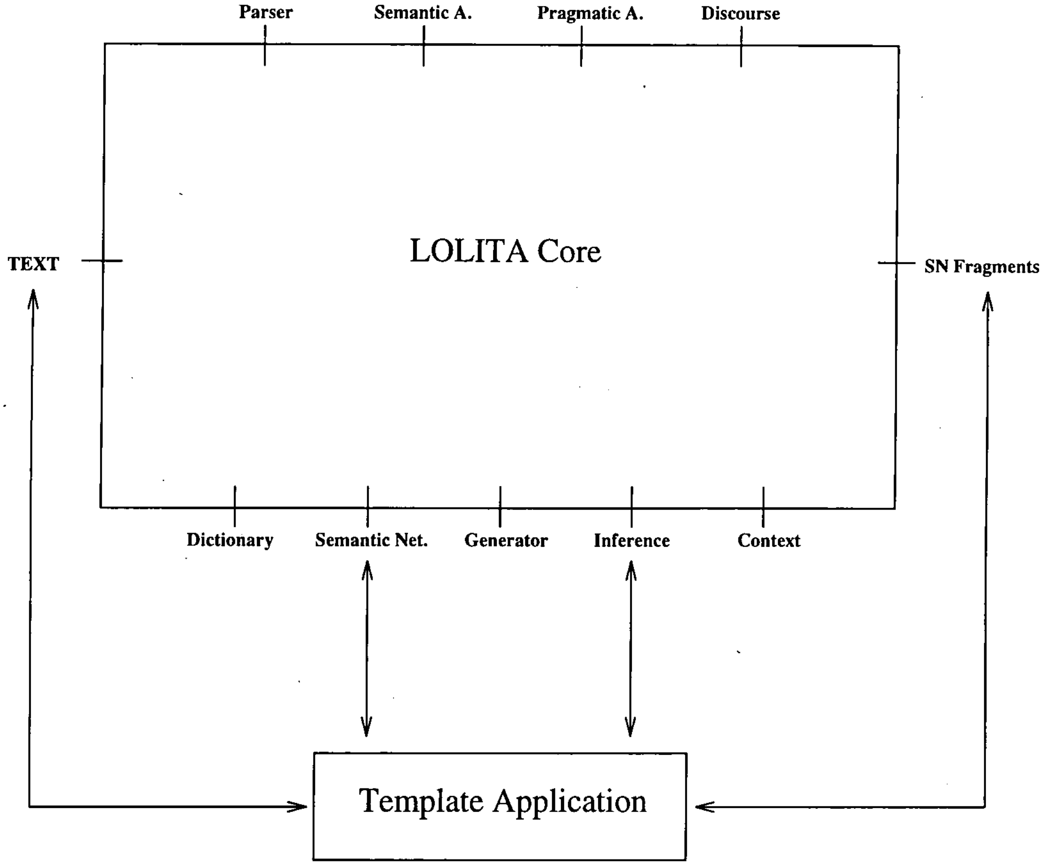

Figure 5.1: Information extraction within the LOLITA system.

of the source article, which is used by the *pragmatic analysis* module to assist with the disambiguation of the meanings of specific nodes.

The way in which information extraction is performed in the LOLITA System differs from other approaches and, in particular, from the systems emphasising *finite-state* analysis or *fragment parsing* techniques for three main reasons:

- the LOLITA System performs a uniform full analysis of all sentences of the source text. This allows the identification of important information within sentences which can appear non-relevant if analysed superficially;

- a full word sense disambiguation is performed on the source sentences. The system does not assume the meaning of a word according to the specific topic or domain of the source text. Differently, semantic and pragmatic analysis is used for the disambiguation of the word senses. Although the current *topic* is taken into account, it is only one of the many factors employed for the disambiguation;

- the large LOLITA knowledge bases, grammar, semantics and pragmatics are general and have not been specifically designed for a particular domain. This allows the easy portability of the system towards different domains.

This approach implies that the information extraction module relies directly on the performance of the core system and has the advantage of allowing the maximum degree of portability towards new domains, since the information extraction module is built on the domain independent LOLITA core.

5.3 The LOLITA templates

The LOLITA information extraction application is based on templates. The information extraction application provides an API (Application Programmer Interface) which allows the programmer of the LOLITA System to define new templates within the information extraction framework.

Each template is organised as a structure of four main elements (figure 5.2). The first element is the *template-name* which identifies the specific template among the templates available in the system. The second element is the *template rule*, which defined under which condition the template must be filled by the system. The template rule is called the *main-event condition* because the template is built by the application if the condition is *true*. The condition can by built by checking the control variables of the new or updated nodes produced by the analysis of the source text and verify that they satisfy certain conditions. The main-event condition can also check generalisations or specialisations of nodes and, in general, follow any link of the node.

For example, the *main-condition* of the *incident* concept-based template (see section 5.4.1) is built by checking if an event produced by the analysis of the source text has generalization *incident_events* which include: attack, explode, kill, intimidate, retaliate, wound, shoot and bomb. When the main-event condition

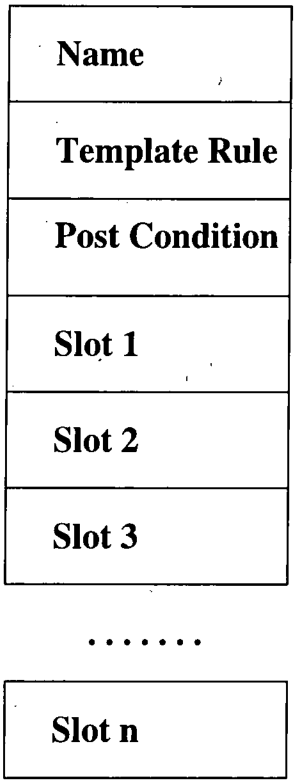

Figure 5.2: The structure of a LOLITA template.

is *True*, the node corresponding to the successful event or entity is stored in the *TemNodes* data structure, which will include all the nodes for which the main-event condition is *True*.

The third element of the template is the *template post-condition*, which is used by the template application to decide whether to build or discard the template according to additional conditions and the information identified in the template.

The fourth element of the template structure are the slots, which are filled by the application once the template main-events have been identified. Each of the slots has a name, a rule for filling the slot with the appropriate information and a set of values which are used for filling the slot (figure 5.3). In figure 5.4 the way in which the template application fills a template is shown. Slot rules define a relationship that must hold. The relationship is represented in different ways depending on the type of slots involved.

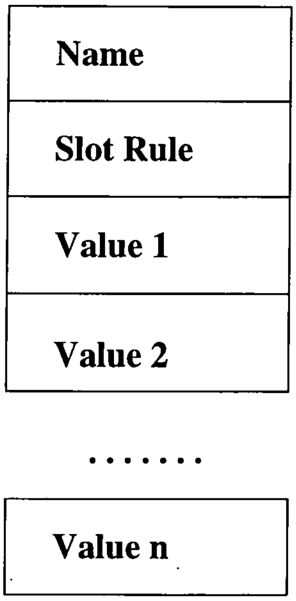

Figure 5.3: The structure of the template slots.

5.3.1 Types of slots

Four kinds of slots can be used in a template depending on the kind of rule used: *concept* slot, *string* slot, *text reference* slot and *template reference* [Costantino *et al.*, 1996a]. The slots differ in the way they are filled by the template application. The slot rule associated to each slot defines how the slot is filled by the application. Differently from the main-event, the slot rule condition cannot generally be a True/False condition[1], but must produce a result in the appropriate form which will represent the final contents of the slot. Similarly to the main-event condition, the slot rules can be built by checking the control variables, such as *rank*, *family* etc. and the links of the nodes, such as *generalisation*, *specialisation* etc. or using inference functions.

Concept slot

The *concept* slot represents the generic LOLITA template slot. The rule associated to the slot is used to identify the relevant concepts in the semantic network which are passed to the generator obtaining the corresponding English natural language

[1] An exception is represented by *string slots* which accept True-False assertions (see section 5.3.1).

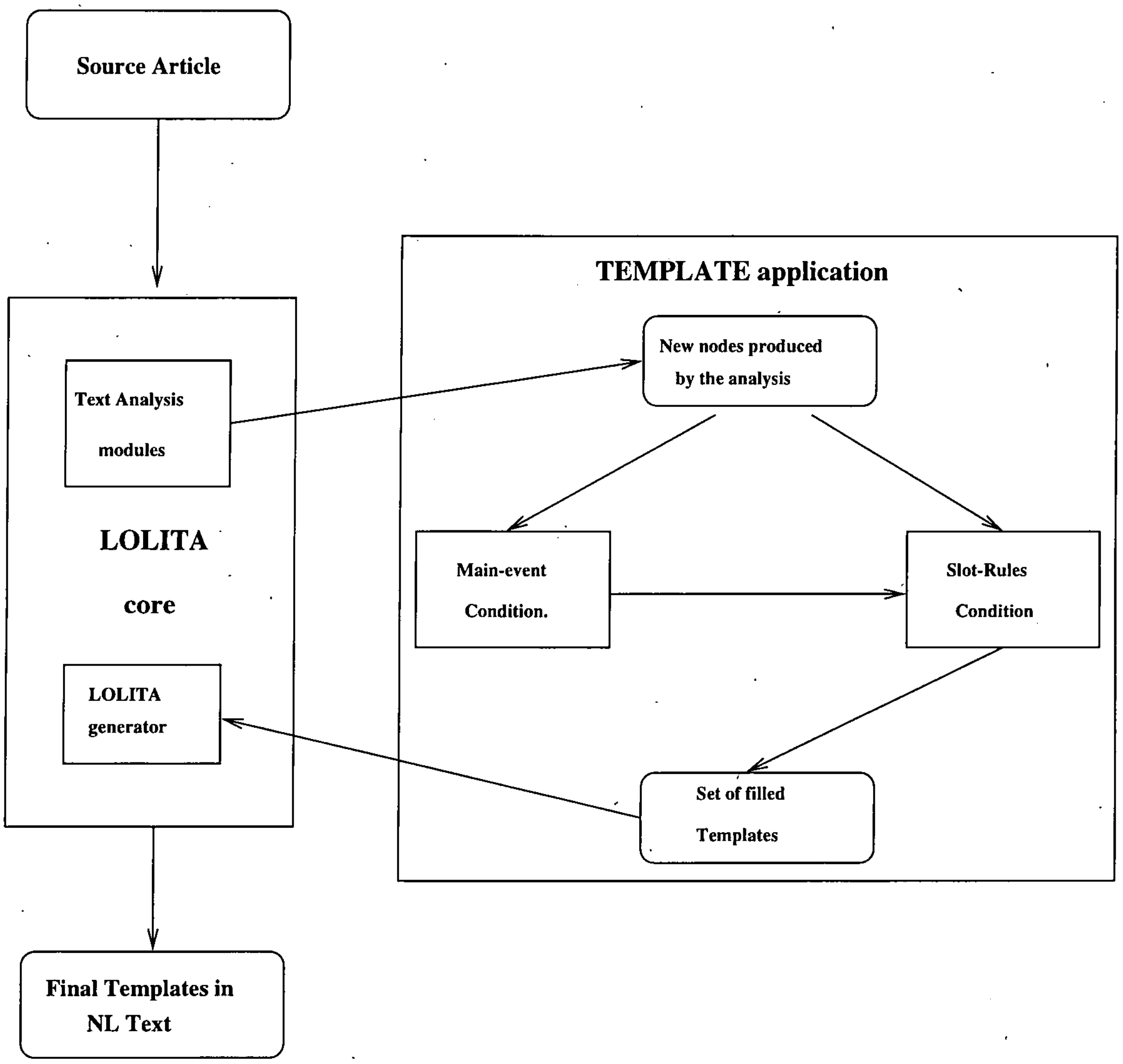

Figure 5.4: Information extraction within the LOLITA system.

text. This rule is extensively used in numerous LOLITA templates. For example, the rule for the *responsible* slot in the *incident* concept-based template (see section 5.4.1 returns the subject of the main-event under the condition that it belongs to *family_human* and returns its *noderef.*

String slot

This slot is used to produce the output directly from a given list of strings and not by using the English generator. It is mainly useful to produce a slot with a predefined number of alternatives which may not be present in the original text. The rules associated to the slot will produce a boolean True/False value which will be used to choose among the various strings available.

Text reference Slot

Text reference slots are used to produce output referring directly to fragments of the original text where is possible and the generator if a semantic network's concept does not correspond to any fragments of the original text. The slot is used when the exact copy of the original text is needed and is used extensively in the MUC-6 templates [Morgan *et al.*, 1995]. The text reference mechanism, introduced in the system specifically for the MUC-6 competition, provides a way of referring back to the original text without using the generator. The generator, in fact, relies heavily on the core analysis, and although it performs well given a correct analysis, errors in the analysis of the source text can lead to intelligible output. Textrefs, instead, are more robust in this sense, since their output consists of direct fragments of the original text. On the other side, the textref mechanism does not allow information which is not directly available in the original text to be represented, e.g. information produced by the inference system. In this case, the generator will provide the correct output.

The use of text reference slots in the MUC-6 templates was also due to the fact that the scorer required an exact copy of the source text and, therefore, the use of the generator would have reduced the number of correct fills.

Template reference slot

This slot is used to create a link to another template and, potentially, to other sources of information. The output of the slot consists of a pointer to the new template. The *template reference* slot provide the basic mechanism for handling *hyper-templates* in LOLITA which are widely used in the MUC-6 scenario templates (see section 5.4.3). The MUC-6 scenario templates are actually an *acyclic graph.* The LOLITA hyper-templates mechanism allows the use of *acyclic* and *cyclic* templates similar to the links available in hypertext documents.

The template reference slot refers to another template by stating the destination template and the rule from which the reference depends. In addition, the definition of the slot can force the system to produce one or more than one child templates.

The LOLITA hyper-templates can be produced in HTML format, which allows to follow the different templates using a standard HTML browser.

5.4 Types of templates available in the system

Three different kinds of templates can be built by the system depending on the kind of template main-event and the slots used.

5.4.1 Concept-based templates

Concept-based templates are structures where it is possible to identify a clear underlying top-level event to which all the information of the template's slots can be referred [Garigliano *et al.*, 1993]. Any kind of condition can be used for the definition of the main-event.

For example, the *takeover* of a company by another company can be considered a suitable event for building a concept-based *takeover* template [Costantino

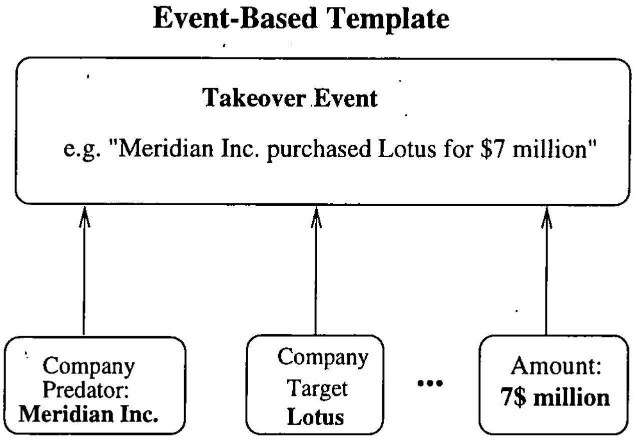
Event-Based Template
Takeover Event
e.g. "Meridian Inc. purchased Lotus for $7 million"
Company Predator: Meridian Inc.
Company Target Lotus
•••
Amount: 7$ million

Figure 5.5: The slots in a concept-based template refer to the main-event.

et al., 1996b]. In fact, all the information regarding the takeover: *amount*, *type of takeover*, *company target*, *company predator* etc. can be associated to the "parent" *takeover event* (see figure 5.5).

More than one concept-based template can be identified in a source document, according to the number of relevant top-level events that are generated by the semantic analysis of the article.

Once the template main-event has been identified, each slot is filled by searching the semantic network for the relevant information according to the slot rules, starting the search from the main-event.

Incident Template

The *incident* template has been the first template built in the LOLITA System and is composed of six slots as follows:

incident: the slot is filled with the main event of the template. All other slots are related to the contents of this slot;

where: the slot should contain the position or the location of the incident. Positions are defined as "coordinates in space" which do not necessarily have any concrete entity occupying that space, but are specified relatively to some of those entities. Locations are positions which are occupied by some concrete entity. A location is therefore a position, but not vice versa;

when: the slot should contain the date or time of the incident event.

responsible: the slot should be filled with the responsible of the incident. The responsible can either be the subject of the incident, in case the incident implies a human subject or the maker of the inanimate manmade that caused the incident;

target: the target of the incident is the object of the incident event. In case the object does not exist, the target is plausibly inferred from the position of the incident;

damage: the slot is filled with the objects that have been damaged because of the incident event which are identified in four different ways:

- the object of damage events with the original incident event as subject of the damage event;
- the location of the incident events that are subject of a damage event;
- the event itself in case it is a damage event;

5.4.2 Summary templates

Summary templates represent structures for which it is not possible to identify an underlying main-event. A summary template is basically a collection of objects stored in different slots which may not directly refer to the same concept or relate to each other (see figure 5.6). For example, a summary template can be composed of the following slots: *personal names*, *organizations*, *locations*, *temporal*, *acronyms*, *monetary values*, *descriptions*, *animates*, *inanimates* etc.

Summary templates are built differently from the concept-based templates. In fact, the main-event is *True* for all the new nodes generated by the semantic analysis of the source text. Consequently, the slot fill-in rules will be applied on all nodes, since the main-event is *True* for all nodes.

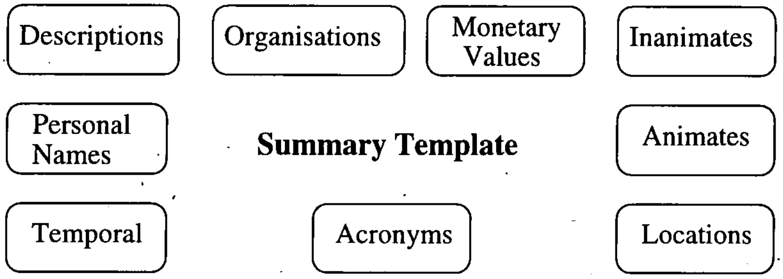

Figure 5.6: The slots in a summary template do not refer to the same concept.

Although the information is stored in a template, *summary* templates are more similar to a *named-entity* task, than to a proper template. A summary template, in fact, is unable to provide any relation between the various information which have been extracted from the source text.

5.4.3 Hyper templates

Hyper-templates are structures whose slots can refer to other templates. This mechanism is extensively used in the MUC-6 management scenario templates. The hyper-templates are implemented using template reference slots to point to the templates (see section 5.3.1). The hyper-templates mechanism is potentially usable for linking different kinds of information to the template, not necessarily extracted from the source text, such as company databases or historical share prices.

5.4.4 The MUC-6 templates

The MUC-6 competition has provided an opportunity to evaluate the approach used in the LOLITA System on some very specific tasks [Morgan *et al.*, 1995]. The results of the final evaluation are quite far from those of the best systems based on finite-state or statistical techniques. The results, however, have shown that the general-purpose LOLITA core based on deep natural language understanding can be successfully used for domain-specific applications. Three different interdependent templates have been used for the final evaluation: *organization*, *person* and *management scenario*. The first two templates have been employed for both template element and scenario templates tasks and have been implemented as concept-based templates, while the *management scenario* template has been implemented as *hyper-template.* The MUC-6 templates are built using text-reference slot to match the requirements of the scorer.

The Organization template

The organization template is implemented as *concept-based* template. The template is built under the condition that the node being analyzed is an organization. A MUC-6 organization is defined in LOLITA as the node belonging to the families *organization* or *hyman_organisation* and having *rank_individual.* Differently from the *incident* template, the slots are filled with information which is not necessarily semantically connected to the main-event but can be referred to it. The slot "ORG_ALIAS", which contains the alternative names used in the text to describe the organization, is an example of this. The slot is filled with concepts which are not semantically connected to the main-event but are relevant to the slot.

The Person template

The person template is also implemented as an *concept-based* template under the condition that the node being analyzed is a person. A MUC-6 person is defined in LOLITA as a node belonging to the family *human* or *animalOrHuman* and having *rank_individual.* Similarly to the organization template, the slots are filled with information which is not necessarily semantically connected to the main-event.

The Management template

The management template is implemented as a collection of 3 *hyper-templates*, *TEMPLATE*, *SUCCESSION_EVENT* and *IN_AND_OUT* which refer to the *ORGANIZATION* and to the *PERSON* templates (see figure 3.11). The top-level template is simply called *TEMPLATE* and is linked to the *SUCCESSION_EVENT* template. The *SUCCESSION_TEMPLATE* is linked to the *IN_AND_OUT* template and to the it ORGANIZATION template. Finally, the *IN_AND_OUT* template is linked to the *ORGANIZATION* and to the *PERSON* template. The *management scenario* template is therefore implemented as a collection of templates which are built according to specific rules associated to the *template-reference* slot.

5.5 Advantages and disadvantages of the implementation

As we have already seen, the information extraction module is separated from the core system. This solution offers two main advantages. First of all, any improvement in the analysis capabilities of the LOLITA core directly improves the performance of the information extraction application, which does not need to be rewritten. Second, new templates can be added independently, without the need of changing any part of the core system, allowing the maximum degree of portability towards new domains.

Moreover, the two can be developed separately. Any improvement in the LOLITA core analysis is directly reflected in the information extraction performance, while new templates can be added independently from the original core system allowing the maximum degree of portability towards new domains.

The main disadvantage of the LOLITA information extraction module is that it requires a highly technical knowledge of the semantic network representation. The user of the application does not have any control over the definition and modification of the templates which, although can be acceptable within a research group, it is not for real applications.

This problem is addressed and solved in this work and will discussed in detail in chapters 8 and 9.

Chapter 6

Definition of the financial templates

6.1 Introduction

This chapter focuses on the LOLITA financial information extraction system, which has been designed in close contact with experts from the financial sector. The design of the application has been done following the natural language engineering principles described in section 1.3. The templates identified in this chapter present a reasonable solution for the design of an information extraction system for finance. In section 6.2 we discuss the approach taken for designing the LOLITA financial information extraction system, while in section 6.3 the architecture of the system is outlined, with particular emphasis on the financial activities approach, which has been employed for the definition of the financial templates.

6.2 Designing a financial information extraction system

Information extraction applied to finance represents a new issue in the financial community and, therefore, development strategies for financial information extraction systems have not yet been studied.

The lack of a known methodology for the definition of the requirements of a financial information extraction system has lead us to the use of an empirical approach which tries to define the needs of the financial operators and, from there, leads to the definition of the characteristics of the application.

In our view, three main crucial aspects have to be defined for the success of a financial information extraction system [Costantino *et al.*, 1996b]: the type of system, the information to be extracted and the user interface.

6.2.1 The type of system: real-time / batch

Two different types of systems have been identified according to the kind of source articles to be processed by the system.

A first group of systems comprises those able to process *real-time news.* These systems should be able to process the incoming data in a reasonable amount of time and show the results (consisting of templates) to the operator. Ideally, the application could be used by information providers who would be able to provide some qualitative analysis to their customers (in addition to the relevant amount of quantitative analysis already supplied). However, a real-time information extraction system could also be used by other financial operators to process and summarize incoming real-time news from on-line services - i.e. *Dow-Jones* or *Bloomberg.* The critical aspect of this first group of systems is the ability to process the news in real-time in sufficient detail. Therefore the issue of *feasibility* (which comprises *efficiency*) is extremely relevant.

A second group of systems would include those able to process articles from other sources, such as newspapers (e.g. *The Financial Times* or *The Wall Street Journal*) or from magazines (e.g. *The Economist*). These articles tend to include much deeper analysis of the news (see figure 3.22). The system would therefore extract the most relevant information and the financial operators could make use of the information for "*off-line*" tasks, such as analysis of the effects of news on price behaviour. Given the size and complexity of some of the source articles, the operator cold reduce his data-overload of qualitative information.

6.2.2 The information to be extracted (Task definition).

The definition of the extraction task is the key-point for the design of a successful financial information extraction system. A hypothetical system with recall and precision both equal to 100 per cent but extracting the wrong information would be totally useless for the financial community. The information that can be extracted can vary greatly. For example company names, values, quotations of shares, relationships between companies (ownerships, stake etc.), names of products and many other kinds of events can potentially be extracted.

The definition of the information to be extracted is directly linked to the kind of output to be produced. A first decision is the number of possible templates to be extracted (one or more) and the slots associated to the templates. Few methodologies for the design of templates have been identified. Onyshkevych [Onyshkevych, 1993] defined 6 aspects related to the design of templates (see section 3.4): *descriptive adequacy*, *clarity*, *determinacy*, *perspicuity*, *monotonicity* and *reusability.* However, rather than a methodology for building new templates, the criteria are requirements that the templates need to satisfy to be acceptable.

Two main elements must be chosen for the definition of the appropriate templates for an information extraction application:

- **The kind and number of templates allowed**: a single template, more than one templates, multiple templates organised in a tree or a graph structure (e.g. the MUC-6 scenario templates).

- **The method for defining the templates**: pre-defined templates, user-definable templates, automatic extraction of templates.

Defining a single template.

The first solution would be to define only one template. This template should be able to represent all the relevant information in the input articles. Even if, in theory, this should be possible, in practice it would probably lead to a situation in which the template consists of an extremely large number of slots which would be unfilled for most of the articles. Moreover, the template should probably be constructed as a composition of various "sub-templates", each of which with different main-event associated conditions. The kind of articles that can be found in the financial domain is extremely large. A financial article can include information about, for example, *takeovers*, *mergers*, *privatisations*, etc. which are all relevant and probably associated with other relevant data, for example the *value* of the takeover, the *date* of the announcement etc. Therefore, the creation of a single template is possible, but probably not very useful for the financial operator [Costantino *et al.*, 1996b].

Defining more than one template.

A better possibility for the design of a financial information extraction system is to define more than one template. This solution seems reasonable considering the large amount of different kinds of events and objects that can be found in the financial domain. However, the identification of the templates is rather difficult and, if carried out wrongly, can lead to the extraction of information which does not reflect or is unable to express the contents of the original article. A template for a financial article could, for example, be composed as follows (summary template):

```
Template: Summary
        Personal names:
        Organizations:
        Temporal:
        Acronyms:
```

in this template each *slot* does not refer to a specific event. The operator will therefore find it difficult to relate the information of the template to the contents of the source article. The template does not share the *descriptive adequacy* property, as described in [Onyshkevych, 1993].

A possible way to ensure that the templates share the *descriptive adequacy* property is to define them in close contact with financial operators, who can provide some relevant feedback for each of the possible template definitions.

Other templates structures

The templates can be potentially organised in other structures, such as graphs and trees. An example of complex template structure are the MUC-6 scenario templates, which have been analysed in more detail in section 3.5.1 and, in particular, in figure 3.11.

Pre-defined templates

As far as the method for defining the templates is concerned, the simplest way is to provide them directly in the information extraction system. This allows the user to have immediate access to the information extraction capabilities of the system. Most of the systems which have been implemented so far and, in particular, the ones regarding the financial domain (e.g. ATRANS [Lytinen and Gershman, 1986]), are based on pre-defined templates. This is because the systems were built for specific tasks already fully determined during the design of the system. Pre-defined templates can therefore be used when the requirements of the systems are fully known at the time of its design. Information extraction systems based on pre-defined templates, however, do not normally allow the user to make any change to the existing templates, or add new templates to the collection. This can be a serious limitation for specific kinds of applications.

Automatic selection of templates.

Another possibility would be to automatically generate possible templates definitions directly from a sample of the source articles. The system developed by Collier

Source: The Financial Times (Internet) - http://www.ft.com - Mar 25, 1996
BELGRADE/MOSCOW - SERB - Bosnia's former warring parties grudgingly stepped up the release of prisoners over the weekend after a warning from the international community that reconstruction aid could be delayed unless the Dayton peace agreement was observed. The warning was issued at a meeting in Moscow of foreign ministers from the the contact group - the US, Russia, Britain, France and Germany - which sponsored last year's accord. Late on Saturday, Bosnia's Moslem-led government freed 109 Bosnian Serbs from a jail in the northern town of Tuzla.

Figure 6.1: An non-relevant article for a generic financial operator.

[Collier, 1994, Collier, 1996], for example, is able to produce a template definition from source texts (e.g. the MUC-6 scenario templates). This possibility has not been considered in the context of this work. This is because the templates definitions automatically created by the system may include information which is not relevant for the financial operator. The article in figure 6.1, for example, contains non-relevant information for the financial operator's investment decision-making process. In the same way, the automatic template generation process could potentially skip some information which is instead important for the financial operator.

User-definable templates.

The last option for the definition of the information to be extracted from a source text is to allow the user to define his own templates, without providing any predefined structures. Ideally, the user should state the information to be extracted by supplying natural language input sentences which describe the conditions under which the templates and the slots should be filled. This possibility should, in theory, remove any constraints for the users who should be able to program the system to exactly match their needs. In practice, for a financial information extraction application, the users would often be unable to provide in enough detail the requirements for the definition of the templates. The financial operators, in fact, are usually not familiar with the information extraction (and NLP in general) technology. It seems therefore reasonable that a minimum predefined framework within the application should be provided (for example basic general-use templates), while the user could define specific templates which are not present in the basic installation, once they are familiar with the technology and the functionality of the system[1]

6.2.3 The user interface.

The user interface is an important element for the financial application as a "complete" package. Financial operators are often not accustomed to the use of text-based or command-based user-interfaces. However, text-based interfaces can be potentially used if the target customers of the financial information extraction application are *information providers* who will supply the extracted information to their customers using their own specific user-interface. The financial information extraction system will be probably "embedded" in their own information systems and will not be visible to the end-users. In all other cases, where the potential customers are end-users an appropriate user interface is extremely important for the commercial success of the application. Most of information extraction systems developed so far are strictly text based, since they have been developed for specific and "internal" uses, rather than for commercial purposes.

6.3 The LOLITA financial information extraction system

The design of the LOLITA information extraction system has been done in close contact with experts of the financial sector[2] and according to the NLE methodology (see section 1.3).

[1] The pre-defined templates are relatively similar to the concept of *"template-elements"* in the MUC-6 competition (see section 3.5.1), where the basic-templates such as *organization* or *person* are pre-defined and have then been used for the definition of the scenario-based templates. Differently from the MUC-6 templates, however, the financial templates presented in this work are also based on *events* rather than only objects.

[2] I am extremely grateful to Stephen Eckett, managing director of Numa Financial Systems Ltd and former director of Bankers Trust Securities Ltd., for his precious help in designing the application and in following each stage of its development.

6.3.1 The type of system

As far as the type of input for the system is concerned, the target source documents that have been chosen for the LOLITA financial information extraction system are articles from financial newspapers and, in particular, from the *Financial Times* on CD-ROM. Articles from newspapers or magazines tend to include, as opposed to articles from on-line news services, analysis, interpretation and comments regarding the facts. Therefore, the financial operator has often to spend a considerable amount of time scanning through the articles trying to identify the most relevant information. In case he makes use of information retrieval packages, he will only identify the most relevant articles, but he will still need to identify the most relevant information within the articles selected. The information extraction system will instead be able to select the relevant information in each of the relevant articles leading to a marked reduction of the time usually spent by the financial operator in reading and analyzing the articles. Moreover, if we consider the number of articles on specific topics stored on a single CD-ROM (some thousands), the time of accessing to the relevant information by the operator is extremely high.

Articles from real-time on-line services tend instead to be rather summarized and, most of the times, a simple "report of facts", excluding any sort of interpretation. Further processing of the news, although it is possible, is therefore often unnecessary. Information providers tend usually to categorize and classify the news articles and, therefore, further classification is often not needed. A financial information extraction system processing these kind of data would, in addition, need to extract information in real-time.

The LOLITA information extraction system has been designed with articles from newspapers in mind. However, the system is also able to process articles from on-line news services.

6.3.2 The information to be extracted

The definition of the templates of the LOLITA financial information extraction system has been carried out in close contact with experts of the financial sector. The potential target customers of the LOLITA financial application are any financial operator that needs to have access to qualitative information to support his decision-making process.

The design of the information to be extracted by the application started by identifying the qualitative information which is normally needed by the financial operators to support their decision-making process. Operators in the financial markets are normally interested in reading and analyzing *any* news which is likely to influence the price of shares in the stock exchange, while they will normally skip any other news which is unlikely to influence such price. This observation has lead us to the definition the concept of *financial activity*.

- A **financial activity** is an event which is likely to influence the price of shares and, therefore, influence the decision-making process of the players of the market regarding these securities [Costantino *et al.*, 1996c].

The *financial activities* represent the information that the operators are normally interested in reading, analyze and extract from a source article. A sentence such as:

... *"the German central bank has reduced the interest rate of 1 per cent"* ...

can be considered a *financial activity* (interest rates movement), since the event is likely to influence people's behaviour and their investment decisions. The financial operator will therefore be extremely interested to know the relevant information about the event, such as the amount of the increase, when it has been decided, etc. since the event is likely to produce a marked positive impact on the quotations of shares in the stock exchange.

A relevant number of financial activities has been identified, with the help of financial experts and include: merger, takeover, flotation, new issue, privatisation, bankruptcy, etc. The activities chosen are those which are more likely to have a direct impact on the share price of the stock. However, since direct relations between the share-price and the various events cannot be directly measured, the choice of the activities was based mainly on the knowledge and experience of experts of the financial sector.

The LOLITA financial information extraction system is based on the **financial activities approach**, which consists in associating each of the financial activities with a corresponding template [Costantino *et al.*, 1996c]. For example, the *takeover* financial activity is associated with the *takeover* template, which includes all the information which refers to the takeover in appropriate slots. The solution of having more than one template, one for each financial activity, provides a complete representation of the relevant contents of the source articles. If a particular article comprises two different financial activities (e.g. a *takeover* and a *merger*), two different templates will be created for the same article (corresponding to each of the two financial activities).

Following the *usability* criteria of the NLE approach, the user can finally define his own templates using the specific user-definable template interface provided by the system (see section 8.2). In this way, the user can define specific templates which do not necessarily follow the *financial activities approach* but are needed for specific reasons.

The following financial activities have been identified: *takeover*, *merger*, *flotation*, *new issue (of shares)*, *privatisation*, *bankruptcy*, *new stake*, *dividend announcement*, *overseas listing.*

Once the financial activities which can influence the financial operator's behaviour have been defined, a specific template has been associated to each of the activities. The templates have been designed identifying the *objects* which are relevant for each activity (e.g. a takeover). These objects correspond to *slots* of the templates. The objects identified for the templates represent the information needed by the financial operator to gain complete knowledge regarding the specific event. The financial operator should be able to take an appropriate investment decision based on the contents of the template without having to refer to any additional information related to the specific event. The identification of the relevant objects for each financial activity has been carried out in close contact with the experts of the financial sector.

The financial templates have been defined as follows:

- **TAKEOVER template:**
  the template has to be instantiated when a company is, will be or is expected to be acquired by another company, an institution or an operator.

company target:	the name of the company target
company predator:	the name of the person or company performing the takeover
type of takeover:	{FRIENDLY, HOSTILE}
value:	Total value of the takeover
bank adviser predator:	the name of the bank advising the predator
bank adviser target:	the name of the bank advising the target
expiry date:	the date in which the offer expires
attribution:	the name of the company or person source of the news
current stake predator:	the amount or value of shares already owned by the predator in the company target at the time of the takeover.
denial:	the name of the person or company who denies any of the information in the template.

- **MERGER Template:**
  the template has to be instantiated when a company is, will be or is expected to be merged with another company.

company 1:	the name of the first company involved
company 2:	the name of the second company involved
new name:	the name of the new company
date of announce:	the date in which the merger is announced
date of merger:	the date in which the merger takes place
comments:	comments of the company's chairman or directors
attribution:	the name of the company or person source of the news source of the news
denial:	the name of the person or company who denies any of the information in the template

- **FLOTATION Template:**
  the template has to be instantiated whenever a company is, will be or is expected to be floated on the market.

company name:	the name of the company floated
price:	the price of the shares offered
value:	the nominal value of the shares offered
announce date:	the date in which the flotation is announced
listing date:	the date in which the flotation takes place
financial adviser flot.:	the bank advising the flotation or the company which is floated
attribution:	the name of the company or person source of the news source of the news
denial:	the name of the person or company who denies any of the information in the template
industry sector:	the stock market industry sector in which the company will be listed

- **NEW ISSUE Template:**
  the template has to be instantiated whenever a company has, will be or is expected to issue new shares on the market.

company:	the name of the company issuing new shares
comp. financial advisor:	the bank advising the new issue
issue currency:	the currency in which the bonds are issued
issue value:	the price of the bonds offered
announce date:	the date of the announce
launch date:	the date when the issue will take place
listed:	the name of the stock exchange where the shares will be listed
attribution:	the name of the company or person source of the news source of the news
purpose:	the purpose of the new issue of the company
denial:	the name of the person or company who denies any of the information in the template

- PRIVATISATION Template:
  the template has to be instantiated whenever a company is, will be or is expected to be privatised.

company name:	the name of the company privatised
stake to be privatised:	the percentage of the total shares to be privatised
price of the shares:	the price of the shares offered on the market
value of the shares:	the nominal value of the shares offered

announce date:	the date of the announce
privatisation date:	the date in which the privatisation takes place
bank adviser company:	the bank advising the privatisation or the company privatised
attribution:	the name of the company or person source of the news
denial:	the name of the person or company who denies any of the information in the template
industry sector:	the sector of the stock exchange in which the company will be listed

- BANKRUPTCY Template:
  the template has to be instantiated whenever a company is, will be or is expected to be declare bankrupt.

company name:	the name of the company declared bankrupt
receivers:	the name of the receivers of the company
date of announce:	the date of announce of the bankruptcy
denial:	the name of the person or company who denies any of the information in the template

slots

- NEW STAKE Template:
  the template has to be instantiated whenever a company or a person has, will or is expected to acquire a new stake in another company or an institution.

company target:	the name of the target company or institution
company taking the stake:	the name of the person or the company acquiring the stake
Value:	the amount of the stake to be bought
attribution:	the name of the company or person source of the news
denial:	the name of the person or company who denies any of the information in the template

- DIVIDEND ANNOUNCEMENT Template:
  the template has to be instantiated whenever a company has, will or is expected to announce dividends.

company name:	the name of the company announcing the dividends.
dividend per share:	the value of the dividend per share of the company
type of dividend:	Type of dividend to be distributed

- OVERSEAS LISTING Template:
  the template has to be instantiated whenever a company is, will or is expected to be listed in a foreign Stock Exchange.

```
company name:
COMPANY_NAME
overseas exchange:        the name of the overseas Stock Exchange
type of securities:       the type of securities of the company
                          that will be listed at the Stock Exchange
announce date:            the date of the announce
date of listing:          the date when the listing takes place
attribution:              the name of the company or person source
                          of the news
denial:                   the name of the person or company who
                          denies any of the information in the
                          template
```

Chapter 7

Implementation of the financial templates

7.1 Introduction

In this chapter we analyse the implementation of the pre-defined templates of the LOLITA financial information extraction system described in chapter 6 within the existing LOLITA framework. The LOLITA general-purpose system has been discussed in chapter 4, while the generic approach for information extraction within the LOLITA system has been presented in chapter 5.

7.2 Defining the rules for the financial templates

After defining the kind of output that the financial information system has to produce, the second important step is the definition of the rules that the system will use to fill the template's slots. The first main step is identifying the rule for the main-event. Since the financial-templates are classifiable as *concept-based* templates (see section 5.4.1), the identification of the correct main-event is of crucial importance. If, for example, we are trying to fill a takeover template (see section 6.3.2), first of all, we will need to define *when* and under which *conditions* the template has to be created. The system will then scan through all the new nodes created by the semantic analysis of the financial article, looking for specific events/nodes satisfying the main-event condition.

The identification of the appropriate rule is a difficult and complex process since, if is done improperly, it will lead to great losses of recall/precision. Therefore, the identification must be done on a significant quantity of relevant and non-relevant articles.

The identification of the appropriate rule for the main-event and, in general, for all the slots, has to be carried out designing the relevant *semantic structures* able to identify the relevant information in the source text. These semantic-structures differ from one another for the different *meaning*, rather than for differences in the structure or form of the sentence. The LOLITA *semantic structures* are therefore rather different from the *patterns* of information extraction systems based on *finite-state* techniques. For example, the following two sentences share the same *semantic structure*, although the form of the sentences is different:

```
FIAT acquired RENAULT.
```

and:

```
the acquisition of RENAULT by FIAT.
```

The *semantic structure* of the two sentences is:

```
Event 1:
   Action:  acquire
   object:  RENAULT
   subject: FIAT
```

The above notation is used in this chapter to represent semantic structures.

Once the semantic-structures have been identified, it is important to verify whether they will be correctly recognized by the general-purpose LOLITA semantic

/ pragmatics / inference rules and categorized according to the user's need or they need to be supported by domain-specific knowledge to be added to the system. Some of the rules, in fact, might present semantic output which differs from what we would expect in the analysis of a generic text. For example, the system would normally be unable to *infer* that the sentence "Company XX took full control of company YY" corresponds to a company takeover. Therefore, the domain-specific knowledge will have to be added to the system. The rule will therefore look for any of the *semantic structures* identified and, in case of the main-event, return "True" and produce the template or, in case of slot rules, fill the slot with the appropriate information retrieved from the semantic network.

7.2.1 Prototypes

Prototypical information (see section 4.2.1) is used within the LOLITA system to restrict the kind of entities that can be used within a particular event according to the appropriate action. Prototypical information is particularly useful in filling *concept-based* templates. This is because it will ensure that the events produced from the analysis of the source text include the correct kind of entities.

7.2.2 Domain-specific knowledge

The performance of the information extraction system can be improved with the addition of domain-specific knowledge which may differ from the LOLITA system world knowledge. The domain-specific knowledge is used to capture specific meaning of the source text for the particular domain, for example the financial domain. For example, the system would normally be unable to *infer* from the sentence:

```
Company XX tool full control of company YY
```

that this corresponds to a *takeover*. The domain-specific knowledge ensures that the event is correctly understood by the financial application as a *company takeover.*

This knowledge can be placed at two different levels in the system. A first choice is to store the specific knowledge directly in the Semantic Network of the LOLITA system using existing facilities. The LOLITA core system would use the knowledge during the analysis of the source text. The introduction of this knowledge in the semantic network, however, can be difficult and can cause logical problems within the network. This leads to the second possibility, which is to incorporate the domain-specific knowledge directly in the information extraction system, rather than in the core system.

The domain-specific knowledge for the LOLITA information extraction system is stored directly within the application, rather than stored in the semantic network. The knowledge consists of "*semantic-rules*" which are used to capture specific events and objects from the analysis of the source articles performed by the core system. The domain-specific knowledge of the financial application consists of *semantic-rules* which are used to locate specific events and objects from the analysis of the source article which are relevant for the financial domain.

7.2.3 Unification

One of the most relevant issues for the success of the financial templates is the correct unification of events referring to the same concept by the LOLITA core system. This applies to the *main-event* and any other *slot*. The concept of unification is that if two events are referring to the same *semantic* concept, but they are reported in two different places in the same text, the LOLITA core should *unify* them and create a unique event, as composition of the relevant information extracted from both source events. For example, the two following sentences should be unified by the LOLITA system:

sentence 1

Airtours, the UK's second largest holiday tour operator after Thomson, has acquired one of the UK's leading long-haul brand names.

sentence 2

Tradewinds has been acquired from International Travel Connections, a privately-owned company based in Chester, for Pounds 250,000 cash.

The system should *unify* the two events creating one single event, as composition of the relevant pieces of information of the two sentences. A correct unified event for the two sentences should be, ideally:

```
Airtours, the UK's second largest holiday tour operator after Thomson,
has acquired Tradewinds, one of the UK's leading long-haul brand names,
for Pounds 250,000 cash.
```

The unification would allow the financial template application to retrieve all relevant information attached to the acquisition event, otherwise impossible to identify.

The unification is extremely relevant for the main-event of the template as well as any other slots (i.e. takeover value, company target etc.). The system should also be able to identify that two different nodes refer to the same object. In the two sentences shown above, the system should be able to identify that "*the leading long-haul brand names*" and "*Tradewinds*" refer to the same object and, therefore, should share the same node.

Unification is carried out at different stages of the processing of the source text by the LOLITA core system (see chapter 4). A first identification of possible events to be unified is carried out during the semantic analysis (section 4.2.3) before the new knowledge is stored in the semantic network. A later stage of the analysis examines the recently built pieces of the network and attempts to unify information which meet specific criteria.

7.3 The takeover financial template

This section describes the rules that have been chosen for the takeover template. The analysis has been carried out on a total of 80 takeover-relevant articles, identifying all the possible different "*semantic structures*" associated to the source text.

The *semantic structures* for both the main-event and each of the slot-rules have been coded directly into the financial application and they locate relevant information within the new nodes produced by the analysis of the source text.

7.3.1 The takeover main-event

The identification of the *semantic-structures* for the main-event of the takeover template has been carried out analysing a total of 80 relevant takeover articles. Four different relevant semantic structures have been found for the main-event of the takeover template with different statistical relevance. Some of the semantic structures correspond to domain-specific knowledge.

- **Semantic-Structure 1:**

```
Event 1:
  Action:  acquire | purchase | buy | take-over
  Subject: company | person
  Object:  company
```

This semantic structure covers situations in which the information that makes it possible to recognise a takeover is carried by the *action* of the sentence, thus, the verb. This semantic structure captures different kind of sentences in the financial source texts. The following sentences from the Financial Times belong to this semantic structure:

SHIELD GROUP, the holding company that owns Stickley Kent, the auctioneer and property insolvency specialist, HAS ACQUIRED KAMCO COMPUTER SYSTEMS.

BF GOODRICH, the Ohio-based specialty chemical and aerospace company, IS TO ACQUIRE ROSEMOUNT AEROSPACE from the Emerson Electric Company for 300 million dollars in cash, pending regulatory approval.

WESTCOAST is set to become Canada's biggest gas utility with its proposed ACQUISITION OF UNION ENERGY.

ARCHER HOLDINGS yesterday announced that it would continue its expansion BY ACQUIRING KELLET HOLDINGS

CEMENTOS MEXICANOS is expected to launch a 1.7 BILLION DOLLARS TAKEOVER for Spain's biggest Producer, VALENCIANA DE CEMENTOS.

BANK OF SCOTLAND is poised to complete its TAKEOVER OF THE COUNTRYWIDE BANKING CORPORATION.

ALEXANDRA TOWING has conditionally agreed to a 52 MILLION DOLLARS TAKEOVER BY HOWARD SMITH.

The TAKEOVER OF the east German oil company MINOL by a consortium including the German steelmaker THYSSEN and the French oil company ELF-AQUITAINE has been approved.

- **Semantic-Structure 2:**

```
Event 1:
  Action:  acquire | purchase | buy | take-over
  Subject: company | person
  Object:  majority stake
Event 2:
  Action:  is_part_of
  Subject: majority stake
  Object:  company
```

This structure covers cases in which there is no mention of an acquisition or a takeover. However, it is possible to infer a takeover/acquisition from the fact that acquiring a majority stake in a company implies a takeover. This structure is an example of domain-specific knowledge stored at the application level. Example sentences from articles from the Financial Times which belong to this semantic structure are:

BRAU AND BRUNNER, the German brewery, is paying 4.112 million DM (2.42 million dollars) for a MAJORITY STAKE IN KAMENITZKA.

ARJO WIGGINS APPLETON, the Franco-British paper group, has taken a MAJORITY STAKE IN NITECH, a Polish paper merchant.

- **Semantic-Structure 3:**

```
Event 1:
  Action:  acquire | purchase | buy | take | take-over
  Subject: company | person
  Object:  control
Event 2:
  Action:  relate_
  Subject: control
  Object:  company
```

This semantic structures covers cases in which there is no mention of an acquisition/takeover which can however be inferred from the fact that taking control of a company implies a takeover (domain-specific knowledge). The following sentences, from the Financial times, belong to this semantic structure:

UNILEVER, the Anglo-Dutch food and consumer products group, is set to become France's largest ice cream producer by TAKING CONTROL OF ORTIZ-MIKO.

GALERIES LAFAYETTE, one of France's leading stores groups, IS TAKING CONTROL of Monoprix, the retail chain.

- **Semantic-Structure 4:**

```
Event 1:
  Action:  pay
  Subject: company | person
  object:  monetary value
  goal:    company
```

This semantic structure is another example of domain-specific knowledge. In fact, there is no mention of an acquisition or a takeover in the rule. However, it is possible to infer a possible takeover from the fact that a company paid an amount of money for the other company. The following sentences, from the Financial Times, belong to this semantic structure:

CONRAD, the Manchester-based sports, leisure and consultancy group, has agreed to PAY 1 MILLION POUNDS in cash and shares FOR INTER RESEARCH.

CALDERBURN, the office furniture group, HAS PAID an initial 6.25 MILLION POUNDS FOR SBFI.

The 80 articles in the sample population have been divided and classified according to the semantic structure they belong to and the statistical frequency distribution of the semantic structures are:

- **Semantic-Structure 1 == 75%**
- **Semantic-Structure 2 == 3%**
- **Semantic-Structure 3 == 3%**
- **Semantic-Structure 4 == 19%**

As the percentages suggest, the semantic structures 1 and 4 are the most relevant, while the semantic structures 2 and 3 are almost irrelevant.

Prototypes for the takeover main-event

The *acquisition* prototype has been added to the LOLITA system for correctly processing acquisition and takeover events. The concept of acquisition, in fact, is the *generalisation* of *takeover* (figure 7.1).

The prototype defines an *acquisition* as following (figure 7.3):

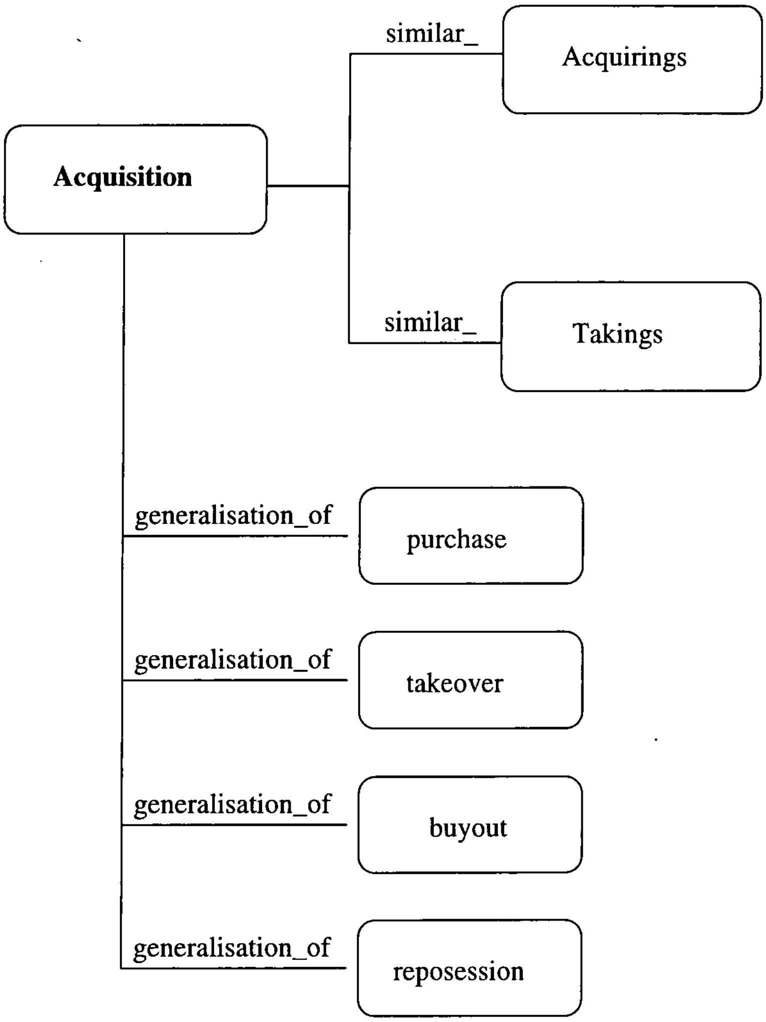

Figure 7.1: The node "acquisition" in the Semantic Network

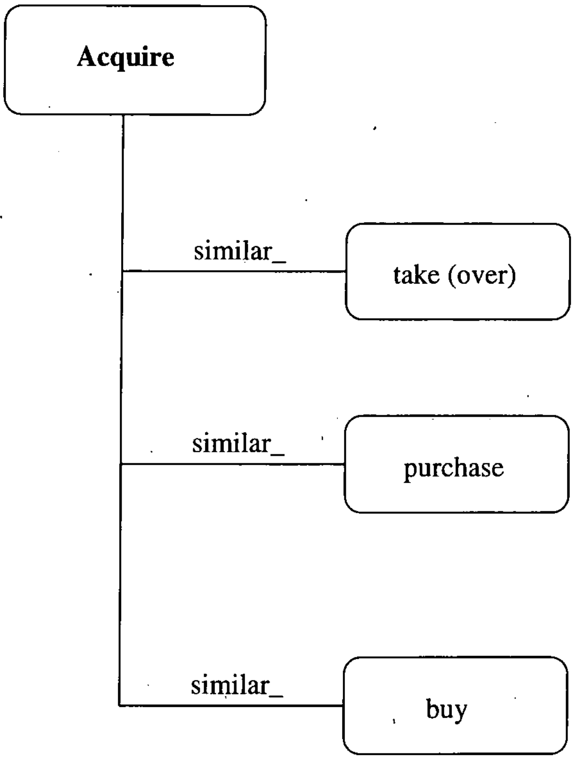

Figure 7.2: The node "acquire" in the Semantic Network

```
Some humans acquire things
```

Any event in which the above occurs is recognised an *acquisition* event. The action chosen for the prototype is "*To acquire*" which is synonym of *purchase*, *take (over)* and *buy* (see figure 7.2). The subject of the acquisition is a set of humans who perform an acquisition. This also includes human-organisations and, therefore, companies, which are stored below the concept of human in the semantic network. The node "some" has been introduced to avoid saying that "all humans acquire things" which would be represented in case the restriction was not included. The object of the acquisition can be anything. The rules of the takeover template, however, restrict the object of the acquisition only to companies.

Domain-specific knowledge for the takeover main-event

Various domain-specific rules have been identified for the takeover main-event:

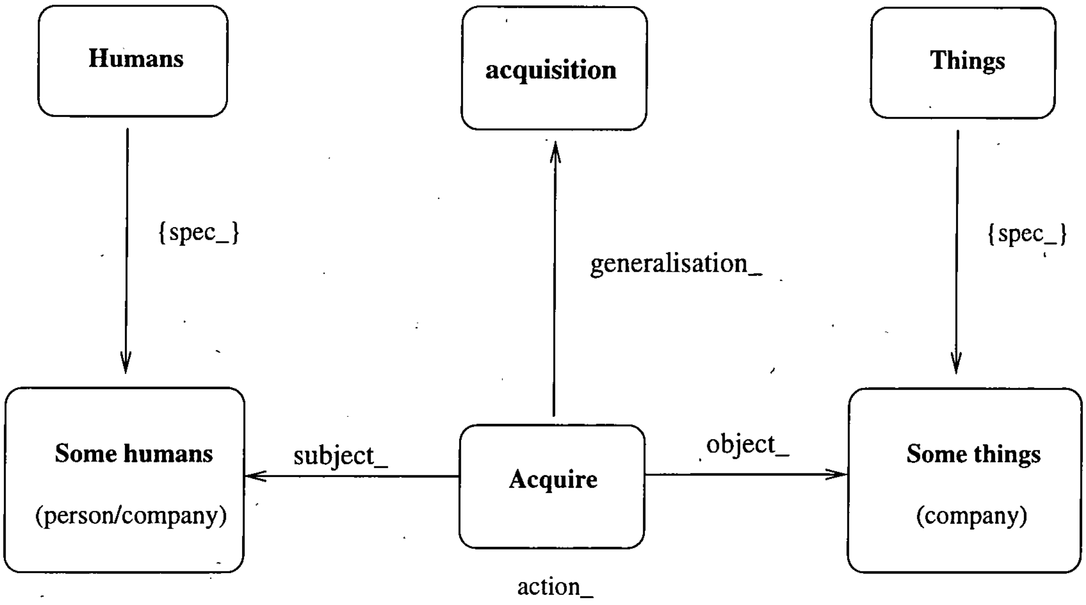

Figure 7.3: The acquisition prototype.

X buys a majority stake in Y, this implies a takeover. This rule is similar to the previous one. Basically, LOLITA wouldn't normally assume that buying a majority stake in a company (although it knows the meaning of "stake") means a takeover. It would only assume that X owns a stake of Y, but nothing else. This rule, instead, implies that a takeover is in place and it is actually equivalent to buying a majority stake. This rule corresponds to the semantic structure number 2.

if X takes control of Y, this implies a takeover. This rules covers the situation in which a company takes full control of another company, without mentioning if this is a takeover. Normally, the system wouldn't classify it as being a takeover, since the concepts of "take control" and "takeover" are not connected. However, in the specific financial context, "taking control" of a company, usually means "taking it over" or "buying a majority stake", which is equivalent. This rule corresponds to the semantic structure number 3.

X pays M for Y and Y is a company, this implies the takeover of Y by X. This case covers cases in which, for example, "IBM pays xx million dollars for Lotus Inc.". LOLITA wouldn't normally understand that IBM bought LOTUS nor that this is a takeover. With this rule, the sentence (and all other sentences with same meaning) is understood to be an acquisition (takeover), by the financial application. This rule corresponds to the semantic structure number 4.

Rule for the takeover main-event

Once the different semantic structures have been identified, the corresponding rules for the takeover main-event can be written. These rules will locate the relevant information within the new knowledge produced by the semantic analysis of the source text. For example, the rule for the semantic structure 1 will locate any event with action corresponding to the appropriate meaning of "*acquire*" (currently corresponding to the node 76275) or any event which has as generalization the appropriate meaning of "*acquisition*" (currently corresponding to the node 17070). The rules are directly coded in the financial application. The rule for the semantic structure 1 described above, for example, will result in the following Haskell code:

```
> isTakeover g nr
>    | is_event g nr &&
>      (has_gen_in takeoverEvents g nr
>       || has_gen_in takeoverActions g action
...
...
>       = True
```

Takeover template post-condition

The additional rule that must be satisfied for the template to be created (template post-condition) is that at least one of the two companies (or person for the company predator) involved in the takeover must be correctly recognized by the system. If no companies are recognized, the possible takeover main-event candidate is discarded by the template code.

7.3.2 The takeover templates slots

This section describes the rules for the various slots associated the takeover template. The identification of the relevant rules has been done in the same way as the main-event. The *semantic structures* for each slots have been identified from the same 80 relevant articles. The rules have been defined taking into account any relevant prototypical information and domain-specific knowledge.

Company Predator

The company predator slot should be filled with the name of the *company* or the *person* which performs the takeover. The rule for the company predator slot is therefore rather straightforward. The slot is filled with the *subject* of the acquisition event, provided that it is either a *company* or a *person.* This rule should catch all possible semantic structures, which are the same identified for the main-event, since the *acquisition prototype* should ensure that the appropriate acquisition event is correctly built for all semantic structures (see figure 7.3).

Company Target

The company target slot should be filled with the name of the *company* acquired by the company predator. The company target slot is filled in a similar way as the company predator. The *company target* is the *object* of the *acquisition event* (figure 7.3) under the condition that it must necessarily be a *company*, since we only consider company takeovers. Therefore others kinds of takeovers are not considered. In the following sentence (from *The Financial Times* on CD-ROM) the takeover event is not relevant for the takeover template, since the object of the event is not a company:

```
BARDER & MARSH, the Lloyd's agency, has reached agreement  in principle with
Wendover Underwriting Agency, to take over the management of five syndicates
administered by Wendover, Richard Lapper writes.
```

the event is referring to a *management takeover* rather than a company and is therefore not relevant.

Type of TAKEOVER

A takeover can be defined as *hostile* when the company target does not want to be acquired, friendly otherwise. Takeovers are generally friendly. In the set of 80 takeovers articles, only two occurrences of *hostile* takeovers were found. In both cases the takeover was hostile because the node *hostile* had been used as *qualification* of the takeover, for example:

MR VIJAY Mallya assaulted the UK drinks distribution market via the hostile takeover of Wiltshire Brewery, the UK quoted drinks company.

The rule for this slots is therefore as follows:

- **HOSTILE** is chosen if the main takeover event is related to the concept *hostile.* This analysis should be produced by the LOLITA core system when *hostile* is used as qualification of the takeover, for example in the following sentence:

  ```
  FIAT acquired RENAULT in a hostile takeover.
  ```

- **FRIENDLY** is chosen in all other cases.

Value of the takeover

The takeover value slots should contain the monetary value that the company predator has paid for acquiring the company target. Three different semantic structures have been identified in the 80 source articles, from which three different rules have been defined.

- **Semantic-Structure 1:** (43% of cases)

```
Event 1:
  action:     acquire | purchase | buy | take-over
  subject:    company | person
  object:     company
  instrument: monetary value
```

In this case the monetary value is directly connected to the main takeover event. The value is the *"instrument"* of the takeover, thus the element that "allows" one company to acquire another company (free acquisitions are not considered here).

The slot rule for this semantic structure will therefore select any nodes which are linked as *"instrument_"* of the main takeover event and are monetary values. The following sentence from the Financial Times belongs to this semantic structure:

Bf Goodrich, the Ohio-based specialty chemical and aerospace company, is to acquire Rosemount Aerospace from the Emerson Electric Company FOR 300 MILLION DOLLARS in cash, pending regulatory approval.

- **Semantic-Structure 2:** (17% of cases)

```
Event 1:
  action:  acquire | purchase | buy | take-over
  subject: company | person
  object:  company
Event 2:
  action:  relate_
  subject: Event 1
  object:  monetary value
```

This semantic structure considers cases in which the monetary value is directly mentioned in the acquisition main event, but is given as "qualification" of the event. The following sentence (from *The Financial Times* on CD-ROM) shows an example of this semantic structure:

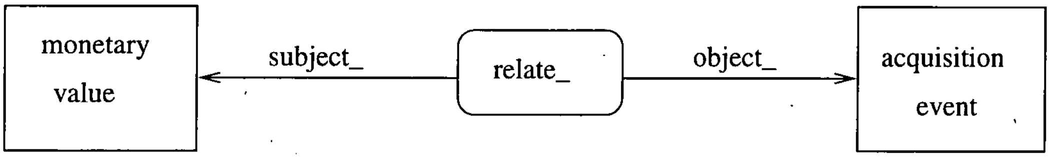

Figure 7.4: The representation of the semantic structure.

CEMENTOS Mexicanos (Cemex), North America and Mexico's largest cement producer, is expected to launch a 1.7 billion dollars takeover for Spain's biggest producer, Valenciana de Cementos.

The slot rule for this semantic structure will look for any nodes which are related (action *relate_*) to the main takeover event and are *monetary values* (figure 7.4).

- **Semantic-Structure 3** (36% of cases)

```
Event 1:
  Action:  pay
  Subject: company | person
  object:  monetary value
  goal:    company
```

In this case the *acquisition* main-event is the *goal_* of the event having as action *to pay*: the first company pays an amount of value *with the goal* of acquiring the second company. The rule for this case will therefore look for any events connected to the main acquisition event with the *goal_of* link. The events connected are checked and the objects of the events with action *pay* (or similar) and that are monetary values are chosen.

- **Semantic-Structure 4:** (2% of cases)

```
Event 1:
  action:  cost
  subject: acquisition event
  object:  monetary value
```

In this case the monetary value is not directly linked to the main event but is stored into a separated event. The rule for this case will look for any event with action *"cost"* (or similar) which have as subject the acquisition main-event and, as object, a monetary value. The slot will be filled with the object.

- **Semantic-Structure 5:** (2% of cases)

```
Event 1:
  action:  is_a
  subject: cost | price
  object:  monetary value
Event 2:
  action:  relate_
  subject: subject of Event 1
  object: acquisition event
```

In this case the slot will be filled with the object (which must be a monetary value) of an event with *is_a* action and subject cost, price (and similar) which is related (*relate_* action) to the main acquisition event.

An important note for this slot concerns with the correct *unification* of the events. The likelihood that the cost of the acquisition is not directly connected to the main event is, in fact, much higher than for other slots (13.21 per cent in the sample population). If the system does not unify correctly the events, the template application will be unable to retrieve the appropriate information. In the following sentence (from *The Financial Times* on CD-ROM), for example, the information regarding the value of the takeover is stored in another event which should be unified by the system.

```
Samsung has acquired a former east German manufacturer of glass for color
TV picture tubes...

...
```

```
Samsung is expected to pay Dollars 30m for the acquisition...
```

Bank adviser Predator

This slot should contain the name of the company / person which is advising the predator in the takeover operation. Only one occurrence of bank adviser has been found in the 80 articles of the sample population. This is probably due to the fact that the big companies are normally advised by the same company for all different kinds of operation and is therefore unlikely that the adviser is chosen in the specific occasion of a takeover. The rule for this slot will choose any company which is an *adviser* (subjects of an event with action *is_a* and object *adviser*) and related to the company predator slot (*relate_* action).

Bank adviser Target

The bank adviser of the target is, in the same way as the predator, the company / person who advises the target during the takeover. Similarly, the slot is defined in the same way as the *Bank adviser Predator*: the slot will be filled with any company which is an *adviser* (subject of an event with action *is_a* and object *adviser* and related to the company target slot (*relate_* action).

Expiry date

This slot should contain the reference to an eventual expiry date of the takeover. The date is usually stated when the takeover has not yet been performed and the company predator is offering a certain amount of money for a fixed amount of time. The company target can then accept or refuse such offer within the expiry date. The expiry date can only refer to the main-event, since for other slot it would not have any sense. Only an occurrence of expiry date in the 80 articles of the sample population has been found.

The rule for the expiry date slot will fill the slot with the object linked with the *time_* link to an event with action *to expire* and subject the *acquisition* main-event.

Takeover Attribution

This slot should contain the person, company, organization who reported, announced or "said something" about the takeover event. The "attribution" information is extremely important for the financial operator. Knowing the source of the information, the operator can judge whether the facts are likely to be true, possible, or just rumours. For being included in the attribution slot, the person or company needs only to say, report or announce something about any of the slots. This is because if a person or company reports something about a particular slot which is related to the main event (e.g. Value of the takeover), it is likely to be that it has also reported the main event.

Two different semantic structures have been identified in the 80 takeover articles of the sample population:

- **Semantic-Structure 1:** (71.42% of cases)

```
Event 1:
  action:  announce | say | report | inform
  subject: company | person
  object:  acquisition event
```

In this case the person / company announces directly the takeover. The attribution event is therefore directly connected to the main event and its recognition is therefore straightforward. The attribution slot will be filled with the subject of any event with action *say*, *announce*, *report* and *inform* and object the *acquisition* main-event. The subject must be either a person or an organization. The following sentence (from *The Financial Times* on CD-ROM) is an example of this semantic structure:

AJ ARCHER HOLDINGS, one of two Lloyd's agency group listed on the Stock Exchange, yesterday announced that it would continue its expansion by acquiring a fellow agency, Kellett Holdings.

- **Semantic-Structure 2:** (28.57% of cases)

```
Event 1:
  action:  announce | say | report | inform
  subject: company | person
  object:  any information in any of the other templates slots
```

In this case, the subject of the acquisition does not announce or report directly the main acquisition event, but other information, consisting in any of the other slots (apart from the slot denial). The template application will therefore *infer* that the attribution slot can be filled with such person. Since there is the possibility that the attribution event is wrong, the full event (sentence) is reported in the slot, rather than the only subject. In this way, the user has the possibility of verifying that the subject is relevant for the acquisition. The rule for the slot will look for any event with action *say*, *announce*, *report*, *inform*, subject organization or person and object any node stored in any other slot.

Current stake predator

This slot should contain the amount or value of the shares already owned by the predator in the company target at the time of the takeover. The implementation of such rule, however, implies that the core system must be able to deal with time information in an extremely complete way. In fact, if the takeover occurred in the past, the system should understand that the ownership of part of the target occurred even in a farer past. Currently, however, the LOLITA core system is not able to deal with time in such a great detail. Therefore, the rule locates any event where the company predator is owner of part of the company target. In the 80 articles of the sample population only one occurrence of this slot appears which suggests that the information is not very common and, therefore, the fact that the temporal information is not checked is likely to be non influent on the slot's fills.

Denial

The slot should contain the name of the person or company who denies any of the information in the template. Two occurrences of denial slots have been identified in the 80 takeover articles of the sample population and are both referring to some slots of the template. The following sentence (from *The Financial Times* on CD-ROM) is an example of denial of the *takeover value* slot.

> *Schroders, the City merchant bank, confirmed yesterday that it is considering taking full control of Wertheim, the Wall Street securities house in which it has held a 50 per cent stake since 1986, writes David Barchard. However, the bank denied that active negotiations for a cash sale had got under way.*

The financial Topic

A specific kind of text, for example financial articles, usually contains specific meanings of words and jargon and the probability that such meanings are used in that context is much higher than in other kinds of texts.

We can therefore define the *topic* of a specific kind of text as the *theme*, the *subject of discourse* or the *probable reasoning* of that article. If, for example, we are reading a *Financial Times* article, we would expect that it will regard information about companies, economics, finance and business and, less likely, information about art, travel etc. If, in this context, we are reading about a takeover, we would expect the meanings of the words *buy* or *acquire* to be:

buy: = To take. , To purchase. -] To acquire;

acquire: = To take. , To purchase. , To buy. -] To acquire.

rather than:

buy: = To corrupt. , To bribe. -] To pay.

acquire: = To produce. , To grow. , To get. , To develop. -] To develop.

Choosing the *topic* of an article is basically making an *assumption* of what the article is more likely to contain and what has to be expected in the text. This helps to disambiguate the meaning of words, for example in cases like the one above.

The *topic* for the financial application includes the appropriate meanings of words and concepts one would expect in a financial context, for example *buy*, *takeover*, etc. The information stored in the *topic* influences the choice of the word meanings. The meanings preferred are those which are semantically closer to the context and the meanings in the topic. The semantic closeness is computed depending on the distance between the nodes in the semantic network. The choice of the appropriate meaning depends also on other factors, such as the system's knowledge of the concept and its frequency of use.

Chapter 8

Design of the user-definable template interface

8.1 Introduction

This chapter presents the solution identified for the project aim 2: the user-definable template interface (see chapter 1). This chapter focuses on the design of the user-definable template interface. The implementation is discussed in chapter 9.

8.2 The user-definable template interface

The partitioning based on the *financial activities* approach should be able, in our view, to identify the information which is relevant for the operator's investment decision-making process. This is because the activities identified with the help and experience of financial experts are those most likely to have a direct impact on the share price, while other events may or may not influence the prices. However, since no direct link can be established between an event and a share price movement, the user might want to personalize the system adding new templates to suit his specific needs. For example, the user might want to add an *organization* template, which describes in detail the information of a company that can be found in a source article.

The LOLITA template user-interface has been designed to allow the user to add new templates to the collection of existing ones using natural language sentences with few restrictions and formal elements (see section 8.2.3) [Costantino *et al.*, 1996e].

The templates are added to the collection of templates already available in the system and the analysis is performed in the same way. This allows the maximum degree of flexibility for the user.

The first step in the definition of the user-definable template interface has been the definition of the interaction way between the user and the system and, in particular, which kind of definitions the user should enter for defining the templates. This corresponds to the definition of the class of objects "*ExtractionNeeds*" described in the TIPSTER phase II document (see section 3.7). Three approaches to the definition of the "*ExtractionNeeds*" of the user have initially been considered:

Menu-based environment. In this case the user constructs the templates using predefined structures / components available in menus and using cut and paste techniques.

Example-based environment. In this case the user would provide the system with a number of examples of relevant articles or of relevant fragments of articles. The system would then extract the relevant patterns which would be used in the extraction of the templates from the source articles. The MUC-6 Hasten system [Krupka, 1995] is an example of a finite-state analysis information extraction system based on user-supplied examples consisting of relevant fragments of the original text (see section 3.7).

Natural Language Text. In this case the user is allowed to enter the full specifications for the main-event and each of the desired slots using sentences in natural language. The system will then translate these information into the appropriate semantic rules.

The design of the LOLITA user-definable template interface has been done assuming the input in natural language text. The other two approaches, in fact, present relevant limitations.

Two main paths could be followed defining a *menu-driven* environment. A first possibility would be to provide very *low-level primitives* which could be employed by the user for the definition of the templates. However, very low-level primitives would make the environment rather complex and the user would need to spend a considerable amount of time for designing the templates. Another possibility would be to provide a *high-level structure* with specific objects already defined (e.g. *company*, *person* etc.). The time needed for entering the templates definition would be considerably lower. However, *high-level structures* would imply limitations to the expressive power of the user.

An *example-based* environment could potentially consist of two different situations. In the first case the user would provide examples of relevant articles for a specific template. The system would then automatically identify the appropriate information. In a second case, the user would provide some examples of relevant text fragments for the specific elements of the template, for example the *Hasten* MUC-6 system [Krupka, 1995]. The user enters a certain amount of example patterns for each template elements which are subsequently used by the system for analyzing the source articles (see section 3.7). However, in a pure example-based environment the user would be required to enter a considerable amount of examples (either source articles or text fragments) which would drastically increase the time needed for the definition of the template elements. For example, the definition of the MUC-6 management scenario template using the *Hasten* system required 132 example patterns [Krupka, 1995].

A user-definable template interface based on natural language sentences should theoretically allow the user for the maximum degree of flexibility, without any significant constraint. No examples are required and the user can simply enter the description of the information to be extracted. However, as we will analyse in more detail in this section, some constraints on the kind and format of the natural language should be practically introduced, to reduce the inevitable *ambiguity* of the input sentences.

The LOLITA user-definable template interface accepts natural language input. The user can supply sentences in free natural language using specific formal elements designed to reduce the *ambiguity* of the input sentences.

The definition of the LOLITA user-definable template interface has been done by analyzing the results of an experiment carried out by seven potential users of the system [Costantino *et al.*, 1997]. The test required the potential users to describe the takeover template using sentences in natural language. More specifically, the users were asked to describe the *condition* under which the template should have been filled (the template *main-condition*) and the specific slots rules. The target of the experiment was to investigate two different kinds of problematics:

- how easy is for the user to define the templates using unconstraint input natural language text;
- how easy would be for the system to make use of such unrestricted input definitions.

The ultimate target was to identify an optimum compromise between the two.

The seven potential users have been divided in two groups. A first group of five users was asked to define the template structure without having access to the financial takeover template definition (see section 6.3.2) allowing the maximum degree of freedom. The template definitions supplied by these users can be seen in figure 8.1.

The correct template definition of the financial takeover template was instead given to the other two potential users (figure 8.2).

Although the test cannot be considered statistically significant because of the small number of tests carried out, the analysis of the results suggests that allowing complete freedom to the user can lead to a difficult situation for both the user and the system:

User No. 1:

Template condition:	one company buying or controlling another
Source company:	the company taking over
Source product:	the market of the source company
Target company:	the company taken over
Target product:	the market of the target company
When:	the date of the takeover
Cost:	the cost of the target company

User No. 2:

Template condition:	one entity makes a bid for a different entity
Predator:	name of the predator
Volume:	volume of takeover in currency unit
Currency:	currency
Type:	friendly/hostile
Start:	when it was announced
End:	when it should happen
Backing:	advisors of predator
Value Target:	value (stock value) of target
Value Predator:	value (stock value) of predator

User No. 3:

Template condition:	A company acquired another company
Company 1:	name of the company acquiring
Company 2:	name of the company which is being taken over
Value:	value of the takeover
Date:	date of the takeover

User No.4:

Template condition:	** no condition supplied by the user **
Type comp 1:	the company's type of business
Type comp 2:	the company's type of business
Type new comp:	the type of business of the new company
Amount bid:	how much the merger cost
Value of joint company:	what is to be the value of the joint company

User No.5:

Template condition:	a company purchases a majority stake in another company
Company 1:	name of the company that is purchasing
Company 2:	name of the company that is purchased
Owners 2:	major shareholders of Company 1
Date of announce:	date in which the merger is announced
Date of merger:	date in which the merger will take place
Market 1:	markets of company 1, products and locations
Market 2:	markets of company 2, products and locations
Previous Stake:	stake owned by company 1 before the takeover
New Stake:	stake owned by company 1 after the takeover
Price:	amount paid for the stake gained
Source:	source of the information

Figure 8.1: The fill-rules of the takeover templates of the first group of users.

User No. 6:

Template Condition: A company or person which buy another company.
COMPANY_TARGET: The company acquired.
COMPANY_PREDATOR: The company or the person which buys company_target.
TYPE_OF_TAKEOVER: HOSTILE: The company_target does not want to be acquired.
FRIENDLY: otherwise
VALUE_TAKEOVER: The price paid by the company_predator for the takeover.
BANK_ADVISER_PREDATOR: The company advising the company_predator.
BANK_ADVISER_TARGET: The company advising the company_target.
EXPIRY_DATE: When the takeover expires.
ATTRIBUTION: The person or organization who announced the takeover.
CURRENT_STAKE_PREDATOR: Which part of company_target was already owned by company_predator before the takeover.
DENIAL: The person or organization who denied the takeover or company_target or company_predator or type_of_takeover or value or bank_adviser_predator or bank_adviser_target or expiry_date or attribution or current_stake_predator

User No. 7:

Template Condition: Did company X or person X buy company Y?
COMPANY_TARGET: Y
COMPANY_PREDATOR: X
TYPE_OF_TAKEOVER: HOSTILE == if Y did not want to be bought.
FRIENDLY == if Y wanted to be bought.
VALUE_TAKEOVER: how much was Y sold for.
BANK_ADVISER_PREDATOR: who advised Y.
BANK_ADVISER_TARGET: who advised X.
EXPIRY_DATE: when does the offer expire?
ATTRIBUTION: who makes these statements?
DENIAL: who denies the takeover?

Figure 8.2: The fill-rules of the takeover templates supplied by the second group of users.

- the user can find it difficult to express the template definitions using unrestricted natural language text and without the support of any formal element;

- the unrestricted user's natural language input can be rather difficult to process for the system and a relevant number of ambiguities can be found in the template definitions. These ambiguities mainly concern with the resolution of *anaphora*. In other words, how to resolve the relations between objects and events in the template-condition and in the slots (coreference resolution).

A relevant number of ambiguities in the templates defined by the first group of users (figure 8.1) have been found. The first user defined a takeover template as follows:

Template Condition: one company buying another or controlling another
Slot: When: the date of the takeover

two sort of ambiguities can be found in the two definitions. In the first place, the template condition is not a full sentence but a Noun Phrase.

Secondly, the definition of the slot *"when"* is based on the concept of *takeover*. However, from the template main condition, the system would normally be unable to *infer* the concept of *takeover* or *acquisition* unless domain-specific knowledge is available in the system.

Differently, the second user defined the takeover template as follows:

Template Condition: one entity makes a bid for a different entity
Slot: Predator: name of the predator
Slot: Volume: volume of takeover in currency unit

The slot *predator* introduces an ambiguity. LOLITA might be unable to *infer* that a *predator* is the *"entity"* which *makes the bid.* This would be difficult because the two sentences are totally separated. Similarly, LOLITA would be unable to *infer* that *an entity making a bid for another entity* is, in fact, a takeover, even making use of the *acquisition prototype* earlier described (see section 7.3.1). The coreference of such concepts would be possible in case domain-specific knowledge was available in the system.

A numerous amount of additional ambiguities can be found in the text of the first group of users (figure 8.1). Other examples include the case of a user who missed the indication of the *main-event* and another introduced a *"program-like"* syntax in the definition of a slot.

The definition provided by the second group of users provides some interesting points of analysis. The sixth user makes use of the slot names to refer to information which has already been defined in other slots. Differently, the seventh user defines the fill rules using questions rather than sentences. This user makes also use of variables to define the fill rules. This approach shows a marked decrease in the number of ambiguities of the definition. The seventh user defines the template condition and the company predator slot as follows:

Template Condition: Did company X or person X buy company Y?

Slot: Company_Predator: X

The use of the variable "*X*" eliminates any possibility of ambiguity. "*X*", in fact, can only be the person or the company which did buy company "*Y*". The definition is therefore much easier to understand for the system than, for example, the definition given by the second user. The definition is also much clearer for the user, who does not need to redefine for every slot a concept which, in fact, has already been defined in the main condition.

After having analyzed the definition supplied by the potential users two independent choices could have been taken:

- creating a dialogue-like environment, in which the system prompts the user and asks any information which is needed to resolve the ambiguities. Using this approach the total time needed by the user to complete the definition of a template would drastically increase, providing better accuracy;
- defining a set of restrictions and formal-elements which do not reduce the user's expressive power for defining the fill rules but reduces the amount of possible ambiguities in the definitions.

The LOLITA user-definable template interface is based on the introduction of specific *formal-elements* which has been chosen instead of a specific dialogue environment. This is because the results of the experiment have shown that the users made use of specific notations and references (e.g. the use of the variable *'X'*) which can help reduce the ambiguity of the source definitions and, therefore, a dialogue enviroment is not required.

8.2.1 The template interface environment

The introduction of the specific formal elements was chosen for the following reasons:

- it is **simple**. The user can easily understand the formal elements introduced (e.g. variables) and is able to make immediate use of them;
- the formal elements are **clearly defined**. No ambiguities arise in the use of the formal elements. The system can therefore easier process the user template definitions;
- the formal elements are **suited to the template definitions**. The introduction of the formal elements drastically reduces the possible ambiguities arising from the template definitions. Ambiguities from coreference between different objects and events in the template definitions are sensibly reduced.

The definition of the formal elements has been done following two main targets, allowing the maximum degree of expressive power for the user and formalize the *anaphora* in the user's template definitions.

The user must have the same expressive power as using free natural language text. The user must not be constraint within predefined schemas such as predefined slot types (e.g. companies etc.), meaning of words etc. The introduction of formal elements does not limit such freedom.

The formal elements *formalize* the anaphora that, otherwise, would be *ambiguous* for the system. In other words, the formal elements formalize what the users were already doing, without introducing any constraint. The formal elements have been designed to eliminate three different kinds of anaphora that could be ambiguous for the system in the slot definitions:

- ambiguities from references to elements in the main event (e.g. the definition of the slot "*predator*" by the second user) have been eliminated with the introduction of *variables*;

- ambiguities arising from references to the template condition as a whole (e.g. the definition of the slot "*volume*" by the second user) have been eliminated with the introduction of a formal element corresponding to the name of the template;

- ambiguities arising from references to the contents of other slots which have already been defined have been eliminated with the introduction of formal elements corresponding to the slot-names.

8.2.2 The template structure

The template structure for the user-definable template interface comprises the following elements (see figure 8.3):

- the **template-name** which is the name of the template and distinguishes it among the other templates in the system;

- the **variables** which are used in the definition of the main conditions and in the slot rules;

- the **main-condition** which represents the condition under which the template must be filled by the system and can be composed by multiple conditions;

- the **slot-names** which identify the specific template's slots;

- the **slot-rules** which identify the condition and the contents of a specific slot:

The user must firstly define the name of the template which will be later used to refer to the template as a whole. For example, the user may enter:

```
Template-Name:          T=TAKEOVER
```

Subsequently, the user must define the variables which will be used in the template main condition. For example, the user may define the two following variables:

```
Template-Name:               TEMPLATE_NAME
Variable definitions:        VARIABLE *
Template-main-condition:     TEMPLATE_CONDITION *

Slot-Name *:                   SLOT_NAME
Slot-Rule *:                   SLOT_RULE *
```

TEMPLATE_NAME: String
VARIABLE: Variable-condition
TEMPLATE_CONDITION: Main-condition
SLOT_NAME: String
SLOT_RULE: Slot-condition

Figure 8.3: The definition of a LOLITA template [BNF notation].

```
V=COMPANY1 is a company.

V=COMPANY2 is a company.
```

which can be subsequently used in the definition of the template main condition.

The template condition is the condition under which the template must be produced by the system[1] (see section 5.4.1). The user can enter any condition using a natural language statement. In the definition of the main condition, the user can make use of the variables previously defined, which allow to refer to the objects contained in the main-event during the definition of the slot rules. For example, the user may enter the following main-event which makes use of the variables previously defined:

```
V=COMPANY1 acquired V=COMPANY2.
```

Finally, the user must define the rules for each of the slots. In the definition of the slot rules, the user can refer to objects in the main-event by using the variables previously defined. For example, the user may want to define the following slot which refers to information defined in the main-event:

[1] We here consider concept-based templates. Summary templates are in fact more similar to named-entity tasks and, therefore, are not considered as important for the definition of the user-definable interface.

```
S=BANK-ADVISER-PREDATOR:              The adviser of V=COMPANY1.
```

The introduction of the variable *V=COMPANY1* allows the user to refer back to the appropriate information in the main event without any ambiguity. An equivalent definition of the slot without the use of the variables could have been:

```
S=BANK-ADVISER-PREDATOR:              The adviser of the company that
                                      acquires the other company
```

As it is clearly possible to see, the use of variables simplifies the input of the user and reduces the possible ambiguities. The system, in fact, may find it difficult to understand that the *company that acquires the other company* is, in reality, the subject of the main-event. Therefore, both the user and the system benefit directly from the introduction of the variables, which do not restrict the user's freedom in defining the templates.

The user can refer to the template condition as a whole by using the name of the template as a reference to the main-condition. The name of the template acts therefore as a variable which is automatically defined by the interface. For example, the following slot refers to the main-event as a whole:

```
S=VALUE-TAKEOVER                      The cost of T=TAKEOVER
```

Finally, the user can refer to the contents of slots which have already been defined by using the slot name. For example the user may define the following slot:

```
S=PRED-ADVISER-APPOINTMENT-DATE:      The date of appointment of
                                      S=BANK-ADVISER-PREDATOR
```

Each slot-rule defined by the user must include at least one reference to any of the formal-elements, template-name, variables or other slot-names. In figure 8.4 the subsequent stages of the definition of a template by the user are shown.

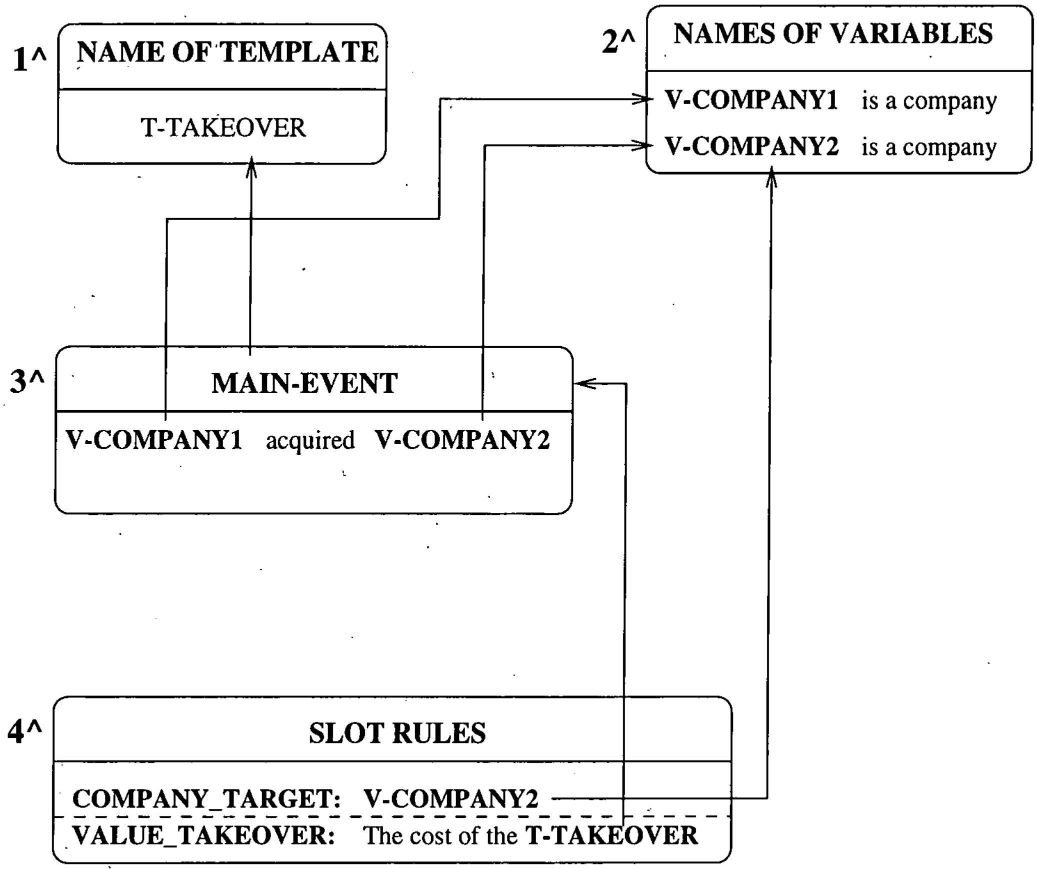

Figure 8.4: The stages of the definition of a template.

8.2.3 The fill rules

Each of the template elements described in the previous section must be defined by the user using natural language statements. These statements can be compared, in a way, with the example patterns of example-based systems such as the Hasten system [Krupka, 1995]. However, the LOLITA user-definable interface statements are different from example patterns in the sense that they refer directly to the meaning of the text. The patterns describe the *meaning*, not the *form* of the events or objects and, therefore, do not necessarily have to correspond to specific fragments of the source texts. For this reason, the number of statements that the user needs to input for defining a specific template is much lower than in pattern-matching example systems. For example, the user may enter the following definition for the takeover main-condition:

```
Template condition:         V=COMPANY1 acquired V=COMPANY2
```

This covers all other cases which share the same meaning, for example:

```
Template condition:         V=COMPANY1 bought    V=COMPANY2
                            V=COMPANY1 purchased V=COMPANY2
                            ...
```

Differently, in a example-based pattern-matching system the user would be required to enter many of the possible real cases and, therefore, the user should refer to fragments of real source texts.

The user must enter the statements for the definition of the template elements according to specific fill rules which are described in the next sections.

The natural language input text

The interface accepts two different kinds of Natural Language input: *sentences* and *noun phrases*.

Three kinds of sentences are relevant for the definition of the templates: *True/False assertions*, *imperative sentences* (commands) and *questions.* The interface accepts only *True-False* assertions. Questions or imperative sentences are not allowed. The following statement is a True/False assertion:

```
V=COMPANY1 acquired V=COMPANY2
```

The user could potentially express the same information using a question, for example:

```
Did V=COMPANY1 acquire V=COMPANY2?
```

To avoid possible ambiguities the user is not allowed to enter questions or imperative sentences, but only True/False assertions[2].

Noun phrases are groups of words which describe particular objects or events but do not form a complete sentence. For example, the following phrase is a *noun phrase*:

```
The company that acquired the other company.
One company buying or controlling another company.
```

Noun phrases are useful to describe specific objects, which cannot be done using *True-False* assertion. The user-definable interface accepts therefore noun phrases as input when the description of a particular object has to be entered.

The name of the template

The user must define the name of the template which can be any sequence of capital letters or numbers according to the following rules:

- the name must start with the string "T=", for example: "*T=TAKEOVER*", "*T=MERGER*" etc.

[2] This possibility may be added in the future.

- the name must be a single word. If more words are necessary, they must be joined with the character "-", for example: "*T=MARKET-MOVEMENT*".

The name of the template can be used in the definition of the slots to refer to the template main-concept.

The variables

The user must define variables for referring to information contained in the main-event in the definition of the slots. The variables must have a name which must follow the following rules:

- the name must be entered in capital letters and starting with the string "V=", for example: "*V=COMPANY1*", "*V=VALUE*" etc.
- the name must be a single word. If more words are needed, they must be joined with the character "-", for example: "*V=COMPANY-ONE*".

The user must define the variables (give a type) using *True-False* assertions, *noun phrases* are not allowed. A legal definition of a variable can be, for example, the following assertion:

```
V=COMPANY1 is a company.
```

The definition of the variables must be entered in the form of a *True-False* assertion. Noun phrases are not allowed.

During the definition of the slot rules, the user can enter the variable to refer to information contained in the main-event under the condition that the variables have been appropriately used in the definition of the main-event.

The main-event condition

The user must enter the main-event condition which is used by the template application to decide when the template has to be created.

The main-template condition must be entered in the form of a *True-False assertion.* Noun phrases are not allowed unless they describe an event. Variables which have been previously defined can be used in the main-event and can be subsequently employed in the slot rules definitions to refer to specific information in the main-event. A legal main-event condition can be, for example:

```
V=COMPANY1 merges with V=COMPANY2 creating V=COMPANY3.
```

A noun phrase cannot be used for the definition of the main-event. For example, the following main-event is not legal:

```
The V=COMPANY1 that merges with V=COMPANY2
```

A template condition can depend upon various main-events. These events can be either *alternative* (OR) or *conjunctive* events. In the first case, the template will be built if one of the conditions is *True* while, for *conjunctive* conditions, all conditions must be *True.* The user can enter alternative events simply by entering them on separate lines, for example:

```
V=COMPANY1 acquired V=COMPANY2.
V=COMPANY1 merges with V=COMPANY2.
```

Conjunctive events, instead, must be entered on the same line using the keyword "*AND*" (in capital letters) between the conditions, for example:

```
V=COMPANY1 merges with V=COMPANY2  AND  V=COMPANY3 is created.
```

The main-event condition is automatically linked to the name of the template and the user can refer to the main-event as a whole by entering the name of the template.

Slots and slot rules

The user must define each of the slots of the template by entering the name of the slot and the associated slot-fill rule. The slot name must be chosen according to the following rules:

- the name must be entered in capital letters and starting with the string "S=", for example: "*S=COMPANY1*", "*S=ATTRIBUTION*" etc.
- no spaces are allowed. If more words are needed, they must be joined with the character "-", for example: "*S=BANK-ADVISER-PREDATOR*".

Slot rules must refer to at least one formal element which refers to information previously defined, the name of the template, the variables and slot-names of slots according to the following syntax:

- the name of the template can be cited when the user needs to refer to the main-concept as a whole. For example, the user may define the following slot:

  ```
  S=VALUE-TAKEOVER:          the cost of T=TAKEOVER
  ```

- the specific variables previously defined and used in the definition of the template main-event can be cited for referring directly to specific information, for example:

  ```
  S=COMPANY_TARGET:          V=COMPANY2
  ```

- The user can refer to the contents of the slots which have already been defined by citing the slot name. For example, the user may want to define a slot which refers to the "*COMPANY_TARGET*" slot which has already been defined and, in this case, can alternatively employ the "*V=COMPANY2*" variable or the "*COMPANY_TARGET*" slot name:

  ```
  S=BANK-ADVISER-PREDATOR:   The adviser of S=COMPANY-TARGET
  ```

The user can define two different kinds of slot: *concept* and *string* slots (see section 5.3.1):

- **concept slots** are defined by entering a *noun phrase* describing the concept of the slot. *True-False* assertions are not allowed. A legal definition of a concept slot is, for example:

  ```
  S=CURRENT-STAKE-PREDATOR:   The stake that V=COMPANY1 owns of
                              V=COMPANY2.
  ```

- **string slots** are defined by entering the required fill-string and a *True-False* assertion representing the slot-fill condition, *noun phrases* are not allowed. A legal definition of a string slot is, for example:

  ```
  S=TYPE-OF-TAKEOVER:
  String-fill:     HOSTILE      T=TAKEOVER is hostile.
  String-fill:     FRIENDLY     otherwise.
  ```

  The keyword "*otherwise*" can be used to specify a default value to be used when all other conditions fail.

Multiple slot definitions can be entered for both concept and string slots. These can act as *alternative* (OR) or *conjunctive* definitions. In the first case, the slot will be filled if one of the conditions is satisfied (string slots) or a relevant concept is found (concept slots). In the second case, all conditions must be satisfied. The user can enter alternative slot-rules by placing the sentences on separate lines. *Conjunctive* rules can be created entering the conditions on the same line and using the operator "*AND*" (in capital letters).

In figure 8.5 the definition of the takeover template using the user-definable interface is shown.

```
Template-name:          T=TAKEOVER

Variables:              V=COMPANY1 is a company.
                        V=COMPANY2 is a company.
                        V=VALUE is money.

Template main-event:    V=COMPANY1 acquired V=COMPANY2.
                        V=COMPANY1 acquired V=COMPANY2 with V=VALUE.
                        The acquisition of V=COMPANY2 by V=COMPANY1.
                        The V=VALUE acquisition of V=COMPANY2 by V=COMPANY1.
                        V=COMPANY1 paid V=VALUE for V=COMPANY2.
                        V=COMPANY1 acquired a majority stake in V=COMPANY2.
                        V=COMPANY1 took full control of V=COMPANY2.

Definition of slots:

S=COMPANY-PREDATOR:     V=COMPANY1

S=COMPANY-TARGET:       V=COMPANY2

S=TYPE-OF-TAKEOVER:
  String-fill: HOSTILE   T=TAKEOVER is hostile.
  String-fill: FRIENDLY  T=TAKEOVER is not hostile.

S=VALUE-OF-TAKEOVER:    The cost of T=TAKEOVER.
                        V=VALUE

S=BANK-ADVISER-PRED:    The adviser of V=COMPANY1.

S=BANK-ADVISER-TARG:    The adviser of V=COMPANY2.

S=EXPIRY-DATE:          The date of expiry of T=TAKEOVER.

S=ATTRIBUTION:          The person or the company that announced T=TAKEOVER

                        The person or the company who said something about
                        T=TAKEOVER or said something about S=COMPANY-PREDATOR
                        or said something about S=COMPANY-TARGET or said
                        something about S=TYPE-OF-TAKEOVER or said something
                        about S=VALUE-OF-TAKEOVER or said something about
                        S=BANK-ADVISER-PRED or said something about
                        S=BANK-ADVISER-TARG or said something about EXPIRY-DATE

S=CURRENT-STAKE-PRED:   The stake that V=COMPANY1 owns of V=COMPANY2

S=DENIAL:               The person or company who denied T=TAKEOVER or
                        denied COMPANY-PREDATOR or denied the
                        COMPANY-TARGET or denied TYPE-OF-TAKEOVER or
                        denied S=BANK_ADVISER-PRED or denied
                        S=BANK-ADVISER-TARG or denied S=VALUE-TAKEOVER
                        or denied EXPIRY-DATE.
```

Figure 8.5: The takeover template as defined for the template user-interface.

Chapter 9

Implementation of the user-definable template interface

9.1 Introduction

In this chapter we will present the implementation of the user-definable template interface described in chapter 8 within the LOLITA system.

9.2 The user-definable template interface

The user-definable inferface described in section 8.2 has been designed to allow the user to enter new templates definitions in addition to the financial templates defined according to the "financial activities approach". The templates definitions are entered by the user by editing a template-definition file such as the one shown in figure 8.5.

9.2.1 Processing the user's template definitions

Once the templates definitions ("*ExtractionNeeds*") have been entered by the user, these are processed by the user-definable interface and stored in an appropriate

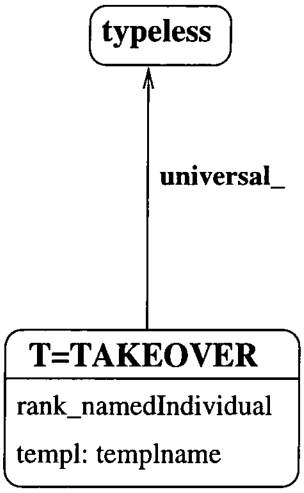

Figure 9.1: The representation of the template-name

format in the semantic network. This corresponds to the operation "*Customise (ExtractionNeed)*" of the TIPSTER phase II document (see section 3.7), since the "*ExtractionNeeds*" of the user are transformed into a "*CustomisedExtractionSystem*". However, the process differs from that described in the TIPSTER phase II document because there is no interaction with the user. This is because the possible ambiguities of the source definitions (the "*ExtractionNeed*") have already been resolved and there is no need for further interaction with the user.

A node corresponding to the name of the template, for example "*T=TAKEOVER*" is firstly created. The node is connected, through a *universal_* link to the node *typeless* and is given *rank_namedIndividual.* The control value "*templ: templname*" is associated to the node. In figure 9.1 the representation of the node corresponding to the template-name "*T=TAKEOVER*" is shown.

The control value "*templ*" has been introduced to identify in the semantic network concepts which correspond to the *formal elements* of the template user interface and can assume the following values:

```
templ: templname
templ: templvariable
templ: templslot
```

The control *templ: templname* is used to identify the template-name, which can be used in the definition of other slots. The control "*templ: templvariable*" identifies the nodes corresponding to *variables*. Finally, "*templ: templslot*" is used to identify the slot-names. The node corresponding to the *template-name* is stored in the *topic* for the current analysis. This will enable the core system to correctly coreference slot-rules refererring to the template name to the appropriate node.

The definitions of the variables (see section 8.2.3) are then processed. Since the definitions of the variables are entered using full sentences, they can be directly processed by the system without any modification or normalization. For example, the definition of the variables:

```
V=COMPANY1 is a company
```

and

```
V=COMPANY2 is a company
```

are processed by the system obtaining an event where the variable "*V=COMPANY1*" and "*V=COMPANY2*" are the subject of two different events with action "*is_a*" and object the generic concept of *company*.

After the definition of the variable has been processed, the control "*templ: templvariable*" is assigned to the variable. The variable is also marked with rank "*rank_namedIndividual*", while the family is assigned automatically by the system during the analysis. The representation for the variable "*V=COMPANY1 is a company*" is shown in figure 9.2.

The template condition definitions are subsequently analyzed by the system. Similarly to the definitions of variables, the main-events are entered by the user in the form of full sentences (see section 8.2.3). The definitions are therefore processed without any modification or normalization. If the main event contains variables, these are directly linked to the variables definition already processed by the system. The definitions are marked as *questions* by the interface since they will represent the questions which will be passed to the inference engine.

The analysis of a main-event such as:

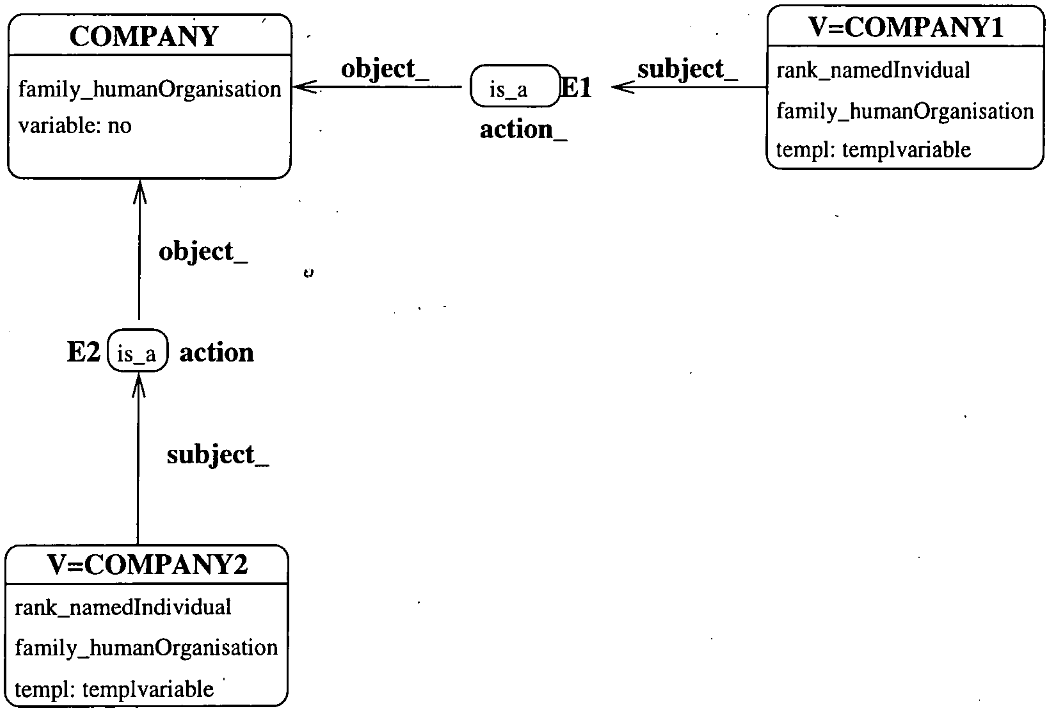

Figure 9.2: The processing of a user-defined variable

```
V=COMPANY1 acquired V=COMPANY2
```

will produce a new event with action *acquire*, subject *V=COMPANY1* and object *V=COMPANY2* and *status_* "*wh_question*" (figure 9.3).

Multiple main-events corresponding to the same template condition are similarly processed and can refer to the same variables.

Each of the noderefs corresponding to the main-event is stored in the *TemplateRule* data structure of the *Template* data structure. A new type of rule has been added to the existing ones, called *TemNetUserDefin* which is defined as follows:

```
> data TemplateRule = TemNetUserDefin MainEvent
> data MainEvent = OrME [MainEvent] | AndME [MainEvent] | ME [Noderef]
```

Alternative main-events (e.g. those for the takeover template in figure 8.5) are stored using "*OrME*", while conjunctive main-events are stored using "*AndME*".

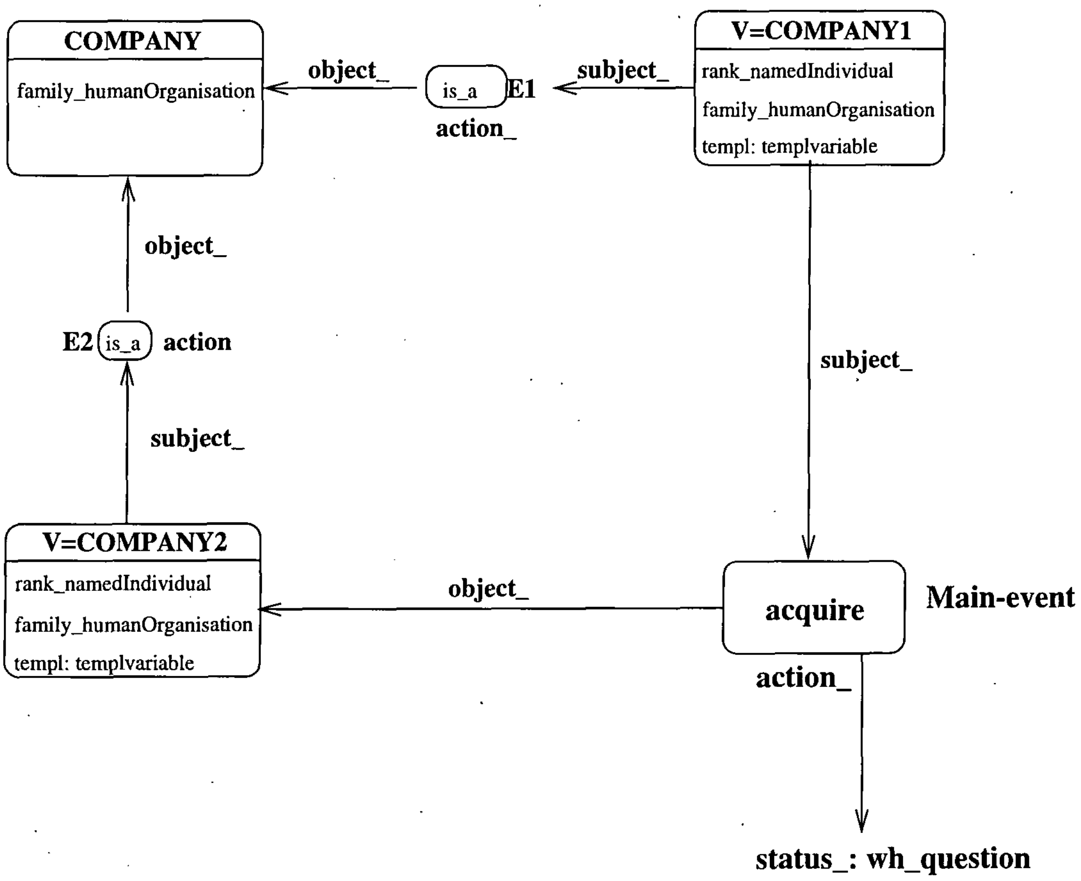

Figure 9.3: The processing of the main-event

This will allow the inference engine to correctly process the two different kinds of main-events (see section 8.2.3).

The final step is the processing of *slot-names* and *slot-rules*. The slot names are directly stored in the template data structure.

The slot-rules are instead analysed by the LOLITA core. The LOLITA parser is used in *noun-phrase* mode, which allows the correct analysis of the slot-rules definitions, consisting of *noun-phrases* (see section 8.2.3). A new special markup has been introduced to identify the text which has to be processed by the parser in "*noun-phrase mode*". The markup follows the MUC-6 SGML markup format and is:

```
<NOUNPHRASE> to mark the beginning of the noun-phrase text
</NOUNPHRASE> to mark the end of the noun-phrase text
```

The markups are automatically added by the interface to the slot-rule definitions. The grammar has been updated with a specific rule for processing the text as noun-phrase:

```
>       xx_nounphraseT &^
>               (jointermphrase +++ quest_typesentence)
>                                               >>> "nounphrase"
```

After the slot-rule definition is processed as a *noun-phrase* a new event is connected to the event corresponding to the slot-definitions. The event has *action* "*is_a*", *subject* the slot-rule definition and *object* the name of the slot, which is marked with the control value "*templ: templslot*". The name of the slot is connected, through a *universal_* link to the node *typeless* and is given *rank_namedIndividual.*

The node corresponding to the *slot-name* is stored in the *topic* for the current analysis. This will enable the core system to correctly unify slot-rules refererring to the slot name to the appropriate node.

The noderef corresponding to the new slot-rule event is finally stored in the *SlotRule* data structure which includes:

```
> data SlotRule =
>     ....
>      SlotNetUserDefin [Noderef]
>     ....
```

Another new event is also added to the slot-rule definition and, in particular to the "*main-concept*" of the slot-rule definition. The "main-concept" is the concept which will have to be identified by the inference rules. For example, in the slot

```
S=BANK-ADVISER-PREDATOR: The adviser of V=COMPANY1
```

the "*main-concept*" of the slot-rule is "*the adviser*". The main-concept will be the object_ of a new event with subject_ the node "*Wh*" and status "*wh_question*". The node "Wh" has been added to the semantic network to specifically identify events whose target is to represent a question for the inference system. The new node will represent the *question* which will be asked to the inference system to return the nodes corresponding to the "*main-concept*" of the slot-rule.

All three kinds of slot-rules definitions described in section 8.2.3: slots referring to variables, slots referring to the template-name and slots referring to other slot-names, are processed in the same way.

Figure 9.4 shows the representation of the slot

```
S=COMPANY-PREDATOR:          V=COMPANY1
```

The slot refers directly to a specific variable. The "*is_a*" event connected with the variable has been added by the user-definable interface and the reference corresponding is stored in the SlotRule data structure. The "*wh_question*" event is attached to the "*main-concept*" of the slot-rule, in this case "*V=COMPANY1*".

Figure 9.5 shows the representation of the slot

```
S=VALUE-TAKEOVER:            The cost of the T=TAKEOVER
```

which refers to the template name. Also in this case, the analysis of the slot-rule has been completed with the "*slot_rule*" event and the "*wh_question*" event.

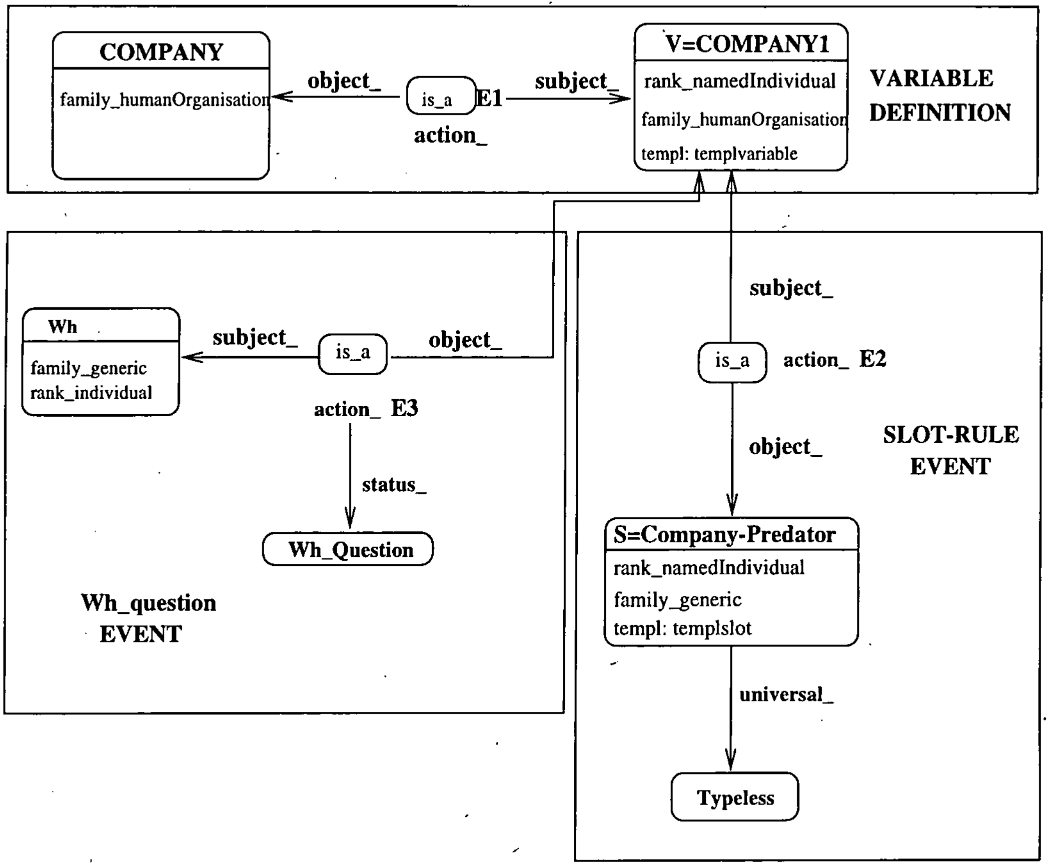

Figure 9.4: The representation of a slot-rule which refers to a variable

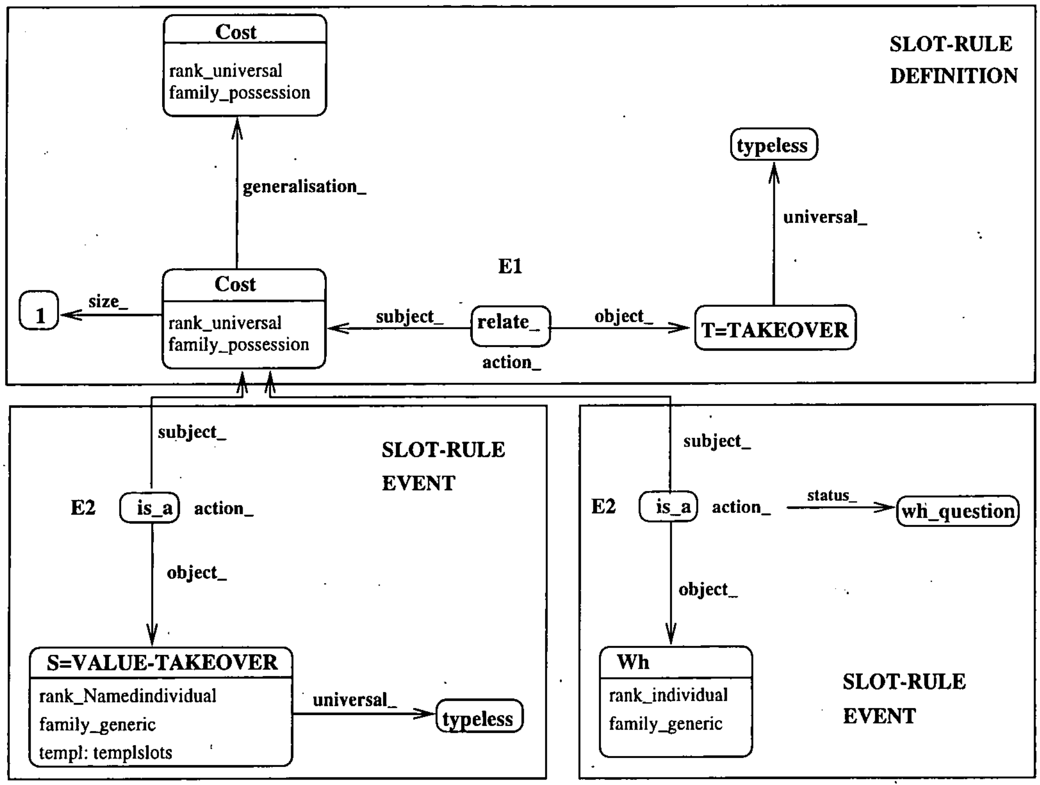

Figure 9.5: The representation of a slot-rule which refers to the template name

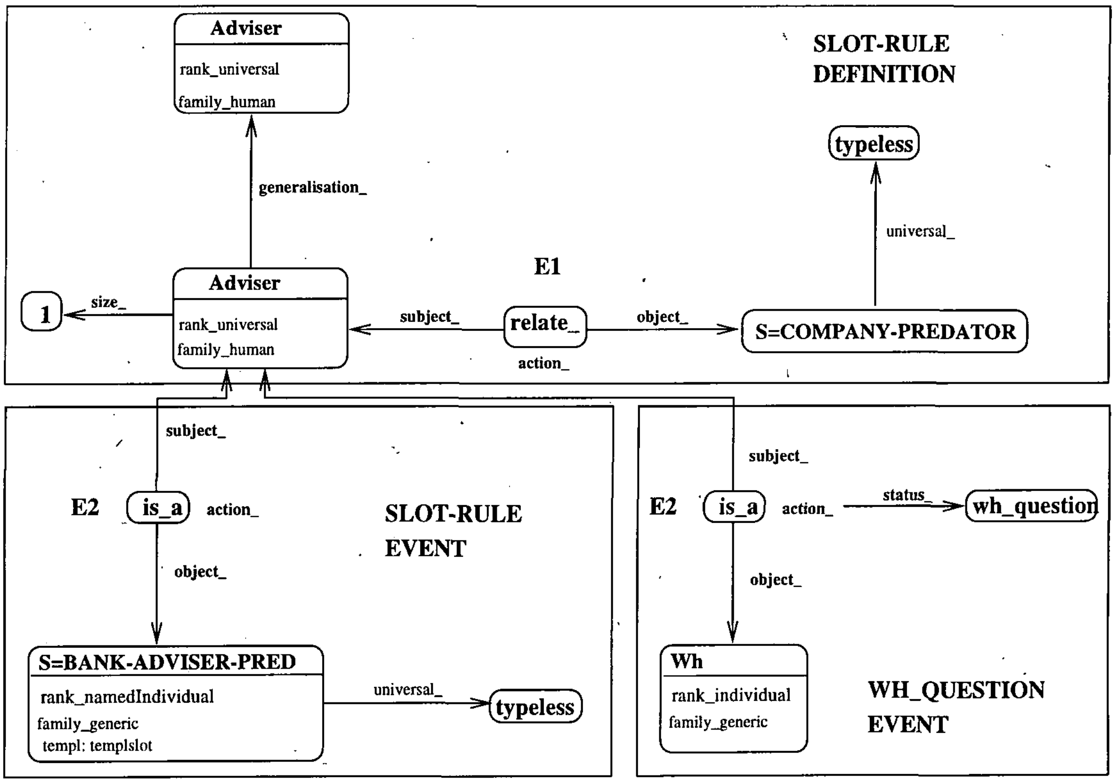

Figure 9.6: The representation of a slot-rule which refers to other slots

Finally, figure 9.6 shows the representation of the slot

```
S=BANK-ADVISER-PREDATOR:   The adviser of S=COMPANY-PREDATOR
```

which refers to another slot previously defined. As for the other two cases a "*slot_rule* event and "*wh_question*" event have been added.

9.2.2 Producing templates using the inference system

The way in which templates are filled by the user-definable template interface is totally different from how information extraction has been previously performed in the LOLITA system.

First of all, no code describing the template rules are available in the system. The LOLITA traditional approach consisted in storing these rules directly in the core system as part of the source code. The new solution allows a major increase in the flexibility and portability of the information extraction application. Since no direct template rules are available in the LOLITA code, the user-definable interface fills the template using the *inference engine* (see section 4.2.4).

The *inference engine* is required to identify entities and events which satisfy the template rules stored in the semantic network corresponding to the *variables* the template *main-events* and the *slot-rules.*

The process of filling the templates using the inference engine corresponds to the operation "*Extract*" of the class of objects "*CustomisedExtractionSystem*" of the TIPSTER phase II document (see section 3.7). The collection of source documents is in fact matched against the templates definitions entered by the user, the "*Customised(ExtractionNeed)*".

Inference and the main-events

The first condition to be analyzed by the inference system is the set of *main-events* stored in the TemNetUserDefin data structure. Each of the main-events, which have already been marked as *wh_question* by the interface (e.g. the event in figure 9.3), is passed as question to the inference engine.

The inference functions will look for any event which matches the given main-event, and, in that case, the event will be a candidate event for filling the templates slots. The inference functions will also identify the relevant nodes which can be matched against the variables which will be used for filling slots which refer to variables. For example, for the main-event shown in figure 9.3:

```
V=COMPANY1 acquired V=COMPANY2
```

the inference engine will recognize that an event such as:

```
FIAT acquired RENAULT
```

is a relevant one, because of the fact that the action is the same and the subject and object can be matched against the variables V=COMPANY1 and V=COMPANY2. In figure 9.7 the representation of the main-event and the candidate event "FIAT

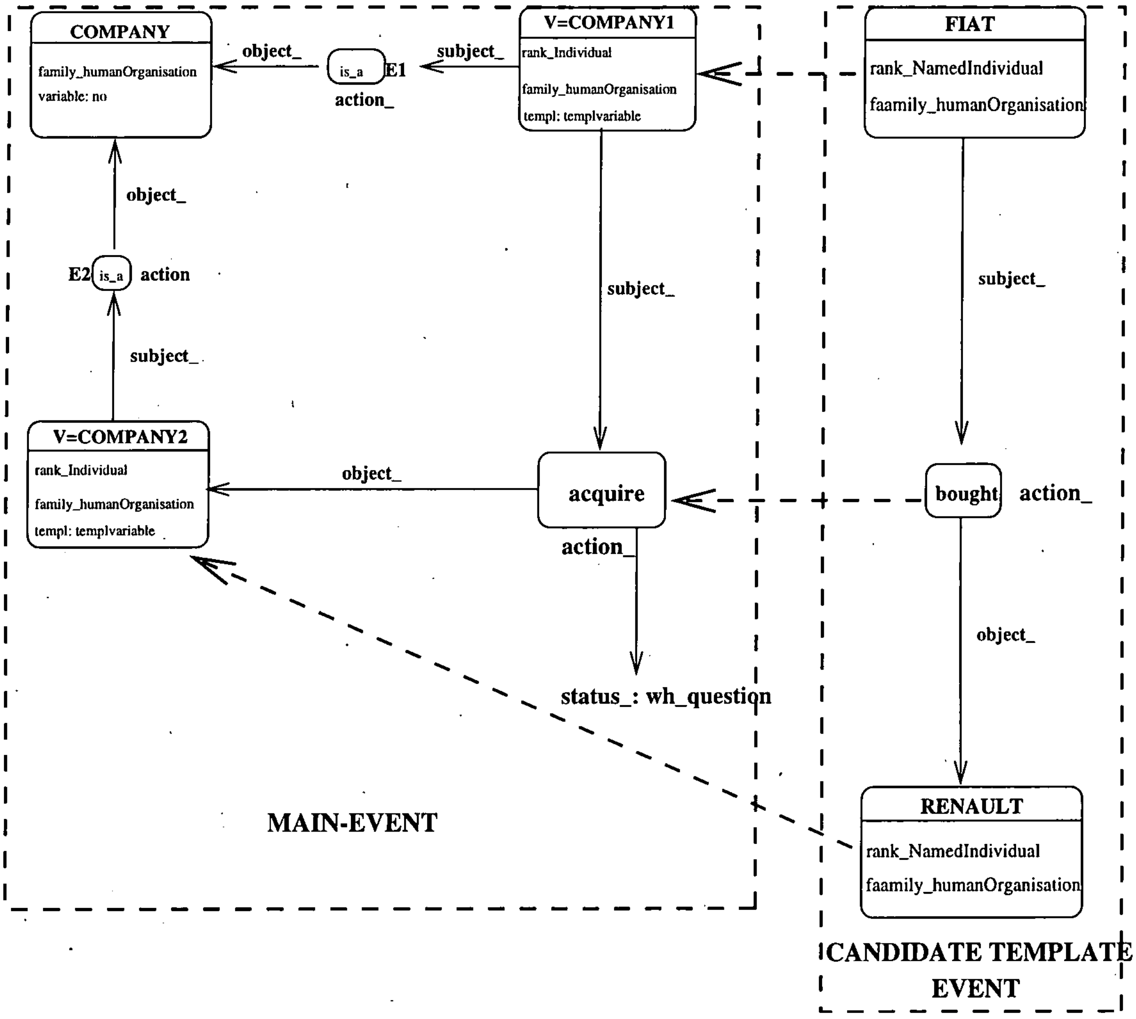

Figure 9.7: Identification of candidate main-events by the inference system.

bought RENAULT" is shown. The inference system tries to match each of the components of the candidate event onto the main-event.

The inference engine will therefore look for an event which satisfies the following condition:

$\exists$ *V=COMPANY1, V=COMPANY2. Acquire(V=COMPANY1,V=COMPANY2)*

A new inference rule has been added to the inference system for handling the identification of the main-event variables. A main event variable is matched against a corresponding concept in a candidate event if the *premis* is below the *question* in the hierarchy:

```
>   | (isTemplVariable q) &&  p <= universal q
```

Once the candidate events have been identified, these can be used by the inference engine for searching for concepts which match the slot rules.

Inference and the variables

The variables are filled by the inference system as part of the processing of the main-events. Therefore, specific calls to the inference system for locating information which corresponds to the variables are not necessary.

For each of the successful main-events, the inference algorithm provides the template application with the list of the concepts which have been associated to each of the variables. For example, the inference of the candidate main-event:

```
FIAT bought RENAULT
```

against the main-event condition:

```
V=COMPANY1 acquired V=COMPANY2
```

will produce a list of variables and associated values which will be returned to the template code and will consist of:

```
[(V=COMPANY1,FIAT),(V=COMPANY2,RENAULT)]
```

The concepts corresponding to the variables identified are subsequently used by the template application to fill the slot rules which refer to the variables.

Inference and the slots

The slot-rules definitions entered by the user can be subdivided into two different categories:

- rules which refer only to a specific variable used in the main-event, for example:

```
S=COMPANY-PREDATOR:        V=COMPANY1
```

- rules which refer to specific variables, the template-name or other slot-names but with additional conditions, for example:

  ```
  S=VALUE-TAKEOVER:          the cost of the T=TAKEOVER
  S=BANK-ADVISER-PREDATOR: the adviser of V=COMPANY1
  S=ATTRIBUTION:             the person or company who said something
                             about the ... S=VALUE-TAKEOVER
  ```

The first type of slots are filled by the template application with the nodes which have been identified for the specific variable. The inference system is therefore not used in this particular situation and the slots are filled directly within the template application. The variable in the slot-rule is matched against those identified by the inference system during the analysis of the main-events and the slot is directly filled. For example, the slot:

```
S=COMPANY-PREDATOR:        V=COMPANY1
```

will be filled obtaining "*FIAT*" as the value corresponding to "*V=COMPANY1*" in the list of variables identified by the inference of the main-event:

```
[(V=COMPANY1,FIAT),(V=COMPANY2,RENAULT)]
```

A different approach is taken when the slot-rules refer to variables but also introduce additional conditions, for example in the following slots:

```
S=VALUE-TAKEOVER:          the cost of the T=TAKEOVER
S=BANK-ADVISER-PREDATOR: the adviser of V=COMPANY1
S=ATTRIBUTION:             the person or company who said something
                           about the ... S=VALUE-TAKEOVER
```

In this case, the application substitutes each of the formal elements in the slot-rule definition with the corresponding information which has already been identified.

More specifically, each link to the formal element will be substituted with a link to the value identified for that formal element, plus an opposite link to the target of the link, while the original links will be removed. For example, the slot-rule "*the cost of the T=TAKEOVER*", will be transformed in "*the cost of the*" concept corresponding to each of the main events, for example "*FIAT bought RENAULT*". In figure 9.8 the representation of the slot "*S-VALUE-TAKEOVER*" before and after the transformation is shown.

Once the formal elements have been eliminated in the slot-rules and substituted with the actual values associated, the inference system is used to match these rules against the new events in the source articles. For example, the event produced by the sentence:

```
The cost of the takeover is 100 million dollars''
```

will be matched against the new slot rule of the "*S=VALUE-TAKEOVER*" slot (see figure 9.8) which will be:

```
The cost of the (event) FIAT bought RENAULT.
```

The same approach is taken for the other formal-elements. For example, the inference system will try to match the following modified slot-rule

```
S=BANK-ADVISER-PREDATOR:          The adviser of V=COMPANY1
```

shown in figure 9.9 against an event corresponding to the sentence:

```
The adviser of FIAT is SBC-Warburg.
```

which is shown in figure 9.10.

The inference system will match the two events by recognising that the nodes "*adviser*" in both events are actually representing the same concept. Once the two nodes have been matched, the inference system will identify the instance "*SBC-Warburg*" as relevant concept for filling the slot.

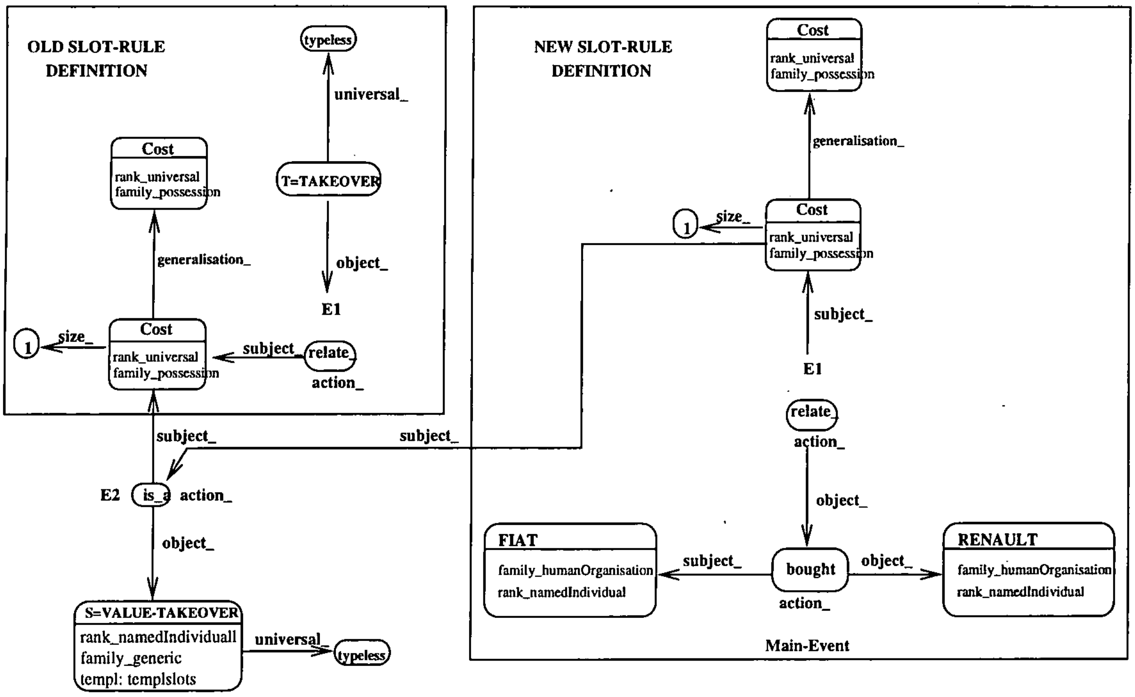

Figure 9.8: The modification of a slot-rule which refers to the template name

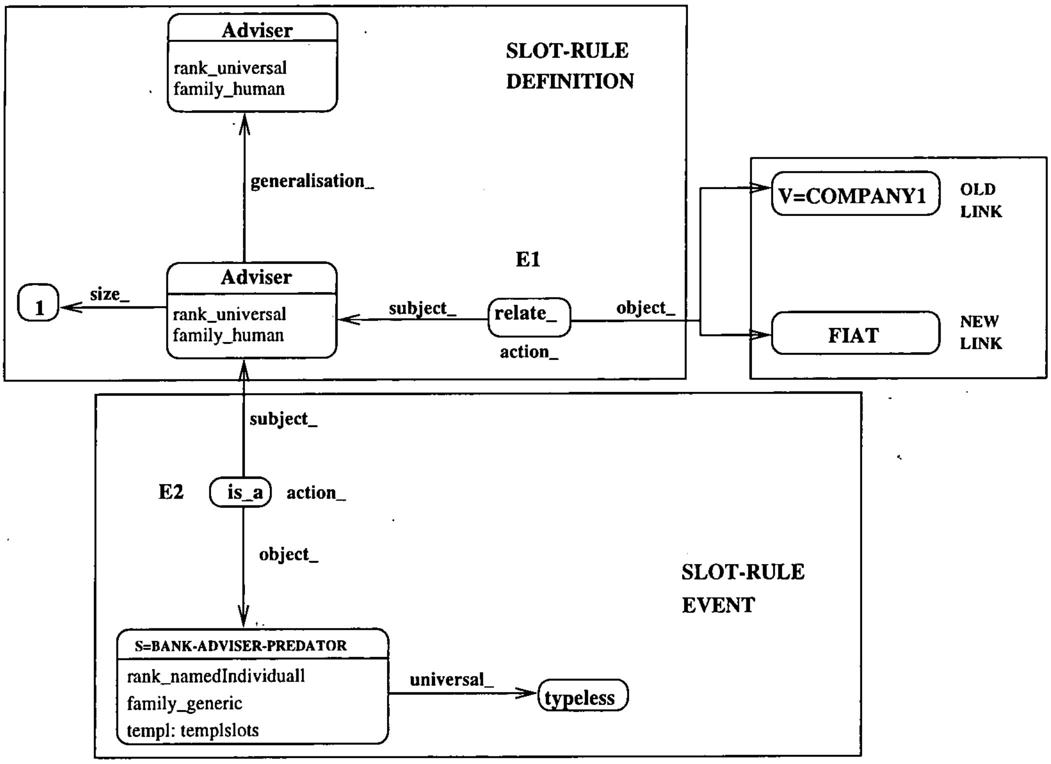

Figure 9.9: The modification of a slot-rule which refers to a variable

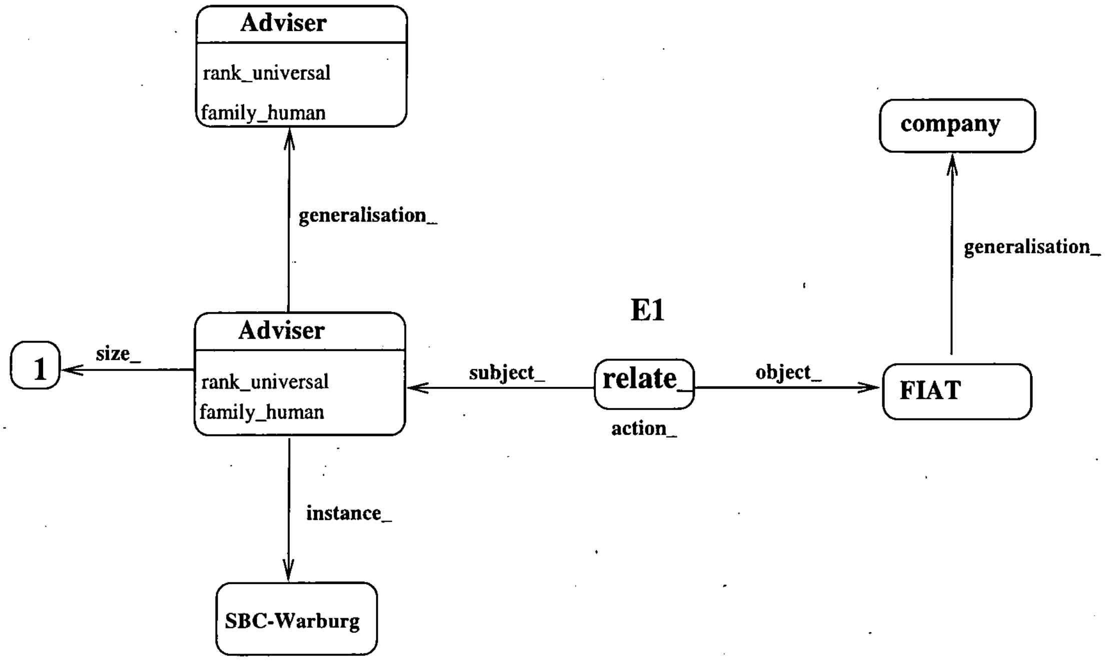

Figure 9.10: A candidate event for the S=BANK-ADVISER-PREDATOR slot-rule.

The identification of the equivalence of the two nodes referring to the concept of *adviser* by the inference system is due to the fact that the semantic network does not currently provide specific normalisation functionalities which should perform such operations. Ideally, nodes corresponding to the same concept should be unified automatically when they are firstly stored in the semantic network and, in this specific case, when the links to the formal-elements of the user-defined template are substituted. At present, the semantic network is unable to unify such events and, therefore, a specific rule has been added to the inference engine, to identify the equivalence of two nodes referring to the same concept:

```
>  | isUniv p && isUniv q && equalgen (samelevelgen q) (samelevelgen p)
>    = tracel tm "RULE 14b"
>      $ proveProperties tm p q
```

The rule identifies two nodes as being equal when they are both "*universals*" and the *premis* shares all the generalisations of the *question*. To avoid expensive searches in the hierarchy, the rule only compares the first level of generalisations of both nodes. If two nodes share the same generalisations above the first level of the hierarchy, they will not be unified.

String slots

String-slots differ from the other kinds of slots for three main reasons:

1. they are entered by the user using full sentences, as opposed to noun-phrases which are used for defining the other kinds of slot-rules;
2. they already provide the string which is used to fill the slot if the associated rule has been proved True.
3. the word "*otherwise*" can be used to provide a default fill if any of the other rules fail.

The string-slot rules are stored in the Template data-structure using a specific SlotRule data-type, which is defined as follows:

```
>      SlotNetUserDefinStr [(String,[Noderef])]
```

for each of the possible string-fills, the fill is stored in the Slot-Rule data-structure together with the associated rules.

Each rule is proved by the inference system in the same way as the template main-events are proved. If any of the rules is proved True, the slot is filled with the associated fill-string. If a statement "otherwise" is encountered and no other condition has been proved to be True, the slot is filled with the string-fill associated to the "otherwise" statement. The inference engine is used to prove each of the statements, which have status "*wh_question*", in the same way as the template main-events are checked.

For example, the following string-slot is filled with "*FRIENDLY*" if the first condition is False:

```
S=TYPE-TAKEOVER
HOSTILE:
T=TAKEOVER is hostile.
FRIENDLY:
otherwise
```

Multiple slot-fills

The LOLITA inference system was originally developed to produce an answer for a specific question given. Although this is satisfactory for many tasks, for example the production of SQL-language expressions from Natural Language queries [Garigliano *et al.*, 1995], for the production of slot-fills this approach can be wrong. In fact, there can be cases in which *all* possible matches for a question should be returned, rather than only one. For example, the following slot rule for the slot "*S=ATTRIBUTION*":

```
S=ATTRIBUTION: The organization that announced T=TAKEOVER
```

matched against a sentence "*FIAT and RENAULT announced that FIAT bought RENAULT*" should produce:

```
S=ATTRIBUTION: FIAT
               RENAULT
```

However, the standard LOLITA inference engine would only produce the first answer, skipping the others.

For the purposes of this work, the system has therefore been updated, adding features for returning all possible correct answers to the questions, rather than only the first one.

The most significant changes to the inference engine is the introduction of the "*Or*" data-type for representing the different kinds of status from the main one in the "*Or*" data-structure as follows:

```
data Or = OrL [Task] |
          SuccessChild Task |
          OrLAll [Task] [Task]
```

where "*OrLAll*" contains the list of the top-level tasks still to be proved or the successfull tasks already proved.

Using this data-structure, all the top-tasks are proved and, depending on the function used to retrieve the data, only the first match or all the possible ones can be returned. The top-level inference rules have also been updated to make use of the new data-structure.

Chapter 10

Results and Evaluation

10.1 Introduction

This chapter aims to evaluate the work described in this thesis with special attention to the thesis aims and the methodological issues discussed in chapter 1.

The evaluation of the pre-defined templates described in chapter 6 and chapter 7 is presented in section 10.2. The evaluation has been carried out analysing the *performance* of the *takeover* template described in chapter 7 according to the MUC-6 evaluation metrics. This template represents an *interactive* template definition which has been designed analysing a considerable amount of relevant articles and improving the definition according to the first results produced by the system. The evaluation of the performance gives an indication regarding the usefulness of the application for the financial operator. The *evaluation of the task*, the financial templates definitions, has not been carried out because specific methodologies for the evaluation of templates definitions have not yet been defined in the literature. In addition, the evaluation of the financial templates definitions would have involved the identification of direct links between an event and a share price movement, which is rather difficult.

The evaluation of the user-definable template interface described in chapter 8 and chapter 9 is presented in section 10.3. The aim is to investigate two main elements: the *ease of use* of the interface and the *performance* of the templates produced by the interface according to the MUC-6 evaluation metrics. The *ease of use* has been evaluated analysing the difficulties encountered by 14 potential users of the system in entering a takeover template definition. The evaluation of the *performance* of the interface has been carried out analysing the results produced by the system using two different template definitions: the *interactive* takeover template definition described in chapter 8 and a *non-interactive* template. The *non-interactive* template definition has been selected among the 14 templates definitions entered by 14 potential users choosing the template closer to the average total time for defining the templates. The takeover *interactive* template definition, which has been designed analysing a considerable amount of relevant articles and improving the definition according to the first results of the system, gives an indication of the performance of a template defined by an expert user of the system, although not a natural language engineer. Differently, the *non-interactive* template gives an indication of the performance of the system processing a template definition entered by a new user of the system.

Finally, the results of the user-definable template interface have been compared with the results of the pre-defined templates. The aim is to investigate the trade-off between the time saved designing the templates using the user-definable template interface and the loss of performance of these templates compared to the pre-defined templates.

The evaluation of the *performance* of both pre-defined and user-definable templates has been carried out using the same set of 55 financial articles from "*The Financial Times*". The templates produced by the system have been matched against a set of keys defined interactively looking at the first results produced by the system. A correct-fill was assumed when the slot reported a fragment of the original text or referred to the same object. The end-user of the system (a financial operator), in fact, is not necessarily concerned with the templates being filled with an exact copy of the source text, but with the fact that the information refers to the same object.

Although the results cannot be considered statistically significant since only one template definition (the takeover template) has been evaluated, they provide an indication of the performance of the system in extracting pre-defined and user-defined templates from source financial texts. Moreover, obtaining statistically significant results has been proven a very difficult task also for state-of-the-art evaluations such as the MUC-6 competition [Chinchor, 1995]. To produce statistically significant results, a high number of templates definitions would have had to be designed and tested, which is beyond the scope of this work.

The methodologies for the evaluation of natural language processing systems, particularly of the LOLITA system, and of information extraction tasks have been discussed in detail in [Callaghan, forthcoming 1997].

10.2 Evaluation of the definition and implementation of the financial templates

This section describes the evaluation of the solution to the project aim 1 which has been described in chapter 6 and chapter 7. In section 10.2.1 we describe the reasons why the evaluation of the financial templates has not been considered relevant within this work, while in section 10.2.2 the performance of the templates using the MUC evaluation metrics is described.

10.2.1 Evaluation of the financial templates

We believe that the definition of the financial templates according to the financial activities approach described in chapter 6 is an effective solution for a financial information extraction system.

However, a complete evaluation of the specific templates has not been carried out in the context of this work. This is because a clear methodology for the evaluation of the definition of templates has not yet been defined in the literature.

Onyshkevych [Onyshkevych, 1993] defined some criteria that the templates should respect. The criteria, which are described in detail in section 3.4 include the following aspects: descriptive adequacy, clarity, determinacy, perspicuity, monotonicity and reusability. These aspects, however, are too general to be used as an efficient evaluation methodology for templates definitions and, therefore, an evaluation based on this criteria has not been performed in this work.

In the context of this work, we therefore focus on the evaluation of the performance of the financial templates which is presented in the following sections.

10.2.2 Evaluation of the performance of the financial templates

In this section we evaluate the performance of the takeover template definitions described in chapters 6 and 7. Although direct conclusions can only be drawn for the takeover template, the performance of this template can provide an indication of how the system could perform extracting other financial templates described in chapter 6.

The evaluation set is based on a total of 55 financial articles from the financial times on CD-ROM year 1994 and 1995. A total of 30 articles were relevant takeover articles, while the other 25 articles were non-relevant financial articles. The articles chosen for the evaluation were different from the articles used during the development of the rules for the takeover template described in section 7.3.

In figure B.1 a relevant takeover article from the evaluation set is shown, while figure B.2 shows a non-relevant article from the evaluation set.

The takeover template definition has been evaluated matching the templates produced by the system against a set of keys associated to each of the articles of the evaluation set. A slot-fill was considered correct if the slot was filled with a fragment of the original text corresponding to the correct information or a fill which referred to the same object/entity. This is because an end-user of the system (a financial operator) is not necessarily concerned with the slots being filled with an exact copy of the source text, but with the fact that the information refers to the same object. For example, the key for the article shown in figure B.1 is:

```
COMPANY_TARGET:   ‘‘Leaside Bus Company. ’’
COMPANY_PREDATOR: ‘‘Cowie Group. ’’
```

Some of the keys were updated after the first run of the system allowing alternative fills which, although were not an exact copy of the source text, referred to the same object. For example, the following key for the article number 14 was updated by adding two alternative possible fills for the slots “COMPANY_TARGET” and “COMPANY_PREDATOR”:

```
<TAKEOVER-0002140000-1> :=
     COMPANY_TARGET: "Kayser-Roth"
     / "the US rights maker Kayser-Roth"
     COMPANY_PREDATOR: "Grupo Synkro"
     / "Synkro Grupo. "
```

A total of 11 alternative fills which did not correspond to a fragment of the original text for the total 35 relevant articles was added to the set of keys. The correctness of the alternative fills was ensured by asking 5 students to check whether such fills carried the same information as the original fill corresponding to the exact copy of the source text. The students reported equivalence judgments for all fills apart from three, which were accepted because 3 out of 5 students returned positive answers.

10.2.3 Evaluation techniques: the MUC-6 evaluation measures

The performance of the system has been evaluated using the MUC-6 evaluation measures (*precision*, *recall* and *'F' measure*) described in section 3.5.3 for two main reasons:

- the MUC evaluation metrics represent the measures which have been most widely used in the evaluation of information extraction systems. A high number of systems and templates has been tested using these measures;

- using standard MUC evaluation metrics can potentially allow to compare the results of this work with equivalent templates produced by other systems;

The evaluation of the performance of the takeover templates has been carried out using a modified version of the MUC-6 scoring program, rather than computing the evaluation measures by hand. The main reason for employing the scoring program is to facilitate further scoring of the same set of articles for eventual further developments of the core system.

The scorer matches each of the templates against the corresponding pre-defined key (e.g. the key for the article B.1shown in figure B.3),producing a report including the complete scoring results for each of the articles processed plus a summary including the *precision*, *recall* and *'F' measure* of the overall evaluation. Figure 10.1 shows the final report for the evaluation of the articles of the evaluation set.

10.2.4 Evaluation of the results using the evaluation metrics

In this section we present the evaluation of the 55 financial articles of the evaluation set according to the MUC-6 evaluation metrics discussed in the previous section.

Figure 10.1 shows the overall results for the 55 takeover templates. The final results show that the system's overall performance measures were:

```
                        P&R     2P&R    P&R
F-MEASURES              51.03   57.41   45.93


OVERALL PRECISION:      63%
OVERALL RECALL:         43%
```

```
Report for the standard takeover templates finalEval2:

 * * * TOTAL SLOT SCORES * * *
-----------------------------+-------------+-------------+-----------------------
SLOT            POS    ACT| COR PAR INC | MIS   SPU   NON| REC PRE UND OVG ERR SUB
-----------------------------+-------------+-------------+-----------------------
takeover         36     38|  28   0    0|  8    10      0|  78  74  22  26  39   0
  companytar     36     21|  12   0    5| 19     4      0|  33  57  53  19  70  29
  companypre     35     31|  18   0    3| 14    10      0|  51  58  40  32  60  14
  typetakeov     36     38|  28   0    0|  8    10      0|  78  74  22  26  39   0
  value          28      7|   3   0    1| 24     3      1|  11  43  86  43  90  25
  badviserpr      0      0|   0   0    0|  0     0      0|   0   0   0   0   0   0
  badviserta      0      0|   0   0    0|  0     0      0|   0   0   0   0   0   0
  expirydate      0      0|   0   0    0|  0     0      0|   0   0   0   0   0   0
  attrib          9      2|   1   0    1|  7     0      0|  11  50  78   0  89  50
  currentsta      0      0|   0   0    0|  0     0      0|   0   0   0   0   0   0
  denial          0      0|   0   0    0|  0     0      0|   0   0   0   0   0   0
-----------------------------+-------------+-------------+-----------------------
ALL OBJECTS     144     99|  62   0   10| 72    27      1|  43  63  50  27  64  14
-----------------------------+-------------+-------------+-----------------------
                                                    P&R         2P&R        P&2R
F-MEASURES                                        51.03        57.41       45.93
```

Figure 10.1: The final score report for the takeover financial template.

The overall performance, *F-Measure*, is equal to 51%. The *precision* measure, which is 63% is higher than the *recall*, which is equal to 43%. This is partly due to the fact that a number of slots "*VALUE_TAKEOVER*" and "*ATTRIBUTION*" were not filled by the system, which reduced the overall *recall*. On the contrary the precision for the slots "*COMPANY_TARGET*" and "*COMPANY_PREDATOR*" was rather high, which contributed to the overall figure.

The reason why the slots "*VALUE*" and "*ATTRIBUTION*" present lower scores is that they are relatively more difficult to be filled by the template application. The information which is relevant for these slots is in fact often linked to different events. Therefore, the core system must be able to analyse the two events correctly and *relate* them to the main *takeover* event before the template application can retrieve it from the semantic network. Figure 10.4 shows an example article from the evaluation set where this problem occurs. In this specific example, the sentence which carries the information regarding the cost of the takeover: "*The business, which is expected to make annual profits of about 250,000 pounds, has been bought through its subsidiary, Dyson Industries, at a cost of 2.3 million pounds*" is not correctly processed by the core system, which makes it impossible to extract the information for the template application.

In figure 10.2 the *recall* for each of the slots is shown. The slot with highest recall is the slot "*TYPE_OF_TAKEOVER*". This is due to the fact that the slot was never left empty by the system but, in case the article did not report any relevant information or the core system did not correctly process the related information, it was filled with the default value "*FRIENDLY*" and most of the articles in the evaluation set were about friendly takeovers.

The slots "*COMPANY_PREDATOR*" and "*COMPANY_TARGET*" show similar values to each other. This is due to the fact that if a takeover event is correctly identified by the template application, it is likely that the two relevant companies will be directly linked to the event. Finally, the slots "*VALUE*" and "*ATTRIBUTION*" present significantly lower figures. The same observations can be made for the measure *precision* which is shown in figure 10.3.

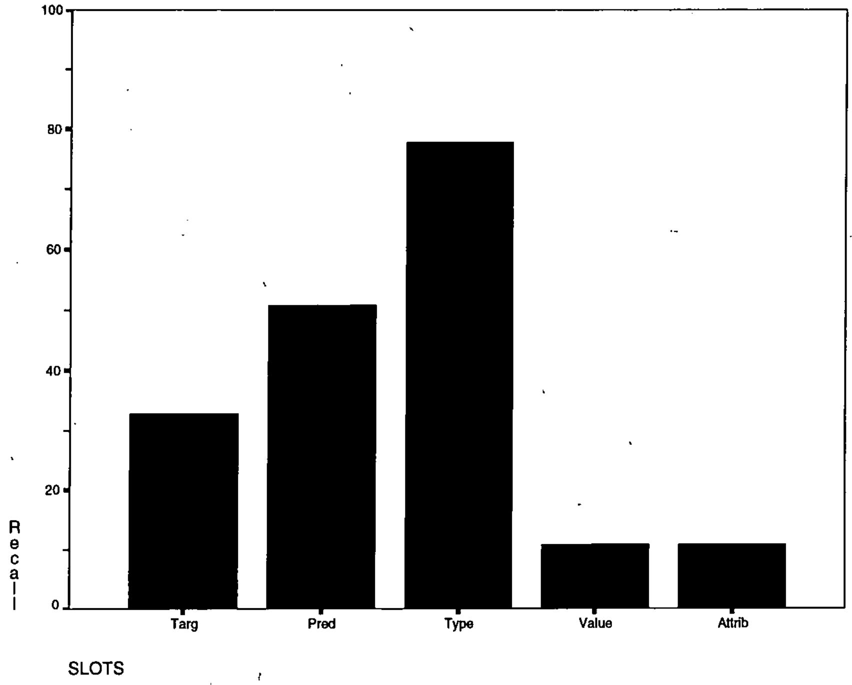

Figure 10.2: The recall for each of the slots of the pre-defined takeover template.

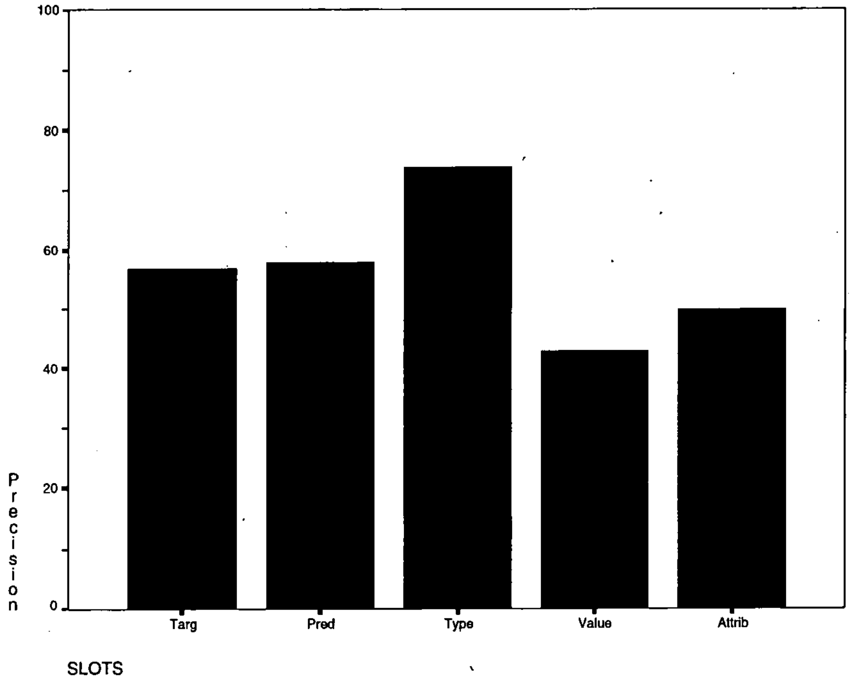

Figure 10.3: The precision for each of the slots of the pre-defined takeover template.

As part of its diversification into ceramics and refractories, J&J Dyson has acquired Norton of Stoke-on-Trent's Gimson secondary kiln furniture and chimney liner business. The business, which is expected to make annual profits of about 250,000 pounds , has been bought through its subsidiary, Dyson Industries, at a cost of 2.3 million pounds.

Figure 10.4: An example article where the slot "*VALUE*" has not been filled by the system.

The loss of precision and recall which can be observed in the overall figure of 51% depends on the incorrect analysis of the source text performed by the core system, rather than the incorrect extraction of the relevant information by the template application. In all cases when the core analysis was correct, the template application was able to correctly identify and extract the relevant information for the takeover template.

The overall figure (51%) is rather high, especially considering that it has been obtained without any specific improvement of the core system. The precision (63%) is significantly higher than the recall (43%). The financial operator could profit from using the system by being able to quickly analyse a high number of source document and extract 43% of the relevant information regarding takeovers. The 63% of the extracted information would be correct, which would allow the operator to gain important knowledge for taking financial investment decisions in a limited time. Such knowledge would be otherwise lost due to the lack of time of the financial operator.

10.3 Evaluation of the user-definable template interface

This section describes the evaluation of the solution of the project aim 2 which has been described in chapter 8 and chapter 9.

The evaluation of the user-definable template interface is based on an experiment similar to the one carried out for the design of the interface (see section 8.2). The potential users of the system were asked to describe a takeover template using the specific syntax of the user-definable interface (see section 8.2.3).

Two were the main aims of the experiment:

- evaluate how much easier and quicker is for the user to define a template using the user-definable template interface compared to hand-coded templates;
- attempt to quantify the loss of performance of a template defined using the user-definable interface compared to an equivalent template hand-coded in the system.

These two evaluation provide an indication of the trade-off between the ease of use and the loss of performance of the templates definitions entered using the user-definable template interface compared to those hand-coded in the LOLITA system. In section 10.3.3 we discuss the evaluation procedure and the results of the first evaluation aim, while in section 10.3.3 we discuss the results for the second evaluation aim.

10.3.1 Evaluation of the design and usability of the user-definable interface: evaluation task and data

The evaluation was based on an experiment involving potential end users of the user-definable template interface. The users were asked to enter the definition of a takeover template using the syntax of the user-definable interface using a form available on a WWW server[1] and corresponding to the structure of a user-definable template[2].

The takeover template was chosen because a "takeover" represents a concept which can be easily explained to the potential users of the system while others, for example the MUC-6 tasks, are more difficult to explain and less meaningful for the end-user of an information extraction system. The decision of using a WWW

[1] The pages can be accessed at the address: http://www.dur.ac.uk/ dcs3mc/udeval.

[2] The implementation of the form is based on the FormMail program by Matt Wight, which can be found at: http://www.worldwidemard.com/scripts/.

The Durham Financial user-Definable Template Interface: on-line WWW form

The task:

This form allows you to enter new template definitions for the Durham Financial User-Definable Template Interface by defining your own takeover template which will be used to extract relevant information from a sample financial article.

You should define a template for extracting information regarding **company takeovers** from a article **similar to the following one**:

Filofax Group, the USM-quoted personal organiser company, has acquired Drakes Office Systems from its founder, Mr Tom Drake, for Pounds 3m, to be satisfied by cash and the issue of 1m ordinary 5p shares. Drakes claims to be the UK market leader in Wire-O bound carbonless duplicate message books. Its Ring Back brand forms a range of telephone message and similar business forms with a dominant market share. In 1992, Drakes made gross profits of Pounds 727,000 on sales of Pounds 1.7m.

Within a **few days** you will receive an e-mail containing:

- comments and syntax errors you made entering the definition
- the financial article on which your definition has been tested
- the template extracted from such article

Figure 10.5: The description of the evaluation task available on the WWW server

interface was taken to allow users from different backgrounds and countries to easily access the interface.

A sample takeover article was available to the users to suggest the information which could be relevant for the takeover template, while a user-defined merger template was available as an example of a user-defined template. In addition, the full description and instructions on how to define templates using the user-definable template interface was available on the WWW server.

Figure 10.5 shows the description of the task which was available on-line, while in figure 10.6 the user-defined merger template is shown. Figure 10.7 shows an example form for a user-defined takeover template submitted by a potential user.

```
Template_Name:          T=MERGER

Variables:              V=COMPANY1 is a company.
                        V=COMPANY2 is a company.
                        V=COMPANY3 is a company.

Template main-events:   V=COMPANY1 merged with V=COMPANY2 creating V=COMPANY3
                        V=COMPANY1 merged with V=COMPANY2

Definition of the slots:
S=FIRST-COMPANY:        V=COMPANY1
S=SECOND-COMPANY:       V=COMPANY2
S=NEW-NAME:             V=COMPANY3
S=DATE-OF-ANNOUNCE:     the date when T=MERGER is announced
S=DATE-OF-MERGER:       the date when T=MERGER takes place
S=ATTRIBUTION:          the person that announced T=MERGER
                        the company that announced T=MERGER
```

Figure 10.6: The user-defined merger template available on the WWW server

```
Template_Name:          T=TAKEOVER
Variable_1:             V=COMPANY1 is a company
Variable_2:             V=COMPANY2 is a company
Variable_3:             V=VALUE is money
Main_Event_1:           V=COMPANY1 bought V=COMPANY2 with V=VALUE
Main_Event_2:           V=COMPANY1 bought V=COMPANY2.
Slot_Name_1:            S=COMPANY-PREDATOR
Slot_Rule_1.1:          V=COMPANY1
Slot_Name_2:            S=COMPANY-TARGET
Slot_Rule_2.1:          V=COMPANY2
Slot_Name_3:            S=VALUE-TAKEOVER
Slot_Rule_3.1:          V=VALUE
Slot_Rule_3.2:          The cost of T=TAKEOVER
Slot_Name_4:            S=ATTRIBUTION
Slot_Rule_4.1:          The company that announced T=TAKEOVER
Time_Instructions:      15 minutes
Time_Form:              10 minutes
```

Figure 10.7: An example submission of a user-definable template.

Dear user,
thank you very much for submitting the form for your user-defined takeover template. This e-mail contains the following information:

1) The source article which has been matched against your template definition

2) The template which should be produced by the system

1) Article matched against your template definition:

BELL ATLANTIC announced it will acquire Tele-Communicatiosn Inc. with 18 billion dollars. The deal will radically change the US communications industry. It will also be one of the country's biggest ever takeovers. JP Morgan, Bell Atlantic's adviser, suggested that the deal is bound to face strong regulatory scruting and it would be the first full merger between a US telephone company and a cable business at a time when the two industries are converging to create a single, multi-media inter-active entertainment and information business. Bell Atlantic, the telecommunications group serving the middle part of America's eastern seaboard, has been one of the most aggressive telephone companies trying to enter the video industry.

2) Template which should be produced by the system according to your template definition:

```
Template:               T=TAKEOVER
S=COMPANY-PREDATOR:     Bell Atlantic
S=COMPANY-TARGET:       Tele-Communications Inc.
S=VALUE-TAKEOVER:       18 billion dollars.
S=ATTRIBUTION:          Bell Atlantic
```

Thank you for your co-operation.

Figure 10.8: The reply e-mail for a user-defined template

After the users submitted the template form, the information was analysed and the users were sent an e-mail containing the following information:

- the source article matched against the template definition entered by the user;
- the template extracted from the source article which should have been produced by the system.

Figure 10.8 shows the reply e-mail for the form shown in figure 10.7.

A total of 14 users from different backgrounds and nationalities participated to the evaluation. The users have been grouped into four categories:

1. A total of 6 users were researchers or PhD students in the field of computer science. However, only four of them were familiar with information extraction technology and only one of them with user-definable interfaces.

2. A total of 2 users were researchers or PhD students in the field of economics, finance and business, with no prior knowledge of artificial intelligence, natural language processing or information extraction.

3. A total of 5 users were researchers or PhD students in other fields including physics, chemistry and electronic engineering.

4. Finally, one user was an Italian designer of information systems for hotels, but external to the University environment.

In appendix B the forms entered by the users are reported ordered according to the above categories.

10.3.2 Evaluation metrics and results

The evaluation of the results is based on three different measures which are described in this section:

1. the first measure, called *SlotErrors*, is used as an indication of the difficulty of defining a user-definable template and it is based on the number of errors identified in the forms submitted by the users;

2. the second measure is the average time taken by the user in entering the user-defined takeover template. The total time comprises the time spent by the users reading the definition and the time spent filling the on-line WWW form;

3. the last metric is a calculation of the time taken by the user for defining a *non-interactive* user-defined takeover template selected among the 14 templates definitions entered by the potential users of the system choosing the template closer to the average of the time taken for entering the templates.

The SlotError measure

The first measure, called **SlotError** measures the difficulty in entering a user-defined template and depends on the number of errors in the forms submitted by the users. The higher is the number of these errors, the more difficult is for the user to define a template using the user-definable template interface.

An error occurs when the user defines an element of the template which cannot be correctly interpreted by the system and leads to a missing or incorrect template or slot. The possible errors have been grouped into 4 different categories:

1. **Syntax errors.** These errors occur when the user defines a template using the wrong syntax, for example defining the name of the template in lower case, defining slot-rules using variables which have not been used in main-events or entering formal elements using the wrong syntax (e.g. the slot-name "*S=Main*" of the template 1.2 in appendix B). Syntax errors can be grouped into two categories. A first group comprises the errors which could be eliminated employing an effective user interface and would not be committed by the user in the first place. For example, the error in defining the slot-name "*S=Main*" could be easily avoided by employing an automated mechanism which transforms the slot-name in capital-letters once identified as such by the interface. More relevantly, these errors could be easily avoided in the first place designing a Graphical User Interface with specific features. For example, the GUI could include fields which already provide the specific syntax for the template's formal elements, for example "S=" or "T=". The second group comprises errors which could not be automatically identified by the system during the insertion of the template definitions. No errors belonging to this second category have been found in the template definitions entered by the 14 users.

2. **Natural language errors.** In this case we refer to errors which cannot be automatically located by the system, for example the definition of a slot-rule using a True/False assertion rather than a noun phrase (see figure 10.9).

```
Template-name:          T=TAKEOVER
Variable_1:             V=COMPANY1 is a company
Variable_2:             V=COMPANY2 is a company
Main_Event_1:           V=COMPANY2 bought V=COMPANY1
Slot_Name_1:            S=PRE-OWNER
Slot_Rule_1.1:          A person owned V=COMPANY1
```

The slot rule 1.1 contains a *natural language error.* This is because the rule has been defined using a *True-False assertion* rather than a *noun phrase* and the system would therefore be unable to fill the slot.

Figure 10.9: An example of natural language error

This kind of errors could theoretically be indentified by the LOLITA system performing an analysis of the source text. The current core system, however, could potentially fail to correctly analyse the sentence, which could lead to the identification of inexistent errors in the user's template definition. We therefore decided to avoid performing such an analysis.

3. **Reference errors.** These errors occur when the user does not correctly refer to the template-name, to a variable-name or to other slot-names to refer to corresponding information. In figure 10.10 an example of this kind of errors is shown. These errors could be automatically identified by the user-interface, since a slot-rule must always refer to at least one valid formal elements previously defined in the template.

4. **Other kinds of errors.** This category includes errors which cannot be included in the above four groups, for example cases in which the user defines two variables of type "company" in the takeover template which are incorrectly assigned to the slots "COMPANY_PREDATOR" and "COMPANY_TARGET". In this case, the relevant information will be correctly extracted, but it will be assigned to the wrong slot name. In figure 10.11 an hypothetical example of this kind of errors is shown. For the takeover event "FIAT bought RENAULT", the template would be correctly generated, but "FIAT" would be assigned to the slot "COMPANY_TARGET" and "RE-

```
Template_Name:          T=TAKEOVER
Variable_1:             V=COMPANY1 is a company
Variable_2:             V=COMPANY2 is a company
Main_Event_1:           V=COMPANY1 tookover V=COMPANY2
Slot_Name_1:            S=FIRST-COMPANY
Slot_Rule_1.1:          V=COMPANY1
Slot_Name_2:            S=SECOND-COMPANY
Slot_Rule_2.1:          V=COMPANY2
Slot_Name_5:            S=DATE-ANNOUNCE
Slot_Rule_5.1:          the date when the takeover is announced
```

The Slot_Rule 5.1 contains a reference error. This is because the user did not employ the name of the template (T=TAKEOVER) to refer to the takeover as a whole, but simply the general concept "takeover".

Figure 10.10: An example of error in the use of the variables

NAULT" to "company_predator", which is obviously wrong.

All 4 kinds of errors have been taken into account for computing the *SlotError* measure. Each kind of error is given the same importance in the evaluation of the results. This is because all errors have currently the same effect from the user's point of view: the incorrect extraction of information from the source texts. If more than one error is found in the same template-element (e.g. a slot), only one of them is taken into account in the index. This could change if a specific Graphical User Interface would be implemented, since it would allow to eliminate most of the syntax errors.

It is important to notice that no judgment is made regarding the kind of information that the user defined in the template, on the number of variables, main-events or slots defined, since no restrictions were defined in the template task (see section 10.3.1).

The *SlotError* index is based on the number of *slots* which have been wrongly defined by the users. This is because a slot directly identifies the information which will be stored in the template.

We therefore compute the total number of slots wrongly defined by the users

```
Variable_1:            V=COMPANY1 is a company
Variable_2:            V=COMPANY2 is a company

Template Main_Event: V=COMPANY1 acquired V=COMPANY2

S=COMPANY_PREDATOR:  V=COMPANY2
S=COMPANY_TARGET:    V=COMPANY1
```

The above template contains an error. V_COMPANY1 and V_COMPANY2 are incorrectly assigned to S=COMPANY_PREDATOR and S=COMPANY_TARGET.

For a takeover-event such as "FIAT bought RENAULT" the template would be incorrectly produced as:

S=COMPANY_PREDATOR: RENAULT
S=COMPANY_TARGET: FIAT

Figure 10.11: An example of error in the use of the variables

and we relate it to the total number of slots defined. In this way we obtain a measure of the number of slots containing errors, which will never be filled by the system or will be filled incorrectly, on the total number of slots defined by the users. This measure, called *SlotErrors* is defined as follows:

$$SlotErrors \doteq \frac{total\ number\ of\ slots\ containing\ errors}{total\ number\ of\ slots\ defined} \times 100$$

For example, the slotError for two templates of 5 slots each where 2 slots have been wrongly defined will be equal to:

$$SlotErrors = \frac{2}{2\times 5} = 20\%$$

which means that the 20% of the information of the templates have been wrongly defined by the user. The target of the evaluation is therefore to keep the measure as low as possible.

If a main-event is wrongly defined by the user and this will compromise the extraction of the template by the system, the total number of slots of the template are considered to be wrong, and added to the total number of slots containing errors.

The 14 templates defined by the users comprised a total of 70 slots. Each of the templates (which can be seen in appendix B) has been analysed and possible errors identified:

- **Template No.1.1** The following slots have been incorrectly defined:
  - the slot "*S=TAKEOVER-DATE*" with slot-rule "date of takeover" has been incorrectly defined by the user, since the slot-rule does not refer to the template as a whole using the name of the template ("*T=TAKEOVER*"). This error belongs to the error category 3.
  - the slot "*S=TAKEOVER-COST*" with slot-rule "cost of taking over the company" contains an error of category 3. Similarly to the case reported above, the slot-rule does not refer to the template as a whole, to variables or slots to refer to the takeover event.
  - the slot "*S=DATE-ANNOUNCE*" with slot-rule "the date when the Takeover is announced" contains a similar error of category 3, since the user does not refer to the template as a whole using the template name.
- **Template No.1.2** The following slots contain errors:
  - the slot "*s=PURCHASE-PRICE*" contains a syntax error since the character 's' has not been entered in capital letters;
  - the slot "*S=Main*" contains a syntax error, since part of the string is not in capital letters;
  - the slot "*S=Date*" contains a syntax error, since part of the string is not in capita letters;
- **Template No.1.3** All slots correctly defined.
- **Template No.1.4** All slots correctly defined
- **Template No.1.5** All slots correctly defined.

- **Template No.2.1** The slot-name "*=DATE-OF-ACQUISITION*" contains a syntax error, since the string does not start with the letter 'S'.
- **Template No.2.2** All slots correctly defined
- **Template No.3.1** All slots correctly defined.
- **Template No.3.2** All slots correctly defined.
- **Template No.3.3** All slots correctly defined.
- **Template No.3.4** All slots correctly defined.
- **Template No.3.5** The following slots have been incorrectly defined:
  - the slot "*S=DATE-OF-TAKEOVER*" with slot-rule "the date when the takeover was announced" is incorrectly defined, because it does not refer to the template-name for referring to the takeover event as a whole (error of category 3).
- **Template No.4.1** All slots correctly defined.

A total of 8 slots containing errors were therefore found in the templates entered by the 14 users. A total of 4 slots contained reference errors, while a total of 4 slots contained syntax errors.

We compute the measure *SlotErrors* for the reference errors as follows:

$$SlotError\ reference\ errors = \tfrac{4}{70} \times 100 = 5.71\%$$

and the measure *SlotErrors* for the syntax errors as follows:

$$SlotError\ syntax\ errors = \tfrac{4}{70} \times 100 = 5.71\%$$

The total *SlotErrors* measure has been computed without considering the *syntax errors* in the measure, because they could be avoided employing a specific Graphical User Interface which could be easily implemented. The final measure is therefore:

$$SlotError = \frac{4}{70} \times 100 = 5.71\%$$

The **SlotError** measures shows that the 14 users correctly defined most of the information of the templates with only the 5.71% of the information wrongly defined. This results cannot be considered statistically significant due the relatively limited number of users involved in the evaluation and to the fact that the evaluation task involved only one template definition (the takeover template). However, we can conclude that the interface is relatively easy to use, since the 14 users were able to define a template with a very low rate of errors and with no prior knowledge of the user-definable template interface and, in most cases, of concepts such as natural language processing and information extraction.

The average time taken for defining the templates

The second measure employed for the evaluation of the user-defined templates is the average time taken by the users in entering the user-defined takeover template. The total time taken by the users can be subdivided into three elements (see figure 10.12):

- **A**: the time taken for identifying the relevant information to be inserted in the template (e.g. company predator, company target, etc.);
- **B**: the time taken for understanding how to enter a template definition using the user-definable template interface reading the instructions available on-line;
- **C**: the time spent filling the on-line form according to the syntax of the user-definable interface.

The on-line WWW form required the users to enter the time for understanding how to define a template using the user-definable template interface and the time spent designing and entering the template definition using the on-line WWW form. Three averages can therefore be computed: the **average time reading the**

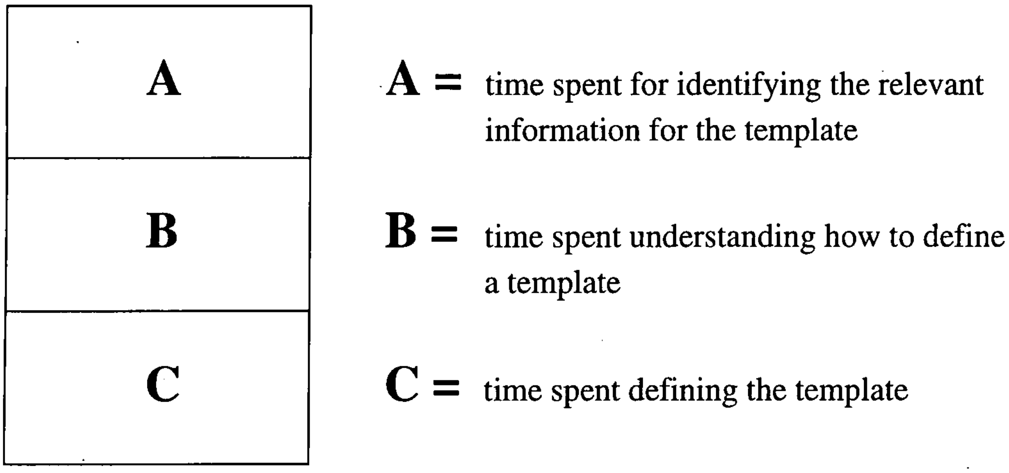
A
B
C
A = time spent for identifying the relevant information for the template
B = time spent understanding how to define a template
C = time spent defining the template

Figure 10.12: The time spent by a user for defining a template.

instructions (B), the **average time designing and entering the template using the form (A+C)** and the **average total time (TT)**.

These data associated to the takeover template definitions entered by the 14 users are shown in figure figure 10.13 and 10.14 respectively, while figure 10.15 shows the total time spent by each of the users for defining the templates.

We following measures have been computed from this data:

$Average\ time\ reading\ the\ instructions\ (\mathbf{B}) = \frac{325\ mins}{14} = 23.21\ mins$

$Average\ time\ defining\ the\ template(\mathbf{A}+\mathbf{C}) = \frac{220\ mins}{14} = 15.71\ mins.$

$Average\ total\ time\ (\mathbf{TT}) = \frac{545\ min}{14} = 38.92\ mins.$

From the above results we can notice that the 14 users took an average of 23.21 minutes (measure **B**) to read and understand the instructions on how to design user-definable templates. This is particularly relevant considering that all users apart from one did not have any prior knowledge of natural language processing, information extraction or template design. The users were therefore totally unfamiliar with concepts such as a template, a slot, a main-event condition etc. Most important of all, the users were totally unfamiliar with the possibility of defining a template using a user-definable interface. The result is therefore quite significant and shows that the syntax of the user-definable interface is indeed rather easy to be understood by the users. Moreover, this time is a "one off cost" for learning how to

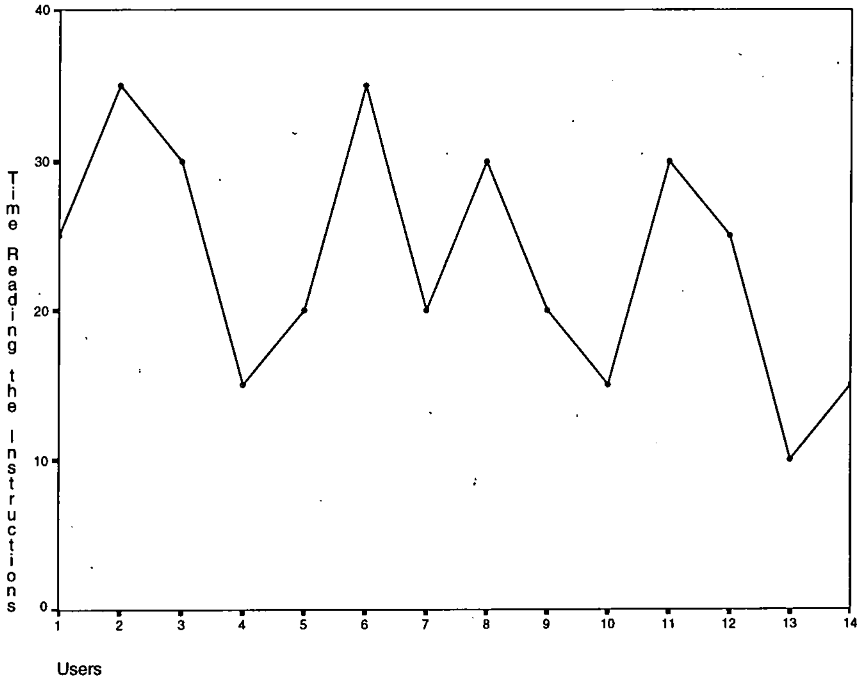

Figure 10.13: Time spent reading the instructions (**B**).

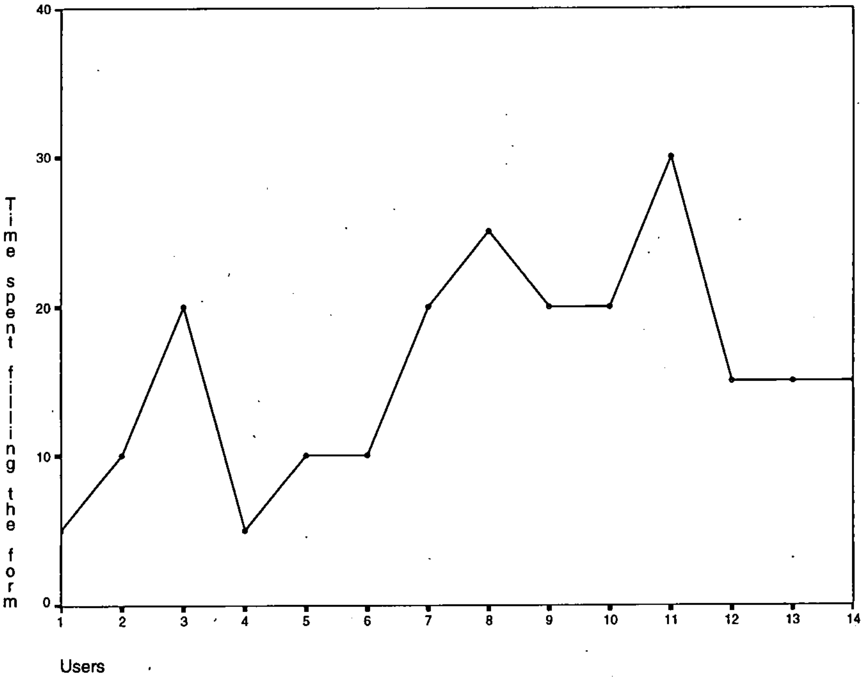

Figure 10.14: Time spent entering the definition (**A+C**).

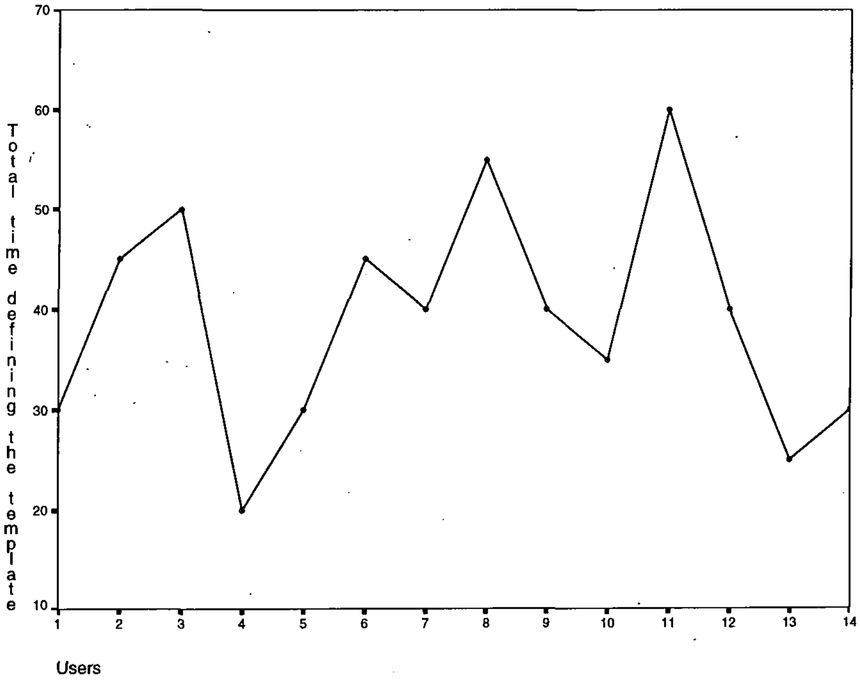

Figure 10.15: Total time spent defining the template (**TT**).

use the interface, since it would not be needed by the user for entering additional templates in the system.

Secondly, the users took an average of 15.71 minutes (measure **A+C**) for designing and entering a template definition. This time is also extremely limited. This is because at this stage the users already knew the syntax to be used for defining the templates, and only needed to type-in the information in the form, eventually looking-up the definitions of particular elements (e.g. how to define a template-name) in the quick-reference. The user could therefore easily define additional templates within the system in the rather limited time of 15.71 minutes for a template with an average number of slots.

Overall, the average user took 38.92 minutes (measure **TT**) for entering the definition of a takeover template. We consider this time particularly low, especially considering that the users who entered the template definitions did not have any specific knowledge of user-definable template interfaces and, in most cases, of natural language processing and information extraction.

Comparison with the time taken for coding the templates in the system

The third measure is the difference between the time needed by an inexperienced potential user of the user-definable interface for entering the definition of a *non-interactive* takeover template and the time needed to code an equivalent template definition in the LOLITA system. The greater is this difference, the greater is the advantage for a generic user to make use of the user-definable template interface for defining a new template rather than coding the definition directly in the system. The *non-interactive* user-defined template has been selected among the 14 templates definitions choosing the template closer to the average total time (**TT**). The time taken by the user for defining the *non-interactive* user-defined template is compared to the time taken by a expert programmer of the LOLITA system to code the same template in the system. Similarly to the user-definable templates, the time spent for coding the templates in the system can be further subdivided in (figure 10.12):

- **A**: the time taken for identifying the information to be inserted in the template;
- **B**: the time taken for understanding how to implement the templates in the LOLITA system: understanding the template code, the functions to access the semantic network etc.
- **C**: the time spent for coding the template and for compiling and testing the system.

In both the user-defined template and in the hand-coded version the time for identifying the relevant information to be extracted is assumed to be equivalent.

We will therefore assume that the user-definable template interface is easier to use than coding a template in the system if the time spent for hand-coding, compiling and testing the template definition in the system (**C**) is greater than the time for identifying the relevant information and defining the template (**A+C**) employing the user-definable template interface. The aim is to show that:

```
Template_Name:          T=COMPANY-TAKEOVER
Variable_1:             V=COMPANY1 is a company
Variable_2:             V=COMPANY2 is a company
Main_Event_1:           V=COMPANY1 acquired V=COMPANY2
Slot_Name_2:            S=COMPANY_PREDATOR
Slot_Rule_2.1:          V=COMPANY1
Slot_Name_3:            S=COMPANY_TARGET
Slot_Rule_3.1:          V=COMPANY2
Slot_Name_4:            S=ATTRIBUTION
Slot_Rule_4.1:          the person who reported T=COMPANY-TAKEOVER
Slot_Name_5:            S=PRICE
Slot_Rule_5.1:          the price that S=COMPANY_PREDATOR paid for
                        S=COMPANY_TARGET
Time_Instructions:      20 min
Time_Form:              20 min
```

Figure 10.16: The *non-interactive* user-defined takeover template selected for the evaluation.

$$C_{hand-coded} > C_{user-defined}$$

which follows from:

$$C_{hand-coded} > A_{user-defined} + C_{user-defined}$$

The *non-interactive* template chosen is the template number 3.1 (see appendix B), which has been defined in the total time of 40 minutes, which is the closest to the average total time of the 14 users, which is 38.92 minutes. Figure 10.16 shows the template 3.1 which has been chosen as *non-interactive* template.

The template shown in figure 10.16 has been coded in the LOLITA system and the time for implementing the template has been recorded.

The following times for the definition of the template have been recorded:

- 1 hours to set up the empty template within the LOLITA system;
- 3 hours for defining the template rules (main-events and slot rules);
- 0.75 hours for identifying the correct meaning of words (e.g. *to report*);

- 1.5 hours for compiling and testing the system.

the total time for coding the template, compiling and testing the system has therefore been 6.25 hours. This time can be considered of category 'C' according to figure 10.12.

We can compare this time with the time needed for identifying the relevant information and defining the template (**A+C**) using the user-definable template interface, which is equal to 15.71 minutes. We conclude that the user-definable template interface is easier to use than coding the *non-interactive* template in the system since the time **C** for hand-coding the *non-interactive* template in the LOLITA system is greater than the measure **A+C** for identifying the relevant information and defining the template using the user-definable template interface:

$C_{hand-coded} > C_{user-defined}$ from $C_{hand-coded} = 6.25\ hours > A + C_{user-defined} = 15.71\ minutes$

The time needed for hand-coding the template in the LOLITA system is therefore at least 24 times higher than designing the template using the user-definable template interface. This result appears even more relevant if we consider that the user who entered the definition of the *non-interactive* takeover template was totally unfamiliar with natural language processing, information extraction and user-definable template interfaces. On the contrary, the expert programmer who coded the definition in the system developed his expertise in information extraction and in programming within the LOLITA system over a period of more than 2 years.

10.3.3 Evaluation of the performance of the user-defined templates

In this section we evaluate the performance of the templates produced using the user-definable template interface comparing the results with the templates extracted using pre-defined templates. Two separate comparisons have been carried out:

1. The performance of the *interactive* user-defined takeover template described in chapter 8 and shown in figure 8.5 has been compared with the performance of the pre-defined *interactive* takeover template (see section 10.2). The *interactive* user-defined takeover template has been designed analysing a considerable amount of articles and improving the definition after the first run of the system. It therefore represents a template definition designed by an expert user of the system. The purpose of this comparison is to identify the loss of performance of a user-defined template implemented by an expert user of the system against an equivalent pre-defined template designed by an expert programmer of the LOLITA system. The loss of performance is related to the difference in time for defining both templates, obtaining an indication of the trade-off between the ease of defining templates using the user-definable interface and the loss of performance;

2. The performance of a *non-interactive* user-defined takeover template has been compared with the performance of the same template hand-coded in the LOLITA system. The *non-interactive* user-defined takeover template has been selected among the templates definitions entered by the 14 potential users of the system choosing the template closer to the average time for entering the templates definitions. The target of this evaluation is to measure the loss of performance of a user-defined template designed by an unfamiliar user in a very limited amount of time in comparison with the same template hand-coded in the system. The loss of performance is related to the difference in time for defining both templates, obtaining an indication of the trade-off between the ease of defining templates using the user-definable template interface and the loss of performance of the user-defined template definition against the hand-coded version.

It is important to notice that the two comparisons do not provide statistically relevant results, since only one template definition is used in both comparisons.

A higher number of templates definitions, ideally designed and implemented by different people, would have had to be used for both comparisons which, however, would have lead to a very expensive evaluation which is beyond the scope of this work. Moreover, obtaining statistically significant results has been proven a very difficult task also for state-of-the-art evaluations such as the MUC-6 competition [Chinchor, 1995].

Although the results cannot be considered statistically significant, they provide a good indication of the performance of the system in producing pre-defined and user-defined templates.

10.3.4 Evaluation of the takeover user-defined template against the hand-coded version

In this section we evaluate the performance of the user-definable takeover template described in chapter 8 and shown in figure 8.5. This template has been designed by the author and represents a template designed by an expert user of both the user-definable template interface and the LOLITA system. The definition of the user-defined takeover template has been based on the experience gained in the definition of the pre-defined takeover template described in chapter 6 and chapter 7.

It is important to notice that both the pre-defined and the user-defined takeover templates have been defined in an "*interactive*" way. This means that the definitions of both templates have been improved according to the results of the first analysis of source articles, and from looking at a relevant number of source articles. The limited world knowledge of the LOLITA core system has been overcome in these template definitions by adding specific definitions for the main-events and slot-rules of the templates.

The evaluation of the user-defined takeover template has been carried out using the same evaluation set, measures and scoring criteria employed for the evaluation of the pre-defined takeover template which have been described in section 10.2.2

```
Report for the user-definable templates finalEvalUD2:

------------------------+-------------+-------------+-----------------------
SLOT            POS   ACT| COR PAR INC | MIS  SPU  NON| REC PRE UND OVG ERR SUB
------------------------+-------------+-------------+-----------------------
takeover         36    23|  14   0   2|  20    7    0|  39  61  56  30  67  13
  companytar     36    23|  11   0   5|  20    7    0|  31  48  56  30  74  31
  companypre     35    23|  11   0   5|  19    7    0|  31  48  54  30  74  31
  typetakeov     36    23|  14   0   2|  20    7    0|  39  61  56  30  67  13
  value          28     4|   4   0   0|  24    0    0|  14 100  86   0  86   0
  badviserpr      0     0|   0   0   0|   0    0    0|   0   0   0   0   0   0
  badviserta      0     0|   0   0   0|   0    0    0|   0   0   0   0   0   0
  expirydate      0     0|   0   0   0|   0    0    0|   0   0   0   0   0   0
  attrib          9     2|   1   0   1|   7    0    0|  11  50  78   0  89  50
  currentsta      0     0|   0   0   0|   0    0    0|   0   0   0   0   0   0
  denial          0     0|   0   0   0|   0    0    0|   0   0   0   0   0   0
------------------------+-------------+-------------+-----------------------
ALL OBJECTS     144    75|  41   0  13|  90   21    0|  28  55  63  28  75  24
------------------------+-------------+-------------+-----------------------
                                                 P&R       2P&R        P&2R
F-MEASURES                                      37.44      46.17       31.49
```

Figure 10.17: The final score report for the user-defined takeover financial template.

and 10.2.3. The evaluation set included the same 55 articles from the Financial Times with the same percentage of relevant and non-relevant articles. This ensures that the results for both evaluations are fully comparable.

Figure 10.17 shows the overall results for the 55 user-defined takeover templates. The overall performance (*F-Measure*) is equal to 37%. The precision, which is equal to 34% is slightly higher than the recall, which is equal to 28%. The three slots "*COMPANY_TARGET*", "*COMPANY_PREDATOR*" and "*TYPE_OF_TAKEOVER*" present similar figures in terms of *precision* and *recall*. Differently, the slots "*VALUE*" and "*ATTRIBUTION*" present lower figures. Similarly to the pre-defined takeover template, the reason why these slots present lower scores is due to the fact that they are relatively more difficult to be filled by the template application, since they rely heavily on deep analysis performed by the core system. The core system must be able to analyse the two events correctly and *relate* them to the main *takeover* event before the template application can retrieve it from the semantic network.

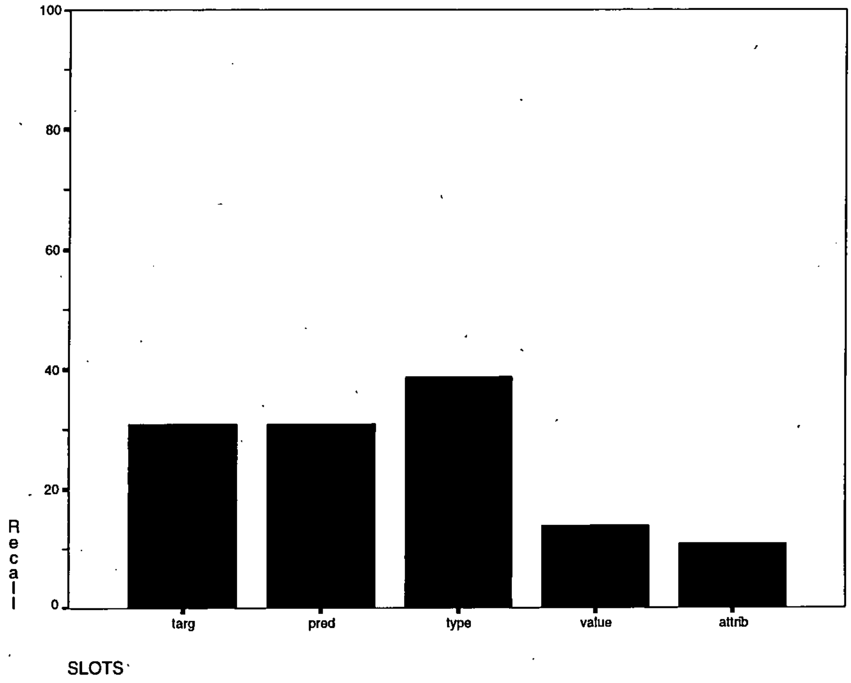

Figure 10.18: The recall for each of the slots of the user-defined takeover template.

Figure 10.19 and 10.18 show the precision and recall measures respectively for each of the slots of the user-defined takeover templates.

We can compare these results with the evaluation of the pre-defined takeover template discussed in section 10.2.4. The results of the pre-defined takeover template can be summarised in the following data:

	P&R	2P&R	P&2R
F-MEASURES	51.03	57.41	45.93

OVERALL PRECISION:	63%
OVERALL RECALL:	43%

The performance of the pre-defined takeover template is therefore higher than the equivalent user-defined takeover template. If we analyse each of the slots separately, the figures are similarly comparable. The overall loss of performance of the user-definable takeover template compared to the pre-defined takeover template is

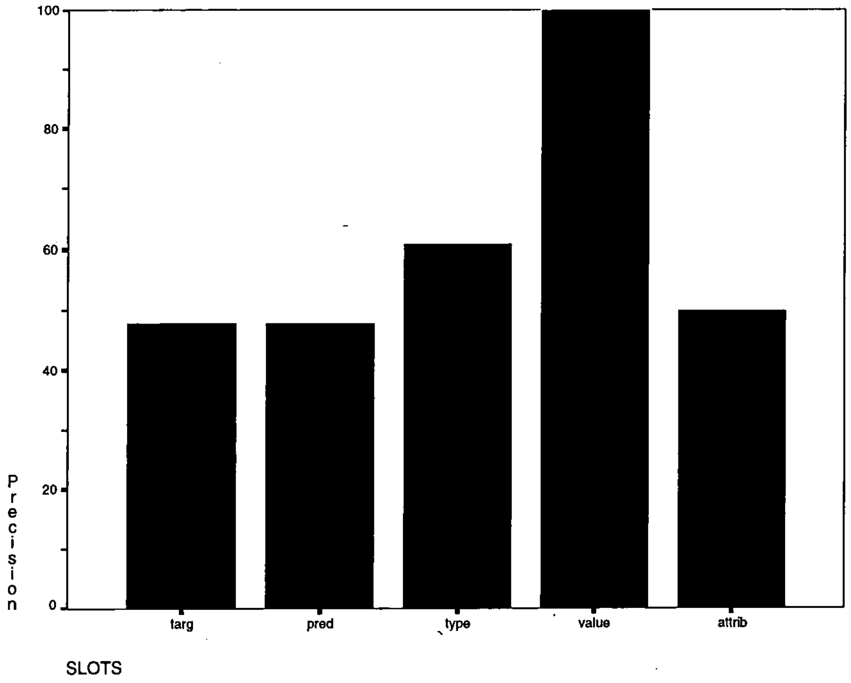

Figure 10.19: The precision for each of the slots of the user-defined takeover template.

therefore:

$$Loss\ of\ performance\ = 100 - \frac{F\text{-}Measure\ user\text{-}definable\ templates}{F\text{-}Measure\ pre\text{-}defined\ templates} \times 100$$

$$Loss\ of\ performance\ = 100 - \frac{37.44}{51.03} \times 100 = 26.63\%$$

The user-defined takeover template presents a loss of performance of 27% compared to the pre-defined template.

The loss of performance is mainly due to the difference in the way the user-defined templates are produced by the system (see chapter 9). While the main-events and slot-rules definitions for the pre-defined takeover templates are directly coded in the system, the equivalent definitions for the user-definable template are instead obtained from the analysis of the source text of the template definition. This additional step relies directly on the analysis of the source text performed by the LOLITA core system. If a main-event or slot-rule definition is not correctly processed by the LOLITA core system, the user-definable interface will be unable to correctly identify the relevant information in the source article.

Although the drop in performance (27%) is significant, the time taken for the development of both the pre-defined and the user-definable templates must be taken into account.

The pre-defined takeover template has been defined and coded within the LOLITA system in a period of time of about 8 months. This period of time included understanding how to code new template definitions in the LOLITA system, identifying the relevant rules for the takeover template, coding, compiling and testing the template definition.

Differently, the implementation of the user-defined takeover template, once the user-definable interface had been coded in the LOLITA system, required a significantly lower amount of time, which can be quantified in a total of 0.5 months.

We can therefore compare the two figures as follows:

$$\frac{time\ for\ defining\ the\ user-defined\ takeover\ template}{time\ for\ defining\ the\ pre-defined\ takeover\ template} = \frac{0.5\ months}{8\ months} \times 100 = 6.25\%$$

The above figures show that defining the takeover template using the user-definable template interface required only 6.25% of the time for defining the pre-defined template and lead to a loss of performance of 27%. A trade-off between the time saved defining the templates employing the user-definable template interface and the loss of performance of the user-definable template over the predefined hand-coded one can therefore be identified.

These figures show that the user-definable templates are a feasible way of defining new templates within the LOLITA system and require a time which is much lower than for coding the new templates within the system, which justifies the loss of performance. The user-definable template interface can be useful for the financial operator who can define a new template in a very limited time and extract relevant information from a large quantity of source texts.

10.3.5 Evaluation of the non-interactive user-defined template against its hand-coded version

In this section we discuss the evaluation of the *non-interactive* user-defined takeover template shown in figure 10.16 and its corresponding hand-coded version (see section 10.3.2). The evaluation of the two *non-interactive* templates provides an indication of the performance of the system extracting templates defined by new users of the system and the loss of performance of the user-defined template against its hand-coded version.

The evaluation is based on the same evaluation articles and evaluation measures as the evaluation of the other takeover templates discussed in sections 10.2.4 and 10.3.4. Figure 10.20 shows the performance of the user-defined *non interactive* template for the 55 financial articles of the evaluation set. The final results show that the system's overall performance were:

	P&R	2P&R	P&2R
F-MEASURES	22.56	36.06	16.41
OVERALL PRECISION:	60%		
OVERALL RECALL:	14%		

Figure 10.21 shows the performance of the *non-interactive* template which has been hand-coded in the LOLITA system for the 55 financial articles of the evaluation set. The final Results show that the system's overall performance was:

	P&R	2P&R	P&2R
F-MEASURES	33.11	44.64	26.32
OVERALL PRECISION:	58%		
OVERALL RECALL:	23%		

The performance of the *non-interactive* user-defined takeover template is therefore lower than the hand-coded version. The overall loss of performance of the

```
Report for the non-interactive user-definable templates finalEvalAVUD

------------------------+-------------+--------------+-----------------------
SLOT            POS   ACT| COR PAR INC | MIS  SPU  NON| REC PRE UND OVG ERR SUB
------------------------+-------------+--------------+-----------------------
takeover         36   12|  10   0    1|  25    1    0|  28  83  69   8  73   9
  companytar     36   12|   6   0    5|  25    1    0|  17  50  69   8  84  45
  companypre     35   12|   9   0    2|  24    1    0|  26  75  69   8  75  18
  price          28    1|   0   0    0|  28    1    0|   0   0 100 100 100   0
  attrib          9    0|   0   0    0|   9    0    0|   0   0 100   0 100   0
------------------------+-------------+--------------+-----------------------
ALL OBJECTS     108   25|  15   0    7|  86    3    0|  14  60  80  12  86  32
------------------------+-------------+--------------+-----------------------
                                                P&R       2P&R       P&2R
F-MEASURES                                     22.56      36.06      16.41
```

Figure 10.20: The final score report for the *non-interactive* user-defined takeover financial template.

```
Report for the non-interactive user-definable templates finalEvalAV

------------------------+-------------+--------------+-----------------------
SLOT            POS   ACT| COR PAR INC | MIS  SPU  NON| REC PRE UND OVG ERR SUB
------------------------+-------------+--------------+-----------------------
takeover         36   32|  21   0    3|  12    8    0|  58  66  33  25  52  13
  companytar     36   19|  12   0    5|  19    2    0|  33  63  53  11  68  29
  companypre     35   24|  13   0    3|  19    8    0|  37  54  54  33  70  19
  price          28    0|   0   0    0|  28    0    1|   0   0 100   0 100   0
  attrib          9    0|   0   0    0|   9    0    0|   0   0 100   0 100   0
------------------------+-------------+--------------+-----------------------
ALL OBJECTS     108   43|  25   0    8|  75   10    1|  23  58  69  23  79  24
------------------------+-------------+--------------+-----------------------
                                                P&R       2P&R       P&2R
F-MEASURES                                     33.11      44.64      26.32
```

Figure 10.21: The final score report for the *non-interactive* takeover template hand-coded in the system.

non-interactive user-defined takeover template compared to the hand-coded version is therefore:

$$loss\ of\ performance = 100 - \frac{F-Measure\ user-definable\ templates}{F-Measure\ pre-defined\ templates} \times 100$$

$$loss\ of\ performance = \frac{22.56}{33.11} \times 100 = 31.86\%$$

The user-defined takeover template presents a loss of performance of 32% compared to the pre-defined template. Although this figure can appear quite significant at a first sight, the time taken for the development of both the pre-defined template and the user-definable takeover templates must be taken into account. As we already described in section 10.3.2 the time for entering the template definition using the user-definable interface was 15.71 minutes, while the time for hand-coding the equivalent template definition in the LOLITA system was equal to 6.25 hours.

The lower performance of the *non-interactive* user-defined template definition compared to the "interactive" takeover template described in section 10.3.4 is due to two main reasons: the limitation of the core system in the analysis of the source text and its limited world knowledge, in particular of the financial domain. An example of specific knowledge in the financial domain is the fact that *taking full control of a company* implies a *takeover*. These two elements require the user to enter more sentences for defining the main-events and slot-rules of the templates. Looking at the *non-interactive* template definition shown in figure 10.16 and the "interactive" takeover template shown in figure 8.5 it is possible to notice the higher number of main-event and slot-rule definitions of the interactive template compared to the *non-interactive* template, which justifies the better performance of the interactive template definition.

10.4 Definition of the other financial templates

A full evaluation of the results of the user-definable template interface described in this work has been carried out only employing definition of takeover templates, both *interactive* and *non-interactive*. In this section we present the templates definitions for the other financial templates described in chapter 6. These templates have been designed by the author in the average of 10 minutes for each of the templates, which is a very limited time. A complete evaluation of each of the template definition would require the selection of a set of 55 relevant and non-relevant articles and the preparation of a corresponding set of template-keys. Moreover, the *drop of performance* of the new user-defined financial templates could have been measured only implementing a corresponding definition in the LOLITA system. Such an evaluation has been considered beyond the scope of this work. Figure 10.22 shows an example template produced using the user-defined merger template definition.

- **Merger template:**

```
Template_Name:          T=MERGER

Variables:              V=COMPANY1 is a company.
                        V=COMPANY2 is a company.
                        V=COMPANY3 is a company.

Template main-events:   V=COMPANY1 merged with V=COMPANY2 creating.
                        V=COMPANY3.
                        V=COMPANY1 merged with V=COMPANY2.

Definition of the slots:
S=FIRST-COMPANY:        V=COMPANY1
S=SECOND-COMPANY:       V=COMPANY2
S=NEW-NAME:             V=COMPANY3
S=DATE-OF-ANNOUNCE:     the date when T=MERGER is announced.
S=DATE-OF-MERGER:       the date when T=MERGER takes place.
S=ATTRIBUTION:          the company that announced T=MERGER.
                        the person that announced T=MERGER.
```

- **Flotation template:**

```
Template_Name:          T=FLOTATION

Variables:              V=COMPANY1 is a company.

Template main-events:   V=COMPANY1 has been floated in the
                        stock exchange.

Definition of the slots:
S=COMPANY-NAME:         V=COMPANY1
S=PRICE:                the price of the shares offered by V=COMPANY1.
S=VALUE:                the value of the shares offered by V=COMPANY1.
```

```
S=ANNOUNCE-DATE:          the date when T=FLOTATION is announced.
S=LISTING-DATE:           the date when T=FLOTATION takes place.
S=FINANCIAL-ADVISER:      the adviser of T=FLOTATION.
                          the adviser of V=COMPANY1.
S=ATTRIBUTION:            the company that announced T=FLOTATION.
                          the person that announced T=FLOTATION.
S=DENIAL:                 the person who denied T=FLOTATION.
                          the company who denied T=FLOTATION.
S=INDUSTRY-SECTOR:        the industry sector of V=COMPANY1.
```

- **New Issue template:**

```
Template_Name:            T=NEW-ISSUE

Variables:                V=COMPANY1 is a company.
                          V=BONDS are bonds.

Template main-events:     V=COMPANY1 issued new V=BONDS.

Definition of the slots:
S=COMPANY-NAME:           V=COMPANY1
S=FINANCIAL-ADVISER:      the adviser of T=NEW-ISSUE.
                          the adviser of V=COMPANY1.
S=ISSUE-VALUE:            the price of V=BONDS.
S=ANNOUNCE-DATE:          the date when T=NEW-ISSUE is announced.
S=LAUNCH-DATE:            the date when T=NEW-ISSUE takes place.
S=LISTED:                 the Stock Exchange where V=BONDS are
                          offered.
S=ATTRIBUTION:            the person who announced T=NEW-ISSUE.
                          the company who announced T=NEW-ISSUE.
S=PURPOSE                 the purpose of V=COMPANY1.
S=DENIAL                  the person who denied T=NEW-ISSUE.
                          the company who denied T=NEW-ISSUE.
```

- **Privatisation template:**

```
Template_Name:            T=PRIVATISATION

Variables:                V=COMPANY1 is a company.
                          V=STAKE is a company stake.

Template main-events:     V=COMPANY1 has been privatised selling V=STAKE.

Definition of the slots:
S=COMPANY-NAME:           V=COMPANY1
S=STAKE-PRIVATISED:       V=STAKE
S=PRICE-OF-SHARES:        the price of V=STAKE.
S=VALUE:                  the value of V=STAKE.
S=ANNOUNCE-DATE:          the date when T=PRIVATISATION is announced.
S=PRIVATISATION-DATE:     the date when T=PRIVATISATION takes place.
```

S=ADVISER:	the adviser of T=PRIVATISATION. the adviser of V=COMPANY1.
S=ATTRIBUTION:	the person who announced T=PRIVATISATION the company who announced T=PRIVATISATION
S=DENIAL:	the person who denied T=PRIVATISATION the company who denied T=PRIVATISATION
S=INDUSTRY-SECTOR:	the industry sector of V=COMPANY1.

- **Bankruptcy template:**

Template_Name:	T=BANKRUPTCY
Variables:	V=COMPANY1 is a company.
Template main-events:	V=COMPANY1 has been declared bankrupt.

Definition of the slots:

S=COMPANY-NAME:	V=COMPANY1
S=RECEIVERS:	the receivers of V=COMPANY1. the receivers of T=BANKRUPTCY.
S=DATE-ANNOUNCE:	the date when T=BANKRUPTCY is announced.
S=DENIAL:	the person who denied T=BANKRUPTCY. the company who denied T=BANKRUPCTY.

- **New stake template:**

Template_Name:	T=NEW-STAKE

Variables:	V=COMPANY1 is a company.
	V=COMPANY2 is a company.
	V=STAKE is a company stake.

Template main-events:	V=COMPANY1 bought a V=STAKE of V=COMPANY2.

Definition of the slots:

S=COMPANY-TARGET:	V=COMPANY2
S=COMPANY-NAME:	V=COMPANY1
S=VALUE:	the value of V=STAKE.
S=ATTRIBUTION:	the company who announced T=NEW-STAKE. the person who announced T=NEW-STAKE.
S=DENIAL:	the company who denied T=NEW-STAKE. the person who denied T=NEW-STAKE.

- **Dividend Announcement template:**

Template_Name:	T=DIVIDEND-ANNOUNCEMENT

Variables:	V=COMPANY1 is a company.
	V=DIVIDEND is a dividend.

```
Template main-events:   V=COMPANY1 announced V=DIVIDEND

Definition of the slots:
S=COMPANY:              V=COMPANY1
S=DIVIDEND:             V=DIVIDEND
S=TYPE-OF-DIVIDEND:     the type of V=DIVIDEND.
```

- **Overseas listing template:**

```
Template_Name:          T=OVERSEAS-LISTING

Variables:              V=COMPANY1 is a company.
                        V=STOCK-EXCHANGE is a stock exchange.

Template main-events:   V=COMPANY1 has been listed at
                        V=STOCK-EXCHANGE

Definition of the slots:
S=COMPANY:              V=COMPANY1
S=STOCK-EXCHANGE:       V=STOCK-EXCHANGE
S=TYPE-SECURITIES:      the type of securities listed by V=COMPANY1.
S=ANNOUNCE-DATE:        the date when T=OVERSEAS-LISTING is
                        announced.
S=ATTRIBUTION:          the company who announced T=OVERSEAS-LISTING.
                        the person who announced T=OVERSEAS-LISTING.
S=DENIAL:               the company who denied T=OVERSEAS-LISTING.
                        the person who denied T=OVERSEAS-LISTING.
```

10.5 Conclusions

From the evaluation carried out in this chapter we can draw the conclusion that the financial operator can benefit from using the system reducing the qualitative data overload for the following reasons:

- the pre-defined financial takeover template showed the overall performance of 51% which would allow the financial operator to use the system for quickly extracting relevant information from a large collection of source articles;

- the user-definable template interface has been found easy to use by the 14 potential end-users of the system which were able to define takeover templates with only 5.71% of information wrongly defined;

An example relevant article for the merger template

Frontier Corporation, formerly Rochester Telephone, has merged with ALC Communications to form the fifth-biggest US long-distance telephone service group in a deal valued at about 1.8 billion dollars. Frontier is a local telephone company which has diversified into long-distance and wireless communications services. It has long-distance revenues of about 500 million dollars. ALC, through its wholly-owned subsidiary Allnet, offers long-distance services to small and medium-sized business customers and has recently announced cellular and paging services. The merger will create a company with consolidated revenues approaching 2 billion dollars and long-distance revenues of about 1.4 billion dollars. ALC shareholders will receive two shares of Frontier stock for each share of ALC's, giving Frontier investors a 49 per cent stake in ALC. 'By combining the rapidly expanding long-distance businesses at Frontier and necessary for long-term success in this industry,' said Mr Ronald Bittner, Frontier chairman and chief executive.

The template extracted by the system using the merger template definition.

```
<T=MERGER-0002020000-96979> :=
    S=SECOND-COMPANY: "ALC Communications. "
    S=FIRST-COMPANY:  "Frontier Corporation. "
```

Figure 10.22: An example relevant article and merger template extracted by the system.

- the performance of the user-definable template interface presents a loss of performance of 27% for the *interactive* takeover template. However, the time needed for designing the user-defined interactive takeover template was only the 6.25% of the pre-defined takeover template. This would allow a financial operator and expert user of the system to design a template in a very short time with a performance which would still allow him to profit from using the application;

- the *non-interactive* user-defined takeover template presents a loss of performance of 32% compared to the hand-coded version of the template, but required only 15.71 minutes for designing the definition, which is 24 times lower than the time taken for coding the template in the system. The financial operator can therefore quickly enter a first definition of a template and analyse a large amount of source articles. After the first results, he can improve the template definition sensibly obtaining sensibly higher results.

From the evaluation of the performance of the user-definable takeover templates, we can conclude that the interface is an effective way for defining new templates in the LOLITA system. Although the loss of performance of the user-defined templates can appear relevant at a first sight, the time required for defining a template using the user-definable template is much lower, which justifies the loss of performance. Moreover, the user can improve the performance by adding additional definitions for the main-event conditions and the slot-rules. The user-definable template interface can therefore be a useful tool for the financial operator who can quickly enter a template definition for extracting financial information from a large collection of source texts, which would otherwise be lost due to the lack of time of the financial operator.

The financial operator could benefit from using the system for extracting relevant information in large collections of financial articles. The templates extracted by the system would allow the operator to gain knowledge which would otherwise be lost due to the lack of time for analysing large collection of financial articles. This knowledge can be extremely relevant for taking appropriate investment decisions.

Chapter 11

Conclusions and future work

This chapter concludes the thesis by discussing the main achievements of this thesis and presenting possible future research developments of this work.

The first achievement of the thesis has been to propose and demonstrate the feasibility of a method of allowing inexperienced users to easily define and extract new templates from a large collection of source articles using sentences in natural language. The thesis has therefore shown that user-defined information extraction is possible within certain constraints. This work has also shown that the general-purpose LOLITA Natural Language Processing System can be successfully used for information extraction tasks in the field of finance with minor changes to the core system. Finally, the thesis has given an indication on the fact that financial operators can profit from using information extraction to reduce their qualitative data overload, extracting relevant information from a large collection of source articles.

11.1 Aim 1: implementation of templates for extracting relevant information from financial articles

A set of pre-defined financial templates has been defined (see chapter 6) to represent relevant information for supporting the financial operator's investment decision making process. The templates definitions have not been formally evaluated since a clear methodology for the evaluation of templates definitions has not been yet defined in the literature.

The pre-defined takeover financial template has been implemented in the system (see chapter 7) and the performance of the template has been evaluated in section 10.2. The overall performance of the system in extracting relevant takeover templates from 55 source articles was 51% (*'F' measure*) which was obtained without any specific improvement of the LOLITA system.

Although direct conclusions can only be drawn for the takeover template, the performance of this template can provide an indication of how the system could perform extracting other financial templates described in chapter 6.

The financial operator could profit from using the system reducing his qualitative data overload by quickly extracting relevant financial information from large collections of financial articles. The templates extracted by the system would allow the operator to gain knowledge which would otherwise be lost due to the lack of time for analysing large collection of financial articles. This knowledge can be extremely relevant for taking appropriate investment decisions.

Although the system can potentially extract wrong information, it is unlikely that this can affect the financial operator's investment decisions. This is because financial operators involved in specific market sectors have a relevant knowledge of the specific companies and players and can therefore easily identify information which is relatively unlikely or misleading, for example a takeover template where the company predator and company target have been swapped.

11.2 Aim 2: design and implementation of a user-definable template interface

The templates based on the *financial activities* approach should be able to identify the information which is relevant for the financial operator's investment decision-making process. However, since no direct link can be established between an event and a share price movement, the user might want to personalise the system adding new templates to suit his specific needs.

The LOLITA template user-definable interface has been designed to allow the user to add new templates to the collection of existing ones using natural language sentences with few restrictions and formal elements (see chapters 8 and 9).

The evaluation of the *ease of use* of the interface has been carried out analysing the templates definitions entered by potential users of the system. A group of 14 new users of the system were able to design new templates definitions in only 15.71 minutes with only 5.71% of the information wrongly defined. The time and error rate are extremely low, especially considering that the 14 users were mostly unfamiliar with concepts such as natural language processing, information extraction and user-definable template interfaces.

The *loss of performance* of the user-definable templates interface compared to the pre-defined takeover templates has been determined analysing the results of two separate comparisons. The performance of the *interactive* takeover template shown in figure 8.5 has been compared with the performance of the *interactive* pre-defined takeover template presented in chapter 7. The *loss of performance* of the user-definable template was 27%. However, the definition of the user-definable template required a time 16 times lower than coding the template in the system. The performance of a *non-interactive* takeover template chosen among 14 takeover definitions entered by potential users of the system has been compared to the performance of the same template directly coded in the system. In this case the loss of performance was 32%. However, the definition of the user-definable template required a time at least 24 times lower than coding the template in the system. In addition, the definition of user-definable templates can be carried out by end users with no prior knowledge regarding natural language or programming skills.

From the data described above, we can conclude that the user-definable template interface has been successful both in terms of *ease of use* and in terms of *performance* compared to pre-defined templates directly coded in the system. The user-definable template interface represents therefore an effective approach to customizable information extraction.

A new user of the system can in fact design a new template definition on average in only a quarter of an hour and quickly extract templates from a large collection of source articles. The user can subsequently improve the template definition reducing the loss of performance compared to templates definitions hand-coded in the system.

The user-definable template interface can therefore help the financial operator reduce his qualitative data overload. The operator can in fact very quickly define a new template definition and extract relevant information from a large collection of source articles. Such information would otherwise be lost due to the lack of time of the financial operator.

While no limitations could be established regarding the *ease of use* of the interface, the templates produced with the user-definable template interface present a *loss of performance* compared to hand-coded templates. This is due to limitations of the LOLITA core system and not of the interface itself. The *loss of performance* could therefore be reduced by improving the capabilities of the LOLITA core system, particularly the world knowledge of the system.

11.3 Impact on the field of information extraction

The most important contribution of this thesis on the field of information extraction is the possibility given to the user of the system to easily define new templates definitions using natural language sentences. The interface overcomes the limitations of existing information extraction systems which do not allow the user to personalise the system by adding new templates definitions. The templates must in fact be coded in the system directly by the developers.

The user-definable template interface described in this work overcomes this problem by allowing the user to quickly add new templates definitions using sentences in natural language. The evaluation has shown that the interface is particularly easy to use, which justifies the loss of performance of the user-defined templates compared to hand-coded templates definitions.

11.4 Impact on the field of finance

The most important contribution of this thesis in the field of finance is the reduction of the financial operator's the qualitative data-overload.

By processing large quantities of financial articles, the financial operator can quickly obtain a summary of the relevant information without having to read the whole article. The financial information extraction system acts therefore as a filter for the news, eliminating those that do not satisfy any of the extraction criteria and producing a template for the relevant information, which can also be defined by the financial operator with the user-definable template interface.

This allows the operator to gain knowledge which can be extremely important for taking appropriate investment decisions and could otherwise be lost due to the lack of time of the financial operator.

11.5 Project Limitations and Suggestions for Further Work

This section describes some of the limitations of the project and suggests areas of further research.

11.5.1 Implementation of the other financial templates

A first improvement of the system would be a complete implementation of the other financial templates described in chapter 6 and a complete evaluation of the performance of the system extracting information corresponding to these templates. The templates have been defined using the user-definanable template interface (see section 10.4). However, the evaluation of these templates definitions has not been performed. This is because the evaluation of the new templates definitions would have required a considerable amount of time and resources. A complete evaluation of each of the template definition, in fact, would require the selection of a set of 55 relevant and non-relevant articles and the preparation of a corresponding set of template-keys. Moreover, the *drop of performance* of the new user-defined financial templates could have been measured only implementing a corresponding definition in the LOLITA system. Such an evaluation has been considered beyond the scope of this work.

11.5.2 Automatic extraction of templates

A possible improvement of the system could be the integration of the user-definable template interface with the automatic template creation system proposed by Collier [Collier, 1994, Collier, 1996]. Differently from the user-definable template interface, Collier's system is able to automatically create a template definition from a set of source documents. The system, however, does not allow the user to modify the templates definitions or to extract the corresponding information from source texts.

The integration between the two systems would be therefore interesting. The new system would first process a set of source documents producing a possible template definition using Collier's approach. The user could subsequently edit, modify or substitute the templates definitions using the user-definable template interface. The templates definitions would then be used for extracting relevant information from the source texts using the user-definable template interface.

This approach would be particularly interesting for the financial operator. Differently from the solution described in this work, the new system would in fact be able to suggest a possible template definition to the financial operator for a specific set of articles. The template definition could be then subsequently improved or substituted by the operator. This approach could help the operator reduce the time needed for locating relevant information within a selected collection of documents.

Appendix A

Glossary

Anaphora resolution: also called *reference resolution*, is the process of relating one concept to another. For example "*FIAT bought RENAULT*". It costed 10 million dollars" "*it*" must has to be associated to "*RENAULT*" rather than to "*FIAT*" or the event itself.

Coreference: a situation in which two objects are referring to the same object (e.g. "the car" and ... "it uses fuel"). A NLP System should be able to understand that these two objects are related (coreference).

Domain-specific knowledge: knowledge which is specific to the particular area or situation and can differ from the general knowledge which is commonly employed by the system.

Expert System: a "knowledge-based system that emulates expert thought to solve significant problems in a particular domain of expertise" [Sell, 1986].

Explanatory financial tool: financial tools which are used for explaining the movements of share prices but cannot be used for the prediction of future prices.

Event-based templates: templates where it is possible to identify a clrea underlying top-level event to which all the information of the template's slots can be referred to.

Fallout: ratio between the number of non relevant items retrieved and the total number of non relevant items in the collection.

Financial activity: an event which is likely to influence the price of shares and, therefore, influence the decision-making process of the players of the market regarding these securities [Costantino *et al.*, 1996c].

Forecasting financial tool: financial tools which are used to predict the future prices of shares (e.g. Neural Network).

General support financial tool: financial tools which are used as support to the analysis of the financial operator or to pre-process the information, but cannot be used for explaining movements of share prices or for predictions.

Generator: the part of a NLP system that translates the internal representation into natural language. It is usually the last module of a natural language processor.

Hyper-templates: structures whose slots can refer to other templates, thus creating a linked chain of templates (e.g. the MUC-6 scenario templates).

Inference: inference "is taken as the process by which a listener or reader draws out those parts of the meaning of a piece of natural language which are not manifest in the text itself" [Tait, 1982]. Inference can be *explanatory* if it explains what was before in the text, or *predicative* if it explains the subsequent input.

Information extraction: the goal of information extraction is to extract specific types of information from a document [Riloff and Lehnert, 1994].

Information retrieval: the goal of information retrieval is normally to retrieve relevant documents from a large collection of various kinds of documents (newspaper articles, legal documents etc.) given a particular query.

LOLITA Topic: a list of suggested meanings which are specific to the domain of the text being analysed by the system and are used to disambiguate between possible meanings by the pragmatic analysis module.

Neural Networks: neural networks have been created with the purpose of simulating the functionalities of the brain and are based on a collection of neurons connected to one another with specific *weights.* They are normally employed in finance for the forecasting of securities prices.

Noun Phrases: groups of words which describe particular objects or events but do not form a complete sentence, for example "The company that acquired the other company".

Parser: the part of a NLP system that determines the kind of information that is contained in a text [McDonald, 1992].

Polysemy: words can have multiple meanings. For example, the word "post" can refer to the name of a newspaper, a vertical support in carpentry, entering a transaction in accounting, or sending a message in computing [Riloff and Lehnert, 1994].

Pragmatic analysis: the process of checking whether the new semantic information produced by the semantic analysis module is consistent with the one already available in the semantic network

Precision: ratio of the number of relevant documents retrieved to the total number of documents retrieved [Rijsbergen, 1979].

Qualitative financial tools: tools which are used to produce quantitative or qualitative information from quantitative data (data which can be expressed in a numeric format).

Qualitative financial tools: tools which are able to process qualitative data and produce qualitative or quantitative information.

Query: the requirements that have to be matched by the documents in the collection in order to be judged relevant. Queries are normally used in information retrieval and rarely in information extraction. They can be either of artificial kind (e.g. SQL) or expressed in natural language.

Recall: ratio of the number of relevant documents retrieved to the total number of relevant documents (both retrieved and not retrieved) [Rijsbergen, 1979].

Semantic analysis: the process of mapping the information produced by the parser onto nodes in the semantic network.

Semantic network: the part of a NLP system that contains the system's knowledge of the world.

Semantic pattern: patterns which refer to the meaning of a sentence rather than on its form or words.

Slot-name (user-definable template interface): a string defined according to a specific syntax which is used to identify each slot of a user-definable template. The slot-names can be used in the definition of other slots to refer to information contained in slots which have already been defined.

Slot-rule (user-definable template interface): sentences in natural language with few restrictions and formal elements used to define the relevant information for each of the slots which will be extracted by the system from the source articles.

Suffix (or word) stripping: the suffix of a word is removed from it, obtaining the stem that can be directly used as a search-term or can be normalised.

Summary templates: templates for which it is not possible to identify an underlying main-event. A summary template is basically a collection of objects stored in different slots which may not directly refer to the same concept or relate to each other.

Synonymy: different words and phrases can express the same concept [Riloff and Lehnert, 1994].

Template: a structure with a predefined set of slots, one for each type of information to be extracted from a text [Riloff and Lehnert, 1994].

Template main-event (user-definable template interface): the condition under which a user-definable template is produced by the system. The system matches each main-event against possible events in the source text and for each event which is successfully matched, a template is instantiated.

Template-name (user-definable template interface): a string defined according to a specific syntax which uniquely identifies a template within the system and can be used in the definition of slot-rules to refer to the template as a whole.

True-False assertions: sentences which explain a condition which can be True or False, for example "Company A acquired company B".

Unification: the process of unifying two objects or events which refer to the same concept.

User-definable Template Interface: user-interface which allows the definition of new templates using sentences in natural language with few restrictions and formal elements.

Variable (user-definable template interface): specific elements which are used to identify relevant information within the main-events. The variables are used in the definition of the slots to extract information identified in the main-events.

Word normalisation: the stem is re-converted into the general form of the term, for example verbs are normalised into their infinitive form. This is usually done using a dictionary.

Appendix B

User-Defined takeover templates

This appendix contains specific data which have been employed for the evaluation of the results of this work. In section B.1 specific articles and keys from the evaluation set for the evaluation of the pre-defined takeover templates are reported. Section B.2 contains contains the takeover templates submitted by the potential users which have been used for the evaluation of the user-definable template interface (see section 10.3). The templates are ordered depending on the category of the users. The information regarding the identity of the users has not been reported, since it is not relevant for the evaluation of the templates.

B.1 Sample data for evaluation of the pre-defined takeover template

Cowie Group, the car leasing and motor trading company, yesterday announced a big expansion of its bus operations with the 29.9 million pounds acquisition of Leaside Bus Company, the subsidiary of London Regional Transport (LRT). The deal, involving a 25.5 million pounds cash payment and 4.4 million pounds to settle intra-group loans, will enlarge Cowie's bus fleet from 128 vehicles to more than 600 and is expected to lead to a fourfold sales increase.

'We paid slightly more than we wanted to, but it was worth it for the enormous growth that it promises,' said Mr Gordon Hodgson, chief executive. The acquisition follows four months of talks between LRT and Cowie, which has been seeking a larger stake in the London bus network for more than two years.

At present, the group's bus and coach operations are dominated by Grey-Green - acquired 14 years ago - which serves 13 bus routes in London and employs 450 drivers. Leaside, by comparison, has a workforce of about 1,800 and operates 28 routes.

Mr Hodgson, who is meeting Leaside managers today, said he was determined to introduce private sector efficiency to the business, which last year made profits of just 607,000 pounds on turnover of 43 million pounds. In the same period, Grey-Green made profits of 1.6 million pounds on sales of 14.4 million pounds. Cowie shares fell 3 1/2 p to 218 1/2 p yesterday - a new low for the year.

Figure B.1: A relevant article for the evaluation set.

Sappi Europe, the European arm of the South African forest products group, is to invest 61 million pounds (95 million dollars) to increase its paper-making capacity in Germany and the UK. The move comes in response to forecast growth in European demand for coated wood-free papers - used for quality magazines, brochures and company reports. Mr Franz Neudeck, chief executive, said: 'Coated wood-free is one of our core products and in spite of the recent apparent weak demand - a result of the destocking after the speculative levels early in 1995 - we expect demand for coated wood-free papers to maintain a real market growth of more than 5 per cent per annum.' The investments will raise the company's output of coated wood-free paper by 110,000 tonnes to 500,000 tonnes a year by early 1997 - a 28 per cent increase.

Figure B.2: A non-relevant article of the evaluation set.

```
<TAKEOVER-0002120000-1> :=
   COMPANY_TARGET: "Leaside Bus Company. "
   COMPANY_PREDATOR: "Cowie Group. "
   VALUE: "25.5 million pounds. "
   ATTRIBUTION: "Mr. Hodgson"
   TYPE_TAKEOVER: FRIENDLY
```

Figure B.3: The key-file for the article from the evaluation set in figure B.1

B.2 Data for the evaluation of the user-definable takeover template

Category of users: 1

- **User 1.1:**

```
Template_Name:          T=Takeover
Variable_1:             V=COMPANY1 is a company
Variable_2:             V=COMPANY2 is a company
Main_Event_1:           V=COMPANY1 bought V=COMPANY2
Slot_Name_1:            S=FIRST-COMPANY
Slot_Rule_1.1:          V=COMPANY1
Slot_Name_2:            S=SECOND-COMPANY
Slot_Rule_2.1:          V=COMPANY2
Slot_Name_3:            S=TAKEOVER-DATE
Slot_Rule_3.1:          date of takeover
Slot_Name_4:            S=TAKEOVER-COST
Slot_Rule_4.1:          cost of taking over the company
Slot_Name_5:            S=DATE-ANNOUNCE
Slot_Rule_5.1:          the date when the takeover is announced
Time_Instructions:      25 min
Time_Form:              5 min
```

- **User 1.2:**

```
Template_Name:          T=TAKEOVER
Variable_1:             V=C1 is a company
Variable_2:             V=C2 is a company
Variable_3:             V=D is a date
Variable_4:             V=C is money
Main_Event_1:           V=C1 bought V=C2
Main_Event_2:           V=C1 bought V=C2 for V=C
Main_Event_3:           V=C1 bought V=C2 at V=D
Slot_Name_1:            S=Main
Slot_Rule_1.1:          V=C1
Slot_Name_2:            S=Taken-Over
Slot_Rule_2.1:          V=C2
Slot_Name_3:            S=COST
Slot_Rule_3.1:          V=C
Slot_Name_4:            S=Date
Slot_Rule_4.1:          V=D
Time_Instructions:      35 min
Time_Form:              10 min
```

- **User 1.3:**

```
Template_Name:          T=COMPANY-ACQUISITION
Variable_1:             V=THE-BUYER is a company
Variable_2:             V=COMPANY1 is a company
Main_Event_1:           V=THE-BUYER bought V=COMPANY1
Slot_Name_1:            S=COMPANY-ACQUIRING
Slot_Rule_1.1:          V=THE-BUYER
Slot_Name_2:            S=COMPANY-ACQUIRED
Slot_Rule_2.1:          V=COMPANY1
Slot_Name_3:            S=TIME-OF-ACQUISITION
Slot_Rule_3.1:          The date V=THE-BUYER purchased V=COMPANY1
Slot_Name_4:            S=NATURE-OF-BUSINESS-OF-BUYER
Slot_Rule_4.1:          The business that V=THE-BUYER deals with
Slot_Name_5:            S=NATURE-OF-BUSINESS-OF-COMPANY1
Slot_Rule_5.1:          The business that V=COMPANY1 deals with
Slot_Name_6:            S=PRICE
Slot_Rule_6.1:          The price V=THE-BUYER paid for V=COMPANY1
Slot_Name_7:            S=COMPANY1-PAST-RECORD
Slot_Rule_7.1:          The known past profits of V=COMPANY1
Time_Instructions:      30 min
Time_Form:              20 min
```

- **User 1.4:**

```
Template_Name:          T=TAKEOVER
Variable_1:             V=COMPANY1 is a company
Variable_2:             V=COMPANY2 is a company
Variable_3:             V=VALUE is a value
Variable_4:             V=OWNER is a company or a human
Main_Event_1:           V=COMPANY1 acquired V=COMPANY2 from V=OWNER for V=VAI
```

Main_Event_2:	V=COMPANY1 acquired V=COMPANY2
Main_Event_3:	V=COMPANY1 acquired V=COMPANY2 from V=OWNER
Main_Event_4:	V=COMPANY1 acquired V=COMPANY2 for V=VALUE
Slot_Name_1:	S=FIRST-COMPANY
Slot_Rule_1.1:	V=COMPANY1
Slot_Name_2:	S=SECOND-COMPANY
Slot_Rule_2.1:	V=COMPANY2
Slot_Name_3:	S=PREVIOUS-OWNER
Slot_Rule_3.1:	V=OWNER
Slot_Name_4:	s=PURCHASE-PRICE
Slot_Rule_4.1:	V=VALUE
Time_Instructions:	15 min
Time_Form:	5 min

- **User 1.5:**

Template_Name:	T=TAKEOVER
Variable_1:	V=COMPANY1 is a company
Variable_2:	V=COMPANY2 is a company
Main_Event_1:	V=COMPANY1 purchased V=COMPANY2
Slot_Name_1:	S=TAKEOVER-PRICE
Slot_Rule_1.1:	the amount paid by V=COMPANY1 for V=COMPANY2
Slot_Name_2:	S=COMPANY2-BUSINESS
Slot_Rule_2.1:	the main product produced by V=COMPANY2
Slot_Name_3:	S=COMPANY1-BUSINESS
Slot_Rule_3.1:	the main product produced by V=COMPANY1
Slot_Name_4:	S=COMPANY2-TURNOVER
Slot_Rule_4.1:	the annual turnover of V=COMPANY2
Slot_Name_5:	S=COMPANY2-PROFITS
Slot_Rule_5.1:	the annual profits of V=COMPANY2
Slot_Name_6:	S=ATTRIBUTION
Slot_Rule_6.1:	the person or company that announced T=TAKEOVER
Slot_Rule_6.2:	the person or company that said something about T=TAKEOVER
Time_Instructions:	20 min
Time_Form:	10 min

- **User 1.6:**

Template_Name:	T=COMPANY-TAKEOVER
Variable_1:	V=COMPANY1 is a company
Variable_2:	V=COMPANY2 is a company
Main_Event_1:	V=COMPANY1 acquired V=COMPANY2
Slot_Name_1:	S=FIRST-COMPANY
Slot_Rule_1.1:	V=COMPANY1
Slot_Name_2:	S=SECOND-COMPANY
Slot_Rule_2.1:	V=COMPANY2
Slot_Name_3:	S=OWNER1
Slot_Rule_3.1:	The owner or founder of S=FIRST-COMPANY
Slot_Name_4:	S=OWNER2

```
Slot_Rule_4.1:          The owner or founder of S=SECOND-COMPANY
Slot_Name_5:            S=COST
Slot_Rule_5.1:          The amount S=FIRST-COMPANY paid for S=SECOND-COMPANY
Time_Instructions:      35 mins
Time_Form:              10 mins
```

Category of Users: 2

- **User 2.1:**

```
Template_Name:          T=TAKEOVER
Variable_1:             V=COMPANY1 is a company
Variable_2:             V=COMPANY2 is a company
Variable_3:             V=COMPANY3 is a company
Variable_4:             V=PERSON1 is a person
Main_Event_1:           V=COMPANY1 acquires V=COMPANY2
Main_Event_2:           V=COMPANY3 sell V=COMPANY2
Main_Event_3:           V=PERSON1 sell V=COMPANY2
Slot_Name_1:            S=DATE-OF-ANNOUNCEMENT
Slot_Rule_1.1:          date of announcement of T=TAKEOVER
Slot_Name_2:            S=VALUE-OF-ACQUISITION
Slot_Rule_2.1:          the value of the T=TAKEOVER
Slot_Rule_2.2:          the value of V=COMPANY2
Slot_Name_3:            S=COMPANY-TARGET
Slot_Rule_3.1:          V=COMPANY2
Slot_Name_4:            S=COMPANY-PREDATOR
Slot_Rule_4.1:          V=COMPANY1
Slot_Name_5:            S=COMPANY-PREVIOUS-OWNER
Slot_Rule_5.1:          V=COMPANY3
Slot_Rule_5.2:          V=PERSON1
Slot_Name_6:            =DATE-OF-ACQUISITION
Slot_Rule_6.1:          date of T=TAKEOVER
Slot_Name_7:            S=ATTRIBUTION
Slot_Rule_7.1:          the person that announced T=TAKEOVER
Slot_Rule_7.2:          the company that announced T=TAKEOVER
Time_Instructions:      20 min
Time_Form:              20 min
```

- **User 2.2:**

```
Template_Name:          T=TAKEOVER
Variable_1:             V=COMPANY1 is a company
Variable_2:             V=COMPANY2 is a company
Variable_3:             V=PRICE is an amount of money
Variable_4:             V=DATE is a date
Main_Event_1:           V=COMPANY1 took over V=COMPANY2
Main_Event_2:           V=COMPANY1 acquired V=COMPANY2
```

Main_Event_3:	V=COMPANY1 paid V=PRICE for V=COMPANY2
Main_Event_4:	V=COMPANY1 announced the takeover on V=DATE
Slot_Name_1:	S=FIRST-COMPANY
Slot_Rule_1.1:	V=COMPANY1
Slot_Name_2:	S=SECOND-COMPANY
Slot_Rule_2.1:	V=COMPANY2
Slot_Name_3:	S=PRICE
Slot_Rule_3.1:	The price paid by V=COMPANY1 for taking over V=COMPANY2
Slot_Rule_3.2:	V=PRICE
Slot_Name_4:	S=DATE
Slot_Rule_4.1:	The date on which the T=TAKEOVER was announced
Slot_Name_5:	S=ADVISER
Slot_Rule_5.1:	The name of the firm advising V=COMPANY1
Time_Instructions:	30 mins
Time_Form:	25 mins

Category of Users: 3

- **User 3.1:**

Template_Name:	T=COMPANY-TAKEOVER
Variable_1:	V=COMPANY1 is a company
Variable_2:	V=COMPANY2 is a company
Main_Event_1:	V=COMPANY1 acquired V=COMPANY2
Slot_Name_2:	S=COMPANY-PREDATOR
Slot_Rule_2.1:	V=COMPANY1
Slot_Name_3:	S=COMPANY-TARGET
Slot_Rule_3.1:	V=COMPANY2
Slot_Name_4:	S=ATTRIBUTION
Slot_Rule_4.1:	the person who reported T=COMPANY-TAKEOVER
Slot_Name_5:	S=PRICE
Slot_Rule_5.1:	the price that S=COMPANY-PREDATOR paid for S=COMPANY-TARGET
Time_Instructions:	20 min
Time_Form:	20 min

- **User 3.2:**

Template_Name:	T=TAKE-OVER
Variable_1:	V=COMPANY1 is a company
Variable_2:	V=COMPANY2 is a company
Main_Event_1:	V=COMPANY1 acquired V=COMPANY2
Slot_Name_1:	S=DATE
Slot_Rule_1.1:	date of T=TAKE-OVER
Slot_Name_2:	S=JOBS
Slot_Rule_2.1:	job losses at V=COMPANY2

```
Slot_Name_3:          S=COMPANY-ONE
Slot_Rule_3.1:        V=COMPANY1
Slot_Name_4:          S=COMPANY-TWO
Slot_Rule_4.1:        V=COMPANY2
Time_Instructions:    15 min
Time_Form:            20 min
```

- **User 3.3:**

```
Template_Name:        T=COMPANY-TAKEOVER
Variable_1:           V=COMPANY1 is a company
Variable_2:           V=COMPANY2 is a company
Main_Event_1:         V=COMPANY1 acquires V=COMPANY2
Slot_Name_1:          S=ACQUIRING-COMPANY
Slot_Rule_1.1:        V=COMPANY1
Slot_Name_2:          S=ACQUIRED-COMPANY
Slot_Rule_2.1:        V=COMPANY2
Slot_Name_3:          S=DATE-OF-TAKEOVER
Slot_Rule_3.1:        the date when T=COMPANY-TAKEOVER takes place
Slot_Name_4:          S=DATE-OF-ANNOUNCE
Slot_Rule_4.1:        the date when T=COMPANY-TAKEOVER is announced
Slot_Name_5:          S=ATTRIBUTION
Slot_Rule_5.1:        the person or company that announced
                      T=COMPANY-TAKEOVER
Slot_Rule_5.2:        the person or company that said something about
                      T=COMPANY-TAKEOVER
Slot_Name_6:          S=PRICE
Slot_Rule_6.1:        the money that V=COMPANY1 paid to acquire V=COMPANY2
Time_Instructions:    30 min
Time_Form:            30 min
```

- **User 3.4:**

```
Template_Name:        T=TAKEOVER
Variable_1:           V=COMPANY1 is a company.
Variable_2:           V=COMPANY2 is a company.
Main_Event_1:         V=COMPANY1 acquired V=COMPANY2
Main_Event_2:         V=COMPANY2 acquired V=COMPANY1
Slot_Name_1:          S=FIRST-COMPANY
Slot_Rule_1.1:        V=COMPANY1
Slot_Name_2:          S=SECOND-COMPANY
Slot_Rule_2.1:        V=COMPANY2
Slot_Name_3:          S=DATE-OF-ANNOUNCE
Slot_Rule_3.1:        the date when the T=TAKEOVER is announced
Slot_Name_4:          S=DATE-OF-TAKEOVER
Slot_Rule_4.1:        the date when the T=TAKEOVER takes place
Slot_Name_5:          S=ATTRIBUTION
Slot_Rule_5.1:        the person or company that announced T=TAKEOVER
Slot_Rule_5.2:        the person or company that said something about
                      T=TAKEOVER
```

Time_Instructions:	25 min
Time_Form:	15 min

- User 3.5:

Template_Name:	T=TAKE-OVER-BID
Variable_1:	V=COMPANY1 is a company
Variable_2:	V=COMPANY2 is a company
Main_Event_1:	V=COMPANY1 aquired V=COMPANY2
Slot_Name_1:	S=FIRST-COMPANY
Slot_Rule_1.1:	V=COMPANY1
Slot_Name_2:	S=SECOND-COMPANY
Slot_Rule_2.1:	V=COMPANY2
Slot_Name_4:	S=TAKEOVER-COMPANY-VALUE
Slot_Rule_4.1:	the value of V=COMPANY2
Slot_Name_5:	S=DATE-OF-TAKEOVER
Slot_Rule_5.1:	the date when the takeover was announced
Time_Instructions:	10
Time_Form:	15

Category of Users: 4

- User 4.1:

Template_Name:	T=TAKEOVER
Variable_1:	V=COMPANY1 is a company
Variable_2:	V=COMPANY2 is a company
Main_Event_1:	V=COMPANY1 acquired V=COMPANY2
Slot_Name_1:	S=PREVIOUS_OWNER
Slot_Rule_1.1:	the person who owned V=COMPANY2
Slot_Name_2:	S=VALUE-TAKEOVER
Slot_Rule_2.1:	the cost of T=TAKEOVER
Slot_Name_3:	S=COMPANY-TARGET
Slot_Rule_3.1:	V=COMPANY2
Slot_Name_4:	S=COMPANY-PREDATOR
Slot_Rule_4.1:	V=COMPANY1
Time_Instructions:	15 min
Time_Form:	15 min

Bibliography

[A. Smeaton, 1996] R. W. A. Smeaton, "Spanish and Chinese Document Retrieval in TREC-5", in *Proceedings of the Fifth Text REtrieval Conference (TREC-5*, NIST and DARPA, November 1996.

[Aberdeen *et al.*, 1993] J. Aberdeen, J. Burger, S. Roberts, and M. Vilain, "MITRE-Bedford: Description of the Alembic System as Used for MUC-5", in *Fifth Messages Understanding Conference (MUC-5)*, Morgan Kaufmann, August 1993.

[Aberdeen *et al.*, 1995] J. Aberdeen, J. Burger, D. Day, L. Hirschman, P. Robinson, and M. Vilain, "Mitre: Description of the Alembic System Used for MUC-6", in *Proceedings of the Sixth Message Understanding Conference*, Morgan Kaufmann Publishers, November 1995.

[Appelt *et al.*, 1993] D. E. Appelt, J. R. Hobbs, J. Bear, D. Israel, M. Kameyama, and M. Tyson, "SRI: Description of the JV-FASTUS System Used for MUC-5", in *Fifth Messages Understanding Conference (MUC-5)*, Morgan Kaufmann, August 1993.

[Azoff, 1994] E. M. Azoff, *Neural network Time Series Forecasting of Financial Markets*, John Wiley and Sons Ltd., 1994.

[Ballard and Jones, 1990] B. Ballard and M. Jones, "Computational Linguistics", in S. C. Shaphiro, editor, *Encyclopedia of Artificial intelligence*, John Wiley and Sons, 1990.

[Belkin and Croft, 1992] N. J. Belkin and W. B. Croft, "Information Filtering and Information Retrieval: Two Sides of the Same Coin?", *Communications of the ACM*, 35 No.12:29–38, 1992.

[Bokma and Garigliano, 1992] A. F. Bokma and R. Garigliano, "Uncertainty Management through Source Control: A Heuristic Approach", in *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, Mallorca, Spain, July 1992.

[Brill, 1994] E. Brill, "Some advances in rule-based part of speech tagging", in *AAAI-94*, 1994.

[Brown and Bentley, 1995] D. L. Brown and K. Bentley, "To Buy or Not to Buy", *The Magazine of Artificial Intelligence in Finance*, 2 No.2, 1995.

[Callaghan, forthcoming 1997] P. C. Callaghan, *An Evaluation of LOLITA and Related Natural Language Processing Systems*, PhD thesis, University of Durham, Department of Computer Science, Laboratory for Natural Language Engineering, forthcoming, 1997.

[Chinchor and Dungca, 1995] N. Chinchor and G. Dungca, "The Scoring Method for MUC-6", in *Sixth Message Understanding Conference (MUC-6)*, Morgan Kaufmann, November 1995.

[Chinchor, 1992] N. Chinchor, "MUC-4 Evaluation Metrics", in M. Kaufmann, editor, *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, pages 22–29, June 1992.

[Chinchor, 1993] N. Chinchor, "MUC-5 Evaluation Metrics", in *Fifth Messages Understanding Conference (MUC-5)*, Morgan Kaufmann, August 1993.

[Chinchor, 1995] N. Chinchor, "The Statistical Significance of Evaluation Results", in *Sixth Message Understanding Conference (MUC-6)*, Morgan Kaufmann, November 1995.

[Church and Rau, 1995] K. W. Church and L. F. Rau, "Commercial Applications of Natural Language Processing", *Communications of the ACM*, 38 No.11:71–79, 1995.

[Collier, 1994] R. Collier, "N-gram cluster identification during empirical knowledge representation generation", in *Proceedings of the Fifteenth International conference on Computational Linguistics*, pages 1054–1058, 1994.

[Collier, 1996] R. Collier, "Automatic template creation for information extraction, an overview", Technical report, University of Sheffield, 1996.

[Costantino *et al.*, 1996a] M. Costantino, R. J. Collingham, and R. G. Morgan, "Financial Information Extraction at the University of Durham", in *Proceedings of the II Meeting of Artificial Intelligence in Accounting, Finance and Tax*, University of Huelva, Spain, September 1996.

[Costantino *et al.*, 1996b] M. Costantino, R. J. Collingham, and R. G. Morgan, "Information Extraction in the LOLITA System using Templates from Financial News Articles", in *Information Technology Interfaces '96*, June 1996.

[Costantino *et al.*, 1996c] M. Costantino, R. J. Collingham, and R. G. Morgan, "Natural Language Processing in Finance", *The Magazine of Artificial Intelligence in Finance*, 2 No.4, 1996.

[Costantino *et al.*, 1996d] M. Costantino, R. J. Collingham, and R. G. Morgan, "Qualitative Information in Finance: Natural Language Processing and Information Extraction", *NeuroVe$t Journal*, 4 No.6, November 1996.

[Costantino *et al.*, 1996e] M. Costantino, R. G. Morgan, and R. J. Collingham, "Financial Information Extraction using pre-defined and user-definable Templates in the LOLITA System", *CIT - Journal of Computing and Information Technology*, Vol. 4 No. 4, December 1996.

[Costantino *et al.*, 1997] M. Costantino, R. G. Morgan, R. J. Collingham, and R. Garigliano, "Natural Language Processing and Information Extraction: Qualitative Analysis of Financial News Articles", in *Proceedings of the Conference on Computational Intelligence for Financial Engineering (CIFEr '97)*, March 1997.

[Cowie and Lehnert, 1996] J. Cowie and W. Lehnert, "Information Extraction", *Communications of the ACM*, 39 No.1:80–91, 1996.

[Croft and Turtle, 1992] W. B. Croft and H. R. Turtle, "Text Retrieval and Inference", in *Text-Based Intelligent Systems: Current Research and Practice in Information Extraction and Retrieval*, 1992.

[DAR, 1991] DARPA, *Proceedings of the Third Message Understanding Conference (MUC-3)*, Morgan Kaufmann Publishers, May 1991.

[DAR, 1992] DARPA, *Proceedings of the Fourth Message Understanding Conference (MUC-4)*, Morgan Kaufmann Publishers, June 1992.

[DAR, 1993] DARPA, *Proceedings of the Fifth Message Understanding Conference (MUC-5)*, Morgan Kaufmann Publishers, August 1993.

[DAR, 1995] DARPA, *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, Morgan Kaufmann Publishers, November 1995.

[DeJong, 1982] G. F. DeJong, "An overview of the FRUMP system", in *Strategies for Natural Language Processing*, pages 149–176, Elbaum, Hillsdale, N.J., 1982.

[Dekang, 1993] L. Dekang, "University of Manitoba: Description of the NUBA System as Used for MUC-5", in *Fifth Messages Understanding Conference (MUC-5)*, August 1993.

[Dine, 1994] J. Dine, *Company Law*, Macmillan Press Ltd. London, 1994.

[Evans and Lefferts, 1994] D. A. Evans and R. G. Lefferts, "Design and Evaluation of the CLARIT-TREC-2 System", in D. Harman, editor, *The Second Text REtrieval Conference (TREC-2)*, march 1994.

[Fernandez, 1995] M. A. Fernandez, "Spanish Generation in the NLP System 'LOLITA"', Master's thesis, Durham University, Department of Computer Science, 1995.

[Fornasini and Bertotti, 1989] A. Fornasini and A. Bertotti, *Analisi tecnica dei mercati finanziari*, Etas, 1989.

[Garigliano *et al.*, 1993] R. Garigliano, R. G. Morgan, and M. H. Smith, "The LOLITA System as a Contents Scanning Tool", in *Avignon '93*, 1993.

[Garigliano *et al.*, 1995] R. Garigliano, R. Morgan, M. H. Smith, and S. P. Jones, "DEAR: Project summary", in *1st AIKMS Conference*, March 1995.

[Garigliano, 1995] R. Garigliano, "Editorial", *Natural Language Engineering*, 1, March 1995.

[Gilbert, 1995] J. Gilbert, "Artificial Intelligence on Wall Street: An Overview and Critique of Applications in the Finance Industry", material for the COSI 35 course, November 1995.

[Grishman and Sterling, 1989] R. Grishman and J. Sterling, "Preference Semantics for Message Understanding", in M. Kaufmann, editor, *Speech and Natural Language Workshop*, pages 71–74, October 1989.

[Grishman and Sterling, 1993] R. Grishman and J. Sterling, "New York University; Description of the PROTEUS System as used for MUC-5", in *Fifth Message Understanding Conference*, Morgan Kaufmann Publishers, August 1993.

[Grishman, 1995a] R. Grishman, "The NYU System for MUC-6 or Where's the Syntax?", in *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, Morgan Kaufmann Publishers, November 1995.

[Grishman, 1995b] R. Grishman, "Tipster Phase II Architecture Design Document (Tinman Architecture)", 1995.

[Harman, 1994a] D. Harman, "Overview of the second text retrieval conference (TREC-2)", in *ARPA Workshop on Human Language Technology*, pages 328–334, M. Kaufmann, March 1994.

[Harman, 1994b] D. K. Harman, editor, *Proceedings of the Third Text REtrieval Conference (TREC-3)*, NIST and DARPA, November 1994.

[Harman, 1994c] D. K. Harman, editor, *The Second Text REtrieval Conference (TREC-2)*, NIST and DARPA, March 1994.

[Harman, 1995] D. K. Harman, editor, *Proceedings of the Fourth Text REtrieval Conference (TREC-4)*, NIST and DARPA, December 1995.

[Harman, 1996a] D. Harman, "Overview of the Fifth Text REtrieval Conference (TREC-5)", in *Proceedings of the Fifth Text REtrieval Conference (TREC-5)*, NIST and DARPA, November 1996.

[Harman, 1996b] D. K. Harman, editor, *Proceedings of the Fifth Text REtrieval Conference (TREC-5)*, NIST and DARPA, November 1996.

[Hatamian, 1995] T. Hatamian, "Price Forecasting I: Linear Prediction", *The Magazine of Artificial Intelligence in Finance*, 2 No.3:48–54, 1995.

[Hederman, 1992] L. Hederman, *Natural Language Processing and Information Extraction from Texts, Technical Report*, University of Dublin, 1992.

[Hobbs and Israel, 1994] J. Hobbs and D. Israel, "Principles of Template Design", in *ARPA Workshop on Human Language Technology*, pages 172–176, M. Kaufmann, March 1994.

[Hobbs *et al.*, 1992] R. J. Hobbs, D. E. Appelt, J. Bear, M. Tyson, and D. Magerman, "Robust Processing of Real-World Natural-Language Texts", in *Text-Based Intelligent Systems: Current Research and Practice in Information Extraction and Retrieval*, 1992.

[Hobbs, 1993] J. Hobbs, "The Generic Information Extraction System", in *Fifth Messages Understanding Conference (MUC-5)*, Morgan Kaufmann, August 1993.

[Hursch and Hursch, 1988] C. J. Hursch and J. Hursch, *SQL: the Structured Query Language*, Tab Books Inc., U.S.A., 1988.

[Jacobs and Rauf, 1990] P. S. Jacobs and L. F. Rauf, "SCISOR: Extraction Information from On-Line News", *Communications of the ACM*, 33 No.11:88–97, 1990.

[Jacobs *et al.*, 1993] P. S. Jacobs, G. Krupka, L. Rau, M. L. Mauldin, T. Mitamura, T. Kitani, I. Sider, and L. Childs, "GE-CMU: description of the Shogun system used for MUC-5", in *Fifth Messages Understanding Conference (MUC-5)*, Morgan Kaufmann, August 1993.

[Jacobs, 1992] P. S. Jacobs, *Text-Based Intelligent Systems: Current Research and Practice in Information Extraction and Retrieval*, 1992.

[Jones, 1992] K. S. Jones, "Assumptions and Issues in Text-Based Retrieval", in *Text-Based Intelligent Systems: Current Research and Practice in Information Extraction and Retrieval*, 1992.

[Jones, 1994] C. Jones, *Dialogue Structure Models: An Engineering Approach to the Analysis and Generation of Natural English Dialogues*, PhD thesis, 1994.

[Kaufman, 1989] H. Kaufman, *Interest Rates, the Markets, and the New Financial World*, I.B. Tauris Ltd., London, 1989.

[Kingdon, 1994] J. Kingdon, "Criteria for Performance in Gilt Futures Pricing", in A. P. Refenes, editor, *Neural Networks in the Capital Markets*, pages 261–276, John Wiley and Sons, 1994.

[Klimasauskas, 1995] C. Klimasauskas, "Training Neural Nets to Predict Profits", *The Magazine of Artificial Intelligence in Finance*, 2 No.3:21–25, 1995.

[Krupka, 1995] G. R. Krupka, "SRA: Description of the SRA System as Used for MUC-6", in *Proceedings of the Sixth Message Understanding Conference*, Morgan Kaufmann Publishers, November 1995.

[Lee, 1995] R. Lee, "Sterling Software: A NLToolset-based System for MUC-6", in *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, Morgan Kaufmann Publishers, November 1995.

[Levitt, 1994] M. E. Levitt, "Machine Learning for Foreign Exchange Trading", in A. P. Refenes, editor, *Neural Networks in the Capital Markets*, pages 233–243, John Wiley and Sons, 1994.

[Lewis and Jones, 1996] D. D. Lewis and K. S. Jones, "Natural Language Processing for Information Retrieval", *Communications of the ACM*, 39 No.1:92–100, 1996.

[Liddy and Myaeng, 1994] E. D. Liddy and S. Myaeng, "Dr-Link: A system Update for TREC2", in D. Harman, editor, *The Second Text REtrieval Conference (TREC-2)*, March 1994.

[Long and Garigliano, 1994] D. Long and R. Garigliano, *Reasoning by Analogy and Causality, a model and application*, Ellis Horwood, 1994.

[lre, 1992] *Linguistic Research and Engineering European Programme*, 1992.

[Luconi *et al.*, 1994] F. L. Luconi, T. W. Malone, and M. S. S. Morton, "Expert Systems: The Next Challenge for Managers", in R. H. Sprague and H. J. Watson, editors, *Decision Support Systems: Putting Theory into Practice*, pages 365–379, Prentice-Hall, New Jersey, USA, 1994.

[Lytinen and Gershman, 1986] S. L. Lytinen and A. Gershman, "ATRANS: Automatic Processing of Money Transfer Messages", in M. Kaufmann, editor, *9th International Joint Concerence on Artificial Intelligence*, pages 821–825, 1986.

[McCarthy *et al.*, 1992] W. McCarthy, E. Denna, G. Gal, and S. Rockwell, "Expert Systems and AI-based Decision Support in Auditing: Progress and Perspectives", *International Journal of Intelligent Systems in Accounting Finance and Management*, 1 No. 1:53–63, 1992.

[McDonald, 1992] D. D. McDonald, "Robust Partial-Parsing Through Incremental, Multi - Algorithm Processing", in *Text-Based Intelligent Systems: Current Research and Practice in Information Extraction and Retrieval*, 1992.

[Mich and Garigliano, 1994] L. Mich and R. Garigliano, "A Linguistic Approach to the Development of Object Oriented Systems using the Natural Language System LOLITA", in *International Symposium Object-Oriented Methodologies and Systems*, pages 371–386, Sprinter Verlag, Lecture notes in Computer Science 858, September 1994.

[Mich, 1996] L. Mich, "NL-OOPS: From Natural language Requirements to Object Oriented Requirements using the Natural Language Processing System LOLITA", *Journal of Natural Language Engineering*, 2, Part.1, 1996.

[Miller, 1990] G. Miller, "Wordnet: An online lexical database", *International Journal of Lexicography*, 3 No.4, 1990.

[Morgan *et al.*, 1994] R. G. Morgan, M. H. Smith, and S. Short, "Translation by Meaning and Style in LOLITA", in *Proceedings of Machine Translation: Ten years on*, Cranfield University and British Computer Society, November 1994.

[Morgan *et al.*, 1995] R. Morgan, R. Garigliano, P. Callaghan, S. Poria, M. Smith, A. Urbanowicz, R. Collingham, M. Costantino, C. Cooper, and The LOLITA Group, "University of Durham: Description of the LOLITA System as used in MUC-6", in *Sixth Messages Understanding Conference (MUC-6)*, Morgan Kaufmann, November 1995.

[Nelson, 1994] P. E. Nelson, "The ConQuest System", in D. Harman, editor, *The Second Text REtrieval Conference (TREC-2)*, March 1994.

[Nettleton and Garigliano, 1995] D. J. Nettleton and R. Garigliano, "Evolutionary Algorithms and Dialogue", in *Practical Handbook of Genetic Algorithms: New Frontiers*, volume 2, L. Chambers (ed), CRC Press, 1995.

[Nettleton, 1995] D. J. Nettleton, *Evolutionary Algorithms in Artificial Intelligence: A Comparative Study Through Applications*, PhD thesis, 1995.

[Newquist, 1995] H. P. Newquist, "Why is AI Still in Finance?", *The Magazine of Artificial Intelligence in Finance*, 2 No.2, 1995.

[Noble, 1988] H. M. Noble, *Natural Language Engineering*, Blackwell Scientific Publications, 1988.

[Ogden *et al.*, 1994] W. Ogden, J. Cowie, and V. Rauls, "Tabula Rasa Meta-Tool Text Extraction Tool Builder", Technical report, Computing Research Laboratory - New Mexico State University, 1994.

[O'Keefe *et al.*, 1993] R. M. O'Keefe, D. E. O'Leary, D. Rebne, and Q. B. Chung, "The Impact of Expert Systems in Accounting: System Characteristics, Productivity and Work Unit Effects", *International Journal of Intelligent Systems in Accounting Finance and Management*, 2 No. 3:177–189, August 1993.

[Onyshkevych, 1993] B. Onyshkevych, "Template design for information extraction system", in *Fifth Messages Understanding Conference (MUC-5)*, Morgan Kaufmann, August 1993.

[Parker, 1994] B. Parker, "Spell Checking in LOLITA", Master's thesis, University of Durham, 1994.

[Polakoff and Durkin, 1981] M. E. Polakoff and T. A. Durkin, *Financial Institutions and Markets*, Houghton Mifflin Company, 1981.

[Rau, 1987] L. F. Rau, "Knowledge Organization and access in a conceptual information system", *Information Processing and Management*, 23 No.4, 1987.

[Refenes *et al.*, 1994] A. P. Refenes, A. D. Zapranis, and G. Francis, "Modelling Stock Returns in the Framework of APT: A Comparative Study with Regression Models", in A. P. Refenes, editor, *Neural Networks in the Capital Markets*, pages 101–125, John Wiley and Sons, 1994.

[Refenes, 1996] A. P. Refenes, editor, *Neural Networks in the Capital Markets*, John Wiley and Sons, November 1996.

[Rijsbergen, 1979] C. J. V. Rijsbergen, *Information Retrieval 2nd Edition*, Butterworths, 1979.

[Riloff and Lehnert, 1994] E. Riloff and W. Lehnert, "Information Extraction as a Basis for High-Precision Text Classification", *ACM Transactions on Information Systems*, 12 No.3:296–333, 1994.

[Rumbaugh *et al.*, 1991] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, and W. Lorensen, *Object-Oriented Modeling and Design*, Englewood Cliffs : Prentice Hall,, 1991.

[Salton, 1986] G. Salton, "Another look at automatic text-retrieval systems", *Communications of the ACM*, 29 No.7:548–556, 1986.

[Schank and Abelson, 1997] R. C. Schank and R. P. Abelson, *Scripts, Plans, Goals and Understanding*, Erlbaum, Ilillsdale, N.J., 1997.

[Schank and Riesbeck, 1981] R. C. Schank and C. K. Riesbeck, *Inside Computer Understanding*, Lawrence Erlbaum Associated, 1981.

[Sell, 1986] S. Sell, *Expert Systems*, Mac Millan, 1986.

[Sen and Oliver, 1994] T. K. Sen and R. Oliver, "Predicting Corporate Mergers", in A. P. Refenes, editor, *Neural Networks in the Capital Markets*, pages 325–340, John Wiley and Sons, 1994.

[Smith *et al.*, 1994] M. H. Smith, R. Garigliano, and R. G. Morgan, "Generation in the LOLITA system: An Engineering Approach", in *7th International NL Generation Workshop*, June 1994.

[Smith, 1996] M. Smith, *Natural Language Generation in the LOLITA System: An Engineering Approach*, PhD thesis, 1996.

[Stanfill, 1992] C. Stanfill, "Statistical Methods, Artificial Intelligence, and Information Retrieval", in *Text-Based Intelligent Systems: Current Research and Practice in Information Extraction and Retrieval*, 1992.

[Steurer, 1994] E. Steurer, "Nonlinear Modelling of the DEM/USD Exchange Rate", in A. P. Refenes, editor, *Neural Networks in the Capital Markets*, pages 199–211, John Wiley and Sons, 1994.

[Strzalkowski, 1994] T. Strzalkowski, "Document representation in natural language text retrieval", in *ARPA Workshop on Human Language Technology*, pages 328–334, M. Kaufmann, March 1994.

[Sundheim, 1993] B. M. Sundheim, "Tipster / MUC-5 Information Extraction System Evaluation", in *Fifth Messages Understanding Conference (MUC-5)*, August 1993.

[Tait, 1982] J. I. Tait, *Automatic Summarising of English Texts, PhD Thesis*, PhD thesis, 1982.

[The PLUM System Group, 1993] The PLUM System Group, "BBN: Description of the PLUM System as Used for MUC-5", in *Fifth Messages Understanding Conference (MUC-5)*, Morgan Kaufmann, August 1993.

[The PLUM System Group, 1995] The PLUM System Group, "BBN: Description of the PLUM System as used for MUC-6", in *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, Morgan Kaufmann Publishers, November 1995.

[Tomita, 1986] M. Tomita, *Efficient Parsing of NL: A Fast Algorithm for Practical Systems*, Kluwer Academic Publishers, Boston, Ma, 1986.

[Walczak, 1995] S. Walczak, "Selecting Between AI Technologies", *The Magazine of Artificial Intelligence in Finance*, 2 No.2:48–54, 1995.

[Wand and Garigliano, 1995] Y. Wand and R. Garigliano, "Empirical Studies and Intelligent Tutoring Systems", *Instructional Science*, 23, 1995.

[Wang and Garigliano, 1992] Y. Wang and R. Garigliano, "Detection and Correction of Transfer by CAL", in *Second International Conference on Intelligent Tutoring Systems (ITS-92)*, Sprinter-Verlag, June 1992.

[Wang, 1994] Y. Wang, *An Intelligent Computer-based Tutoring Approach for the Management of Negative Transfer*, PhD thesis, 1994.

[Watkins and Eliot, 1993] P. Watkins and L. Eliot, editors, *Expert Systems in Business and Finance: Issues and Applications*, John Wiley and Sons Ltd, England, 1993.

[Wright and Rowe, 1993] G. Wright and G. Rowe, "Expert Systems in the UK Life Insurance Industry: Current Status and Future Trends.", *International Journal of Intelligent Systems in Accounting Finance and Management*, 2 No. 2:113–127, August 1993.

[Zahedi, 1993] F. Zahedi, *Intelligent Systems for Business: Expert Systems with Neural Networks*, Wadsworth Publishing Company Belmont, California, 1993.