Information extraction in the LOLITA system using templates from financial news articles

Marco Costantino, Russell J. Collingham, Richard G. Morgan
Laboratory for Natural Language Engineering
Department of Computer Science
University Of Durham
Science Laboratories, South Road
DURHAM, DH1 3LE, U.K.
Tel +44 191 374 2549, Fax +44 191 374 2560
`marco.costantino@durham.ac.uk`

April 23, 1996

Abstract

This paper addresses the issue of information extraction in the financial domain within the framework of a large Natural Language Processing system: LOLITA. The LOLITA system, Large-scale Object-based Linguistic Interactor Translator and Analyser, is a general purpose natural language processing system. Different kinds of applications have been built around the system's core. One of these is the financial information extraction application, which has been designed in close contact with experts from the financial market in order to overcome the lack of usefulness of many other systems. Three different kinds of financial activities have been identified: company related activities, company restructuring activities and general macroeconomic activities, which form the basis of the "financial activities" approach taken in the development of the LOLITA financial information extraction application. After describing LOLITA as a general purpose base NLP system, the paper addresses the issue of how information extraction is performed within the system.

Keywords: Natural Language Engineering, Information Extraction, Finance.

1 Introduction

Many natural language systems have been built to solve specific and limited tasks. LOLITA has been built with no particular application in mind. However, the flexibility of the system allow the realisation of different kinds of applications around the system's core. One of these is the LOLITA financial information extraction application, which is described in this paper. The application is based on the 'financial activities' approach [CCM95], which identifies three main groups of relevant financial activities in the financial market. The main characteristic of how information extraction is performed within the LOLITA system is that deep natural language understanding is used in order to identify the relevant information in the source articles. The work is organised as follows: in section 2 the generic information extraction task and the main systems developed in the past are described. In section 3 we describe the LOLITA system as a general purpose natural language processing system. Finally, in section 4 we describe the way in which information extraction in LOLITA is performed, focusing on the LOLITA financial information extraction system.

2 Information extraction

The goal of *information extraction* is to extract specific kinds of information from a source document [RL94]. In most systems, the output of the system will consist of *templates*, structures with a predefined number of slots.

Many of the actual systems, for example some of those developed for the MUC-5 [DAR94] and MUC-6 [MGC$^+$95] competition, are based on statistical and probabilistical techniques. However, it is our belief that successful information extraction in broad domains will necessarily require deep natural language techniques rather than shallow pattern-matching or fragment parsing techniques [GWG$^+$95]. Pattern-matching and statistical techniques, in fact, are usually triggered to specific tasks and constraint domains and, therefore, their portability is limited. Moving systems based on these techniques towards other domains usually means major changes to the algorithms and pattern employed. Differently, deep natural language processing is not triggered to any specific domain and, therefore, allows the maximum degree of portability towards other domains. The broad financial domain can be considered an example of this. Pattern-matching and statistical techniques would be able to perform successfully in specific situations, but would be unable to consider all possible cases. Deep natural language processing is instead independent from specific cases and, therefore, represents the best choice for such broad domains. Information extraction within the LOLITA system follows this approach.

2.1 Information extraction in finance

The goal of information extraction applied to finance is, as within other domains, to extract relevant information from a text producing an output usually consisting of a template of the original text.

One of the outstanding properties of the texts that make up financial articles, as opposed to general free-text, is the presence of a considerable high number of metaphors. A randomly selected text from the *Financial Times* of October, the 21th 1992 showed 11 metaphorical expressions in the first sentence, which was 37 words long.

Financial Times 21 Oct 92 London Stock Exchange: Equity futures and options trading

THE *German Bundesbank's* decision *to move to* a variable repo rate, *leading to strong speculation* of further *cuts* in UK base rates, *enlivened a dull derivatives sector* and *sent* Footsie futures *moving sharply ahead*, writes Joel Kibazo.

Very few financial information extraction systems have been realised in the past. One of the few systems is ATRANS [LG86], a system for extracting information from telex messages regarding money transfers between banks. The system was based on the script-frames approach and it has been successful mainly because of the extremely limited domain and the limited information to be extracted. The system that competed in the MUC-5 competition [DAR94] were also able to perform the extraction of information from financial articles. However, these systems were only able to extract information regarding *Joint Ventures* and, thus, work in an extremely restricted subset of the financial domain.

3 The LOLITA system

LOLITA[1] is a *general purpose* natural language processing system and has been under development at the University of Durham for the last eight years [GMS93].

The system has been built with no particular application in mind. This means that different kinds of applications can be easily built around the original system's core. The approach taken for designing and implementing the system follows the lines of natural language engineering rather than those of traditional computational linguistics. The NLE approach emphasises the following aspects of engineering that should be considered when building a NL system [GMS93].

Scale: the size of NLE systems must be sufficient for realistic large-scale application. Properties such as as the vocabulary size, grammar coverage and the number of word senses are critical.

Feasibility: this aspect concerns ensuring that constraints on the running of the system are acceptable. For example. hardware requirements should not be too great and execution speed must be adequate. Feasibility incorporates making the system and its components efficient.

Robustness: robustness is a critical aspect of large-scale systems. Robustness concerns not only the linguistic scope of the system but how it deals with input which falls outside of this scope. At the very least, it should be able to carry on and try its best to cope with the conditions it is working under.

Maintainability: maintainability is a measure of how useful the system is over a long period of time. There are four different aspects that can be referred to the term: *corrective maintenance of software repair*, *enhancement*, *perfective maintenance*, *preventive maintenance.*

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 [Gar95].

The LOLITA system is written in the functional programming language Haskell (currently about 45,000 lines of code, corresponding to about 450,000 lines of code in an imperative language) and based on a large, WordNet-compatible semantic network, *SemNet*, (over 100,000 nodes), similar to a conceptual graph [Sow84]. Its core, being the main part of the system around which individual applications are built, consists of 8 main modules (figure 3).

[1]Large-scale Object-based Linguistic Interactor Translator and Analyser

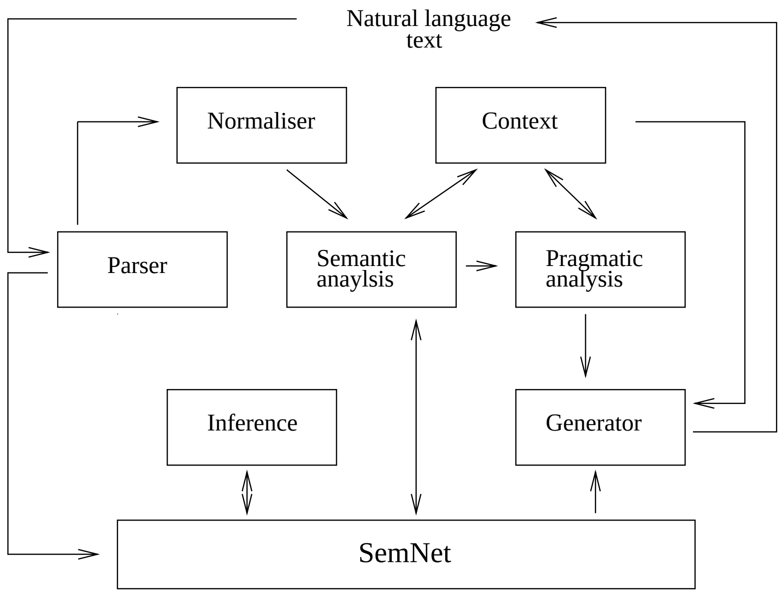

Figure 1: The LOLITA system core.

The semantic network consists of a hierarchy of nodes connected with arcs. The nodes represent entities (*loss*), events (*The company made losses*) and relations (*A company IS A business*). 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. Knowledge is represented in the Semantic Network according to the connectivity between the nodes and arcs. Some of the control variables are:

- **Rank.** This control gives the nodes quantification, i.e. individual, (*the loss Company XY made in the first quarter of '94*), universal (*every loss*), generic (*losses*, or *some losses*), generalization (*shares are a form of investment*), specialization (*one form of investment are shares*) etc.

- **Type.** This control values are very similar to grammatical qualification with few exceptions and additions: entity, relation, typeless, event, fact, greeting etc. The *relation* type mainly represents verbs, *attribute* represents adjectives and *entity* represents nouns [GMS93].

- **Family.** This control groups nodes into the semantic "families", eg. living, animal, human, man-made, abstract, location, organisation, human-organisation etc. [GMS93].

These mechanisms allow the network to contain an elaborate "knowledge base" (i.e. encyclopedic "world" knowledge, linguistic knowledge) which can be expanded via the natural language interface that is part of the system. Before input text is fed to the parser, it is analysed morphologically. The result of the parsing (all sensible parses are stored, ordered according to their acceptability) is normalised before being processed by the semantic analysis subpart, which takes into account pragmatic and context information. The result of the semantic analysis is then used to update and expand the semantic network, i.e. to add information to the knowledge base.

To generate natural language output, the relevant part of the semantic network is fed to the generator component, which is capable of generating natural language output from the internal representation stored in the network [SGM94]. The output from the generator can be varied according to a large set of parameters.

Various kinds of applications have been realised around the LOLITA core including: machine translation from Italian to English, English to Spanish, Language Tutoring [WG92], query application and contents scanning [SGM94].

4 Information extraction in the LOLITA system

The way in which the information extraction application works is quite straightforward. The 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. Finally, the output is produced either using the *generator* or referring to fragments of the source document.

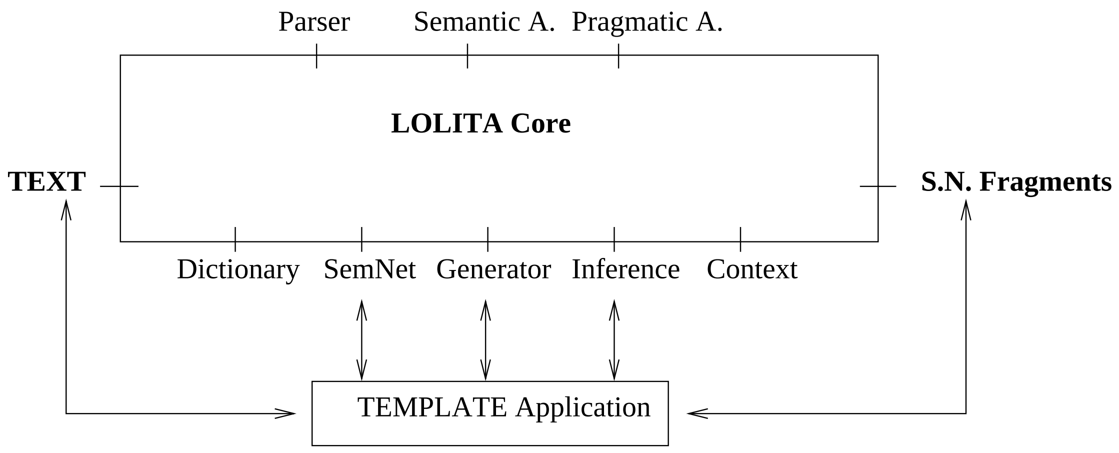

Figure 2: Information extraction using LOLITA.

The template application, thus, interacts with the LOLITA system core retrieving data from the *semantic network* by using the *inference system* and producing output in English either by using the *generator* or by referring to the original text.

The most important characteristic of the way in which information extraction is performed within the LOLITA system is the fact that the templates are built only using deep general purpose natural language techniques, rather than shallow pattern matching techniques. This approach has the advantage of allowing the maximum degree of portability towards new domains, since the template application is built upon the domain independent LOLITA core.

A template is defined in LOLITA as an Haskell data-type comprising a predefined set of slots with associated fill-in rules that direct the search for appropriate information in the semantic network. There are currently five different type of slots available, which differ according to the way in which the slot is filled.

Concept Slot. This slot represents the generic LOLITA templates slot. The rule associated with the slots identifies the relevant concept in the semantic network which is passed to the generator obtaining the corresponding English text.

String Slot. This slot produces the output directly from a given string and not 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, e.g. **Type of Takeover:** *FRIENDLY / HOSTILE.*

Net Slot. In this case, the output is produced by the English generator. However, the slot fill-in rules are retrieved from the semantic network, allowing the creation of user-defined templates or rules by updating portions of semantic network. At present, there are no templates or applications in the system that make use of this feature.

Text Reference Slot. The output is produced using fragments of the original text where possible and the generator if a semantic network's concept doesn't correspond to any fragments of the original text (e.g. when using inference functions). The slot is used when the exact copy of the original text is needed (e.g. in the MUC-6 templates).

Template Reference Slot. This slot is used to create a link to another template. The output of the slot will consist of a pointer to the new template. The *template reference slots* provide the basic mechanism for handling *hyper-templates* in LOLITA which are widely used in the MUC-6 scenario templates [MGC+95].

The slot fill-in rules are used for locating the relevant information for a particular slot in the semantic network. The rules can be built by checking the control variables associated with the nodes or by using the inference functions available in the core system in case the information is not directly stored in the semantic network but has to be inferred. The rules can be used to retrieve any kind of nodes from the semantic network (cf. section 3). However, template slots will usually be filled with *entities* or *events*.

For example, the identification of a company is performed by searching in the semantic network for all the new nodes that belong to the families *organisation* or *human organisation*. The semantic network is in fact updated with all the new nodes created by the semantic analysis of the source documents during the core analysis.

If the information to be located is an *event*, the searching process will involve the identification, through inference functions, of the subjects, objects or target events of the event itself.

Three different kinds of templates are currently available in the LOLITA system, the *Event-based templates*, the *Summary templates* and the *Hyper templates*.

4.1 Event-based Templates

Event-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 to [GMS93].

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

More than one event-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 an article.

Once the template's *parent event* has been identified each slot is filled by searching in the semantic network for the relevant information according to the slots' fill-in rules and starting from the node of the *parent event*.

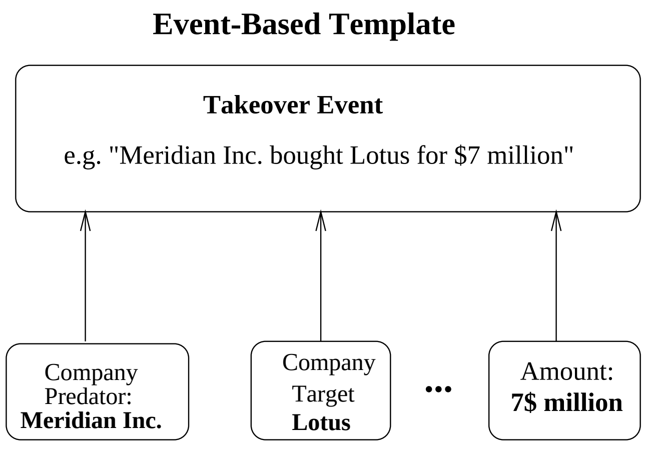
Event-Based Template
Takeover Event
e.g. "Meridian Inc. bought Lotus for $7 million"
Company Predator: Meridian Inc.
Company Target Lotus
...
Amount: 7$ million

Figure 3: Event-based templates.

4.2 Summary-templates

Summary templates represent structures for which it is not possible to identify a *parent 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. For example, a summary template can be composed of the following slots *personal names, organisations, locations, temporal, acronyms, monetary values, descriptions, animates, inanimates.*

Summary templates are built differently from event-based templates. In fact, the slots' fill-in rules are applied to the list of all the nodes generated by the semantic analysis of the source document since a *parent-event* doesn't exist.

4.3 Hyper-Templates

Hyper-templates are structures whose slots can refer to other templates, thus creating a linked *chain* of templates. For example, in the *takeover template* (described later in this section) the slots *company predator* and *company target* could potentially be linked to an *organisation* template which would include detailed information regarding the company. The hyper-template mechanism is potentially usable for linking different kinds of information, not necessarily extracted from the source text, such as company databases or historical share prices. Hyper-templates have been used for implementing the MUC-6 scenario dependent templates (the full description of the implementation of the templates can be found in [MGC$^+$95]).

4.4 Financial information extraction in LOLITA

Four main aspects have to be defined for designing a financial information extraction system: the kind of source articles (from on-line services or from newspapers or magazines), the information to be extracted, the output of the system (summaries or templates) and the interface to the user. Assuming that the output of the system consists of templates, the most important decision is the kind of information to be extracted.

Financial articles represent an extremely wide domain, including different kinds of news: financial, economical, political etc (cf. section 2.1). Therefore, the identification of a unique template able to summarise all the possible financial articles is extremely difficult, if not impossible. The best solution is thus to design more than one template.

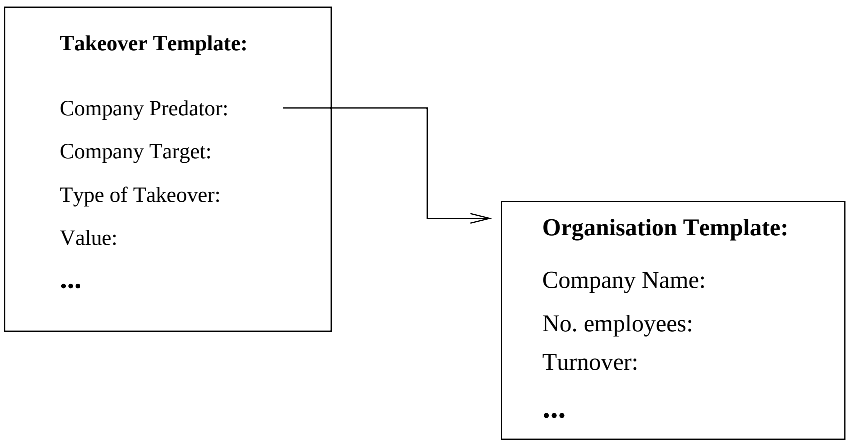

Figure 4: Hyper Templates.

Financial information extraction in the LOLITA system is based on the *financial activities approach* [CCM95]. A *financial activity* is here defined as one potentially able to influence the decisions of the players in the market (brokers, investors, analysts etc.) regarding securities issued by companies. A finite number of relevant *financial activities* are identifiable in the financial market and can be grouped into three different categories.

- **Company related** activities which are those related to the "life" of the company, changes in its status, in the ownership of the company, the number and ownership of its shares etc. This group includes the following financial activities: merger, takeover, flotation, new issue (of shares, bonds etc.), privatisation, market movement, bankruptcy, broker's recommendations, taking a stake, dividend announcement, overseas listing, profits/sales forecasts, profits/sales results, directors' dealings, legal action, investigation.

- **Company restructuring** activities. This activities are related to changes in the productive structure of the company and include: new product, joint venture, staff changes, new factory.

- **General macroeconomics** activities, which include general macroeconomics news that can affect the prices of the shares quoted in the stock exchange and comprises: interest rate movements, currency movements, general macroeconomics data (inflation, unemployment, trade deficit).

A specific template is associated to each of the *financial activities* and comprises a predefined set of slot. In figure 5 the definition of the takeover template in BNF notation is shown.

The source articles of the system consist of large collections of newspapers articles, typically *The Financial Times* CD-ROM collection. However, news from on-line services can be successfully processed by the system.

The information extraction system processes the articles in two stages. First of all, the full semantic analysis of the source articles is performed and the semantic network is updated with the new information. Subsequently, the financial application builds the list of all the *financial activities* included in the source document, which are provided for the user. At this stage the filled templates have not yet been produced. For example, processing an FT article may produce a list as follows:

```
Company Target:           COMPANY_NAME
Company Predator:         COMPANY_NAME
Type of takeover:         {FRIENDLY, HOSTILE}
Value:                    [NUMBER] & CURRENCY_UNIT
Bank adviser predator:    BANK_ADVISER
Bank adviser target:      BANK_ADVISER
Expiry date:              DATE
Attribution:              ATTRIBUTION:
Current Stake predator:   {PERCENTAGE, SHARES, AMOUNT} [NUMBER]
Denial:                   DENIAL

COMPANY_NAME, CURRENCY_UNIT, BANK_ADVISER, ATTRIBUTION, DENIAL: String
DATE: [[0-31][0-12][00-99]]
PERCENTAGE, AMOUNT: Value
```

Figure 5: The Takeover template (BNF notation)

```
3 takeover(s)       found
1 merger(s)         found
2 market movement(s) found
```

At this point, the user can request to see the specific templates which are built by the system.

The implementation of the financial application within the LOLITA system is quite straightforward. The documents are fully processed by the system and the results stored in the semantic network. At this point, the financial application will identify the relevant financial activities and will prompt the user. For example, a *takeover* is identified by the application looking for any *takeover* event in the semantic network. The *takeover* event is defined as follows:

- the event that can be generalised to the concept of *takeover*, *acquisition* or *purchase*. For example "The acquisition of X by Y", where X and Y are companies;
- the event that has a *takeover-action*, e.g. *to buy*, *to purchase*, *to take-over*, etc. and the object is a company. For example: "X has acquired Y for 100 million dollars", where Y is a company.

Similarly, the conditions of all the other financial activities (events) are checked and the full list of the financial activities found is shown to the user. If the user requests the full display of the templates, the slots are filled using the associated rules. For example, the *company predator* of the *takeover* template will be identified as the subject of the takeover event, while the *company target* is the object.

In figure 6 an example of the processing of a source article by the information extraction application and of the user-interface is shown.

All five kinds of slots: concept slots, string slots, net slots, text reference slots and template reference slots (cf. section 4) can be used to define the financial templates. For example, the *company target* slot of the *takeover* template is defined as *concept slot*, while the slot *type of takeover* is defined as *string slot*, since it can only assume values *friendly* or

Source article from the Financial Times:

```
information (line starting with ´.´ to end text):
```

SCOTTS Inc. announced it will acquire Grace-Sierra Horticultural Products with 100 million dollars from a subsidiary of WR Grace, the specialty chemical group, and other investors. Scotts said that after the deal, Grace-Sierra's business and operations would be combined with those of Scotts to form the world's largest turf and horticultural products company, with combined 1993 sales of nearly Dollars 600m. Grace-Sierra manufactures and markets specialty fertilisers for nursery, golf course, greenhouse and consumer markets. WR Grace added that the deal included repayment of Grace-Sierra's indebtedness. The company added that the acquisition was expected to be financed through a combination of long-term sub-ordinated debt and bank borrowings.

.

The system will produce the following output template

```
Template: TAKEOVER
    COMPANY_TARGET: Grace-Sierra Horticultural Products.
    COMPANY_PREDATOR: SCOTTS Inc.
    TYPE_TAKEOVER: FRIENDLY
    VALUE: 100 million dollars.
    ATTRIBUTION: SCOTTS Inc.
```

Figure 6: A example of a LOLITA-produce takeover template

hostile which may not be explicitly stated in the source document and, therefore, must be inferred by the system. The *hyper-template* mechanism can be used in a large number of cases. For example, in the *takeover* template, the slots *company predator*, *company target*, *bank adviser predator* and *bank adviser target* can potentially be linked to other templates or other sources of information, such as company databases. In the same way, the *market movement* template could be easily linked to a historical shares price database.

An important chapter in the development of the financial application is the domain-specific knowledge. In fact, financial articles are based on highly technical and specific knowledge and lexicon. Sentences which would normally have a certain meaning in a normal text might present a totally different one in a financial context. We can think of domain-specific knowledge in the context of a large Natural Language Processing System such as LOLITA as *semantic and pragmatic rules* which are used by the system to correctly *understand* the choose the meaning of a particular sentence in the financial context. As far as the takeover template is concerned, various domain-specific rules have been identified:

- if X takes full control of Y, this implies a takeover;
- X buys a majority stake in Y, this implies a takeover;
- X buys a 51 (or over) per cent stake in Y, this implies a takeover;
- X pays M for Y and Y is a company, this implies the takeover of Y by X.

At present, the financial application is only partially implemented within the LOLITA system. Basically, only the first group of financial activities (company related activities) has been partially coded in the system.

One of the disadvantages of the way in which information extraction is currently performed within the LOLITA system is the fact that only deep natural language techniques are used. Therefore, the performance in terms of speed can be, in particular situations, penalised in comparison to systems based on pure pattern-matching or fragment-parsing techniques.

5 Conclusions

In this paper we have tried to give an overview of how financial information extraction is performed in the LOLITA system. Unlike many other information extraction systems, the emphasis is on full natural language processing of the text. Additional information regarding the LOLITA financial information extraction system and, more specifically, its relevance as an effective financial tool can be found in [CCM95].

In a paper of this size it is impossible to provide details of every aspect of a large NL system such as LOLITA. Because of the commercial value of the LOLITA project, the system is not publicly available. However, we are keen to give demonstrations of the system and serious enquiries should be addressed to the authors.

References

[CCM95] M. Costantino, R.J. Collingham, and R.G. Morgan. Natural language processing in finance. *The Magazine of Artificial Intelligence in Finance*, 2 No.4, 1995.

[DAR94] DARPA. *Proceedings of the Fifth Message Understanding Conference.* Morgan Kaufmann Publishers, August 1994.

[Gar95] R. Garigliano. Editorial. *Natural Language Engineering*, 1, March 1995.

[GMS93] R. Garigliano, R.G. Morgan, and M.H. Smith. The lolita system as a contents scanning tool. In *Avignon '93*, 1993.

[GWG+95] R. Gaizauskas, T. Wakao, R. Garigliano, R.G. Morgan, and R.J. Collingham. Muc-6: The competition, the controversies and the challenges. In *SALT Workshop, University of Sunderland*, 1995.

[LG86] S.L. Lytinen and A. Gershman. Atrans: Automatic processing of money transfer messages. In Morgan Kaufmann, editor, *9th International Joint Concerence on Artificial Intelligence*, pages 821–825, 1986.

[MGC+95] 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)*, October 1995.

[RL94] 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.

[SGM94] 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.

[Sow84] J.F. Sowa. *Conceptual Structures, information processing in mind and machine.* Addison-Wesley, 1984.

[WG92] 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.