Focusing on Scenario Recognition in Information Extraction

Milena Yankova
Linguistic Modelling Department,
Central Laboratory for Parallel Processing,
Bulgarian Academy of Sciences,
25A Acad. G. Bonchev Str.,
1113 Sofia, Bulgaria
myankova@lml.bas.bg

Svetla Boytcheva
Department of Information Technologies,
Faculty of Mathematics & Informatics
Sofia University "St. Kl. Ohridski",
5 James Baurchier Str.,
1164 Sofia, Bulgaria
svetla@fmi.uni-sofia.bg

Abstract

This paper reports a research effort in Information Extraction, especially in template pattern matching. Our approach uses reach domain knowledge in the football (soccer) area and logical form representation for necessary inferences of facts and templates filling. Our system FRET[1] (Football Reports Extraction Templates) is compatible to the language-engineering environment GATE and handles its internal representations and some intermediate analysis results.

1 Introduction

An enormous amount of information exists in natural language texts only but to analyse and process this information automatically, it has to be first distilled into a more structured form. Information Extraction (IE) systems extract pieces of information by mapping natural language texts into predefined structured representation - linguistic patterns, usually sets of attribute-value pairs. Some of the attribute-value pairs are to be filled in by results from morphological analysis, named-entities recognition, and (partial) syntactic analysis. These processes are relatively well studied and most of the IE systems report high precision and recall. However, the semantic analysis - which includes building logical forms, recognition of references and template filling - is a complicated process, which is still far from its ultimate solution.

This paper focuses on the semantic processing in IE. Following the terminology established by the Message Understanding Conferences (MUCs), we shall call the specification of the particular events or relations to be extracted SCENARIO and we shall refer to the final, tabular output format of information extraction as TEMPLATE. The actual structure of the templates used has varied from a flat record structure at MUC-4 [9] to a more complex object oriented definition which was used for Tipster and MUC-5 [2], MUC-6 [7] and MUC-7 [3]. Once filled, templates represent an extract of key information from the text [12]. Extracted information can be stored in databases for various purposes such as text indexing, information highlighting, data mining, natural language summarisation, etc.

Different systems provide different approaches for solving semantic problems in IE. The CRYSTAL system [11], for example, is based on machine-learning covering algorithm for building expected rules for template filling. Large hand-marked training corpus is needed. But the domain is quite static - weather forecast - with explicitly fully expressed information. The system creates a formal representation of the text that is equivalent to related database entries.

Another Information Extraction system is SMES [10], which does not have semantic analysis implemented in it. Fragments extracted by a lexically driven parser are attached to anchors - lexical entries (mainly verbs). If successful, the set of found fragments together with the anchor build up an instantiated template. Filling templates strongly

[1] This work is partially supported by the European Commission via contract ICA1-2000-70016 "BIS-21 Centre of Excellence"

depends on the words and relations between them, as they appear in the text.

In our approach we use the IE system GATE 2.1 beta 1 (GATE - General Architecture for Text Engineering) [4], which provides lexical analysis, named entity recognition, coreference resolution and other NLP modules. The system has been used for many language-processing projects; in particular for Information Extraction in several languages.

In this paper we present work in progress, aiming at the implementation of the system FRET (Football Reports Extraction Templates). FRET provides syntactic analysis and template (scenario) pattern matching from English text. The innovative aspect in our considerations is the relative weight of the semantic analysis, since we use logical forms, a lexical knowledge base and certain inference to match text and templates.

The paper is organised as follows: section 2 presents a short overview of IE as a whole and some difficulties with performing subtasks in the chosen domain. Section 3 describes the structure of the data resource bank integrated in FRET. Section 4 discuses our approach in translation to logical form. Section 5 describes in details the templates' structure. Section 6 explains the algorithm for filling templates with information from the text. Section 7 contains the conclusion.

2 Information extraction

IE can be divided into the following subtasks [6]:

Lexical Analysis, which turns a text into a sequence of sentences, each of them is a sequence of lexical items (tokens). Usually sentences are not marked, so special techniques are required to recognise sentence boundaries. Each token is looked up in the dictionary to determine its possible features and part-of-speech types.

Named Entity (NE) Recognition, which takes a sequence of lexical items and tries to identify reliably determinable structures using a set of regular expressions: proper names, locations, organizations, dates, currency amounts and etc. The max score result reported in MUC-3 [8] trough MUC-7 in this task is f-measure < 97%.

Coreference Resolution, which identifies different descriptions of the same entity in different parts of a text (usually one-two neighbour sentences). These descriptions are the ones identified by NE recognition and their anaphoric references. The best result reported for this task at MUC 3-7 is f-measure < 67,5%.

Syntactic Analysis, which provides some aspects of syntactic analysis and simplifies the phase of fact extraction. The arguments to be extracted often correspond to noun phrases in the text, and relationships to grammatical functional relations. Note that for IE we are only interested in the grammatical relations relevant to the template; correctly determining the other relations may be a waste of time [6].

Template (Scenario) Pattern Matching, which maps the syntactic structures to semantic structures related to the templates to be filled in. This stage extracts the events or relationships relevant to the scenario. The max score result reported for this task is f-measure < 57%.

One of the most important questions is how to recognise the scenario, which we are looking for. For this purpose one specifies a template, as a sequence of slots some of which are marked as obligatory and the others are optional. When the required (marked) slots are filled in then we say that the scenario is matched and the slots in the template represent the wanted information from the processed text. If the information in the processed text is not enough to fill in the necessary slots, the text does not correspond to the scenario.

The domain chosen for tuning and testing FRET is football. The corpus is composed from BBC reports about 31 matches of the Euro2000 championship. These texts have a specific text structure and FRET's parser is tailored to cover it. Match reports and comments have paragraph structure and provide rich temporal information.

Most often, the preferred research domains in IE are fully informative with explicit statically expressed facts, where every statement is true at least in the current text. Such domains are news articles, telegraphic military messages, weather forecast etc., which are used in MUC competitions. On the other hand football reports are dynamic with no assurance, that when once facts are declared they will not be negated later. The needed information sometimes is not fully provided and inferences are required for extracting the implicitly expressed facts. Tuning in a domain that allows frequent changes even in terminology is also an important and actual difficulty. Details about further problems in this domain are given below.

2.1. Named entity recognition

First problem is NE recognition for proper names, especially for foreign names. The players from different nationalities have specific names that can be out of the database for recognising NEs. This is due to the limited list of predefined names. It is impossible to collect all names for all nationalities and distinct ways for transcribing foreign names. Another difficulty are nicknames of the players, which are used in the text. Sometimes players' team numbers are used instead of person names.

```
Example 1:
Ronaldo – soccer superstar; the
Phenomenon
Example 2:
Number nine scores.
```

2.2. Coreference resolution

NE recognition problems described above contribute to the coreference resolution problem. Instances of player's designation by metaphoric description of their performance are more or less unrecognizable.

```
Example 3:
The brazilian superstar rediscov-
ered his enchanting mix of regal
majesty and youthful wonder.
```

For finding metaphors it is necessary to have explicit semantic description for each word (based on meaning postulates, conceptual graphs etc.) to recognize usage of words in a way different from the traditional one. This is a huge time consuming task because of the large amount of words existing in the corpus texts. Correct metaphors recognition is quite a hard task even for most of humans.

FRET uses the results of GATE, which performs the first three IE tasks: *Lexical analysis*, *NE recognition* (f-measure < 96%), and *Coreference resolution* (f-measure<51.9%). Therefore FRET's performance in solving these tasks in football domain depends only on GATE's performance and the built-in GATE data corpus.

3 Data resources in FRET

The process of filling slots in a template doesn't imply certain "full understanding", but only recognizes semantically equivalent representations of the expected concepts. For most of the concepts in the text we need only naive semantic information. However to fulfil the template slots, a more detailed lexical knowledge base is needed, including the necessary information for all concepts and possible relations between them that can be referred in some sense to the template. An expert in our specific domain – football reports, develops this lexical knowledge base in FRET.

FRET's resources, shown in Figure 1 include three types of data:

- Static Resource Bank,
- Dynamic Resource Bank,
- Template's description (see section 5).

Static resource bank contains linguistic knowledge (lexicon, grammar) as well as a knowledge base that represents some main "action relations":

- *effect causality*: an action A causes effects $B_1, B_2, ..., B_n$. There are two types of effects – intentional effects and side effects;
- *preconditions-causality*: an action A may have preconditions $B_1, B_2, ..., B_n$;
- *enablement* - action A enables action B;
- *decomposition* - action A is performed when subactions $B_1, B_2, ..., B_n$ are performed;
- *generation* - action A generates action B.

The knowledge base also includes lists of synonym concepts in the football domain. For example:

```
Example 4:
Synonym objects: [net, home...]
Synonym actions:
  [head, shoot, stab, hit...]
```

One of the more natural ways to attach required semantic information to already syntactically parsed sentences is to translate them into first–order Logical Form (LF). For this purpose we need grammar rules and rules for translation into LF. These rules are kept in Static Resource Bank.

Since in the football reports most of the sentences have quite complex syntax structure, in order to simplify template matching we substitute some of the concepts and relations between them with their normal form (infinitives, base forms etc.). So we use a lexicon including about 65 000 words' base forms and their wordforms. For shortness we do not describe the lexicon into details here, because the focus is on the semantic analysis and resources closely related to it.

Texts in the football domain usually do not include all the information necessary for filling templates. That's why each text is associated with another data resource that contains additional

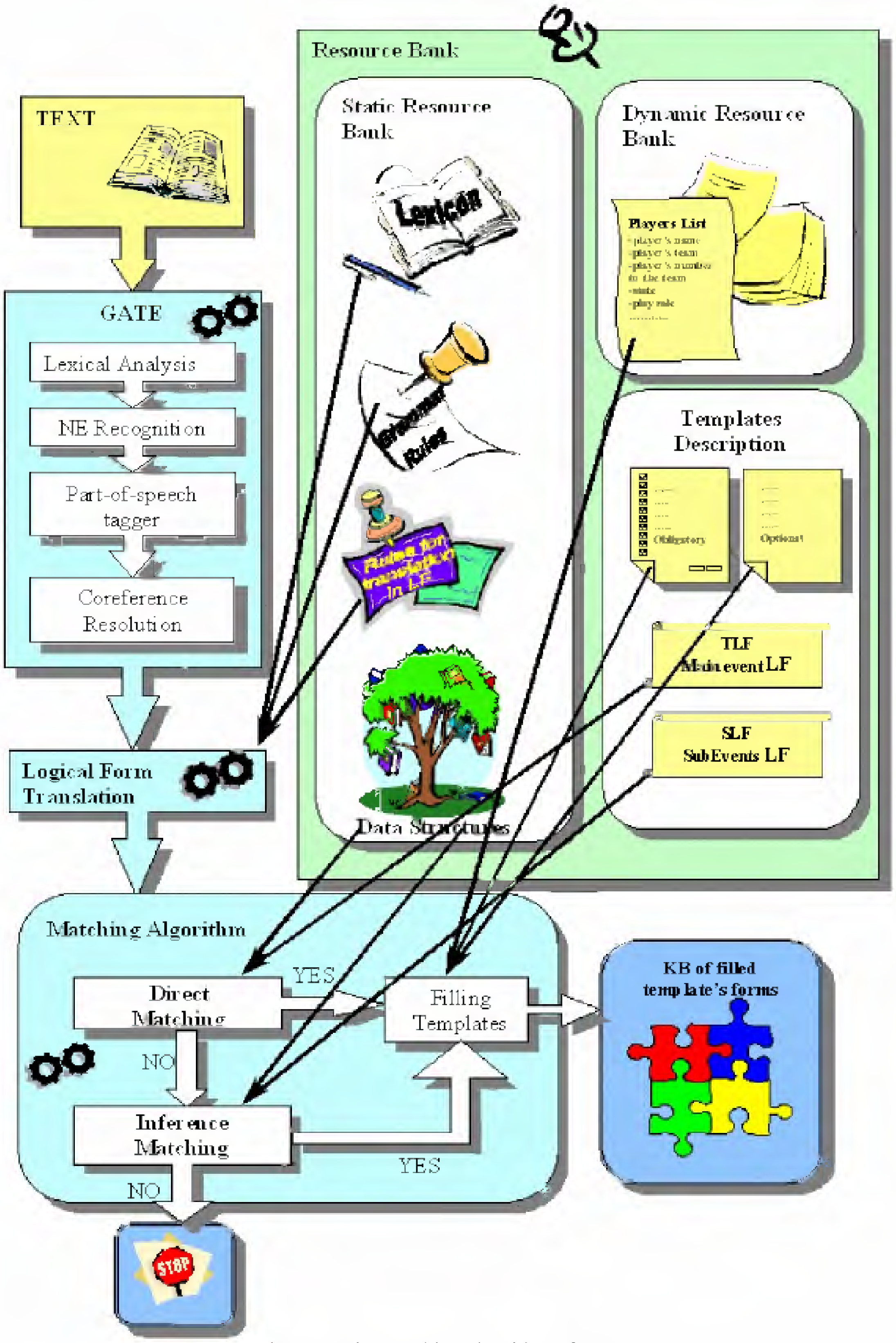


Figure 1: The matching algorithm of FRET

information. For example, team names, lists of players in each of the teams, playing roles, penalties etc. This is fast changing information and cannot be stationary added in the system, but it is reported in the processed texts and is automatically extracted. For example the players in both teams are usually presented in the beginning of the match with their names, numbers and position in the team. All such additional information is stored in the Dynamic Resource Bank (Fig. 1).

4 Logical form translation

A specially developed left-recursive, top-down, depth-first parser, implemented in Sicstus Prolog, is used in FRET for logical form translation. This parser uses grammar rules and rules for translation into LF from our resource bank. In LF we represent all words as predicates with predicate symbol the corresponding base form of the word and one argument. For example the word "squeezes" will be represented in LF as `squeeze(X)`. For thematic roles we also add predicates with predicate symbol "`theta`" and three arguments. The second argument is a constant and represents the thematic role. The rest of the arguments are bound with the corresponding predicates that represent related concepts or constants to this thematic role (see examples). All proper names are represented as constants that occur as arguments of the corresponding thematic roles.

```
Example 5:
Sentence:
53 mins: Beckham shoots the ball
across the penalty area to Alan
Shearer who heads into the back of
the net at the far post and scores.
Logical form: score(_A) &
theta(_A,agnt,'Alan Shearer') &
head(_C) & theta(_C,agnt,'Alan
Shearer') & theta(_C,obj,_D) &
ball(_D) & theta(_C,into,_E) &
net(_E) & shoot(_F) &
theta(_F,agnt,'Beckham') &
theta(_F,obj,_D) &
theta(_F,to,'Alan Shearer') & &
theta(_F,across,_G) & area(_G) &
theta(_G,char,_H) & penalty(_H)
& time(53).
```

Coreference solving provided by GATE in this stage [5] helps for earlier binding of the variables in LF and makes further matching processes easier (especially future inferences).

Usually partial information about an event may be spread over several sentences. This information needs to be combined before a template can be generated. In other cases, some of the information is only implicit, and needs to be made explicit through an inference process. That's why FRET associates the time of the event to each produced LF. Every LF is decomposed to its disjuncts and each of them is marked with the associated time.

Some problems come out while parsing. One of them is the interpretation of negations. As described in [1] and taking into account the specific domain texts, we distinguish explicit and implicit negations.

In explicit usage, "NO" negates sentences immediately preceding the current one.

```
Example 6:
Sentence:
69 min: Jaap Stam will be next.
Surely he has to score. NOOOOOO!
He's blazed it way, way, over.
Logical Form:
not (be(_A) &
theta(_A,agnt,'Jaap Stam') &
theta(_A,char,_B) & next(_B) &
score(_C)& theta(_C,agnt,'Jaap
Stam')) & time(69).
```

In this case the negation is marked in the LF of all previous sentences in the current paragraph, which are bound trough their variables in the discourse.

In implicit usage of negation inside one sentence (marked with words as "but", "however"...), negation is inserted as in LF follows:

- in case of "but" and "however", only *preceding* words in this sentence are negated;
- in case of "however", used in the beginning of the sentence, the preceded sentences restricted by the discourse are negated.

```
Example 7:
Sentence:
87 min:
Barker again came close to score
but his strike failed to hit the
target.
Logical Form:
not(score(_A)& be(_B)&
theta(_B,agnt,'Barker')&
theta(_B,to,_A)& theta(_B,char,_E)&
close(_E)&theta(_A,agnt,'Barker'))&
```

```
strike(_C)& theta(_C,poss,'Barker')
& fail(_D)& theta(_D,agnt,_C)&
theta(_D,to,_G)& hit(_G)& theta(_G,
agnt,_C)& theta(_G,obj,_H)&
target(_H)& time(87).
```

In both cases we are paying attention to not having double usage of negations. Note that we interpret the negation in a rather domain-specific way, which is motivated by our detailed study of the available domain corpus.

5 Template format

Template is described by a table with two types of fields that have to be filled in:

- obligatory fields;
- optional fields (see example in Table 1).

If the obligatory fields are filled in, the template succeeds and the scenario is found and matched to the text. Optional fields can be left empty if there is no information for their filling in the processed text. Both types of fields, taken as a whole, contain the key information presented in the text.

Obligatory		Optional	
●	Player	○	Assistance
●	Time	○	Position
●	Team	○	Type of action (by head, by shoot, ...)
●	Score	○	Player's penalties (red, yellow cards, minutes and etc.)

Table 1 Template table for the scenario Goal

The template scenario also includes information about two types of events:

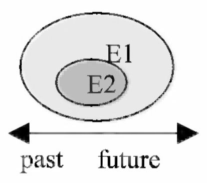
a) *main event* – LF of obligatory and optional fields.

```
Example 8:
LF of the main event Goal:
Obligatory: Score(_A) &
theta(_A,agnt,Player) &
time(Minute)
Optional: Action1(_C) &
theta(_C,agnt,Player) &
theta(_C,obj,_D) & ball(_D) &
theta(_C,Loc,_E) & Location(_E) &
Action2(_F) &
theta(_F,agnt,Assistant) &
theta(_F,obj,_D) &
theta(_F,to,Player)
```

b) *set of subevents* – LFs of events related to the main event and type of relations to the main event.

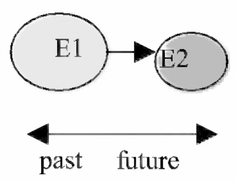
The matching algorithm of FRET is based on relations between events and we present here more details about three special types of implications, used in the next examples.

- Event E2 *is a part of* event E1 (Fig.2)
- Event E1 *enables* event E2, i.e. event E1 happens before the beginning of event E2 and event E1 is a precondition for E2 (Fig. 3)
- Event E1 *entails* event E2, i.e. when E1 happens E2 always happens at the same time (Fig. 4)



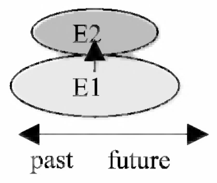
```
Example 9:
E1: Player's shot
hits the net.
```
E2: The ball is into the net.

Figure 2



```
Example 10:
E1: The player
shoots the ball
E2: Player's
shot hits the
net.
```

Figure 3



```
Example 11:
E1: Player's shot
hits the net.
E2: The player
scores.
```

Figure 4

Note that in example 8 the predicate names are capitalized because they are variables. This means that practically the matching procedure is performed in second order logic, further employing the set of synonyms as possible predicate names.

6 Filling template

The matching algorithm of FRET (Fig. 1) has two main steps:

- matching LFs;
- filling templates.

Matching LFs step is based on the unification algorithm.

Direct matching:

Initially the matching algorithm tries to match LFs produced from the text to the LF of the main event.

We call this step *direct matching*. Each situation in the text is described by a set of LFs marked by the same moment of time. Direct matching algorithm searches for necessary information consecutively in each set of individual LFs. In this step we also use synonyms lists and data structures representing action relations from the knowledge base. Direct matching algorithm succeeds when all main event's LFs variables related to template's obligatory fields are bound.

```
Example 12:
Sentence:
12 min:Pessotto steps up.He scores!
Logical form: score(_A) &
theta(_A,agnt,'Pessotto')&time(12).
```

In example 12 we can fill in only the obligatory template fields of "goal" (example 8), because we have no additional information about any kind of assistance, position and etc.

In contrast in Example 5, the direct matching algorithm succeeds and all obligatory and optional template fields will be replete (see Table 2).

Inference matching:

If the direct matching algorithm fails then FRET starts the inference-matching algorithm. Inference-matching algorithm tries to match some of template's subevents LFs with the text LFs similarly to the direct matching algorithm. If we find the necessary information about some subevent, we use the corresponding additional information about the type of relation between this subevent and the main event. Using inference rules and the knowledge base, FRET inference-matching algorithm derives an inference from subevents LFs. If it is possible successfully to match the inferred LFs to the main event LF, then the inference-matching algorithm succeeds.

```
Example 13:
SubEvent: Player shoots the ball
into the net.
SubEvent's logical form:
Action(_A) & theta(_A,agnt,Player)
& theta(_A,obj,_C) & ball(_C) &
theta(_A,into,_D) & Net(_D).
7Sentence:
41 min: From the resulting corner,
Micoud finds Sylvain Wiltord on the
edge of the area. He shoots the
ball into the net.
Logical form: time(41) & shoot(_A)
& theta(_A,agnt,'Sylvain Wiltord')
& theta(_A,obj,_C) & ball(_C) &
theta(_A,into,_D) & net(_D) &
find(_E) & theta(_E,agnt,'Micoud')
& theta(_E,obj, 'Sylvain Wiltord')
& theta(_E,loc,_F) & edge(_F) &
theta(_G,poss,_F) & area(_G) &
theta(_E,from,_H) & corner(_H) &
theta(_H,char,_I) & resulting(_I).
```

This subevent is matched to the "goal" scenario applying inference as shown in example 11: *"He shoots the ball into the net"* implies that *"there is a score"*. Our current evaluation with available domain texts shows that simple relations between events similar to those in examples 9, 10 and 11, are sufficient for covering paraphrases and successful matching of subevents.

Filling template form:

When the matching algorithm succeeds, then we can fill in the template. First we fill the required information in the obligatory fields. If necessary we use some additional information from Dynamic resource bank. At the next step we try to fill those of the optional fields for which there is sufficient information. Table 2 presents the result obtained after filling in a template from Example 5.

Obligatory		Optional	
●	Player: *Alan Shearer*	○	Assistance*: David Beckham*
●	Time: *53 min*	○	Position: *penalty area*
●	Team: *England*	○	Type of action: *heads*
●	Score: *4*	○	Player's penalties *cards(1,yellow ,12 min)*

Table 2

Texts from a total of 31 reports are tested. The scenario templates are filled in with precision: 80%, recall: 50% and f-measure: 44,44%. We have to mention that these measures are approximate, because we report work in progress and FRET is tested only for a few templates (goals – totally 89, sent off – totally 8).

7 Conclusion

In the world of high technologies, extracting information from "free" NL texts is very important. Therefore we try to find an easy and effective way for filling in templates, which may allow for real semantic processing of large text collections.

In this paper we describe on-going work on semantic analysis in IE: our main idea and core technique for realization. We think that the inference is an integral part of finding facts in texts, and that for making inferences it is necessary to represent sentences into LFs. However, not all the information provided in the text is needed for simple template filling; so we choose shallow parsing and partial semantic analysis. Note that when the simpler inference fails the more complicated one is started. The knowledge database has the major role in inferences from the logical forms. Because of the fast changing domain terminology a regular tuning of the database is required with the help of domain experts. Even human beings are embarrassed to recognize domain specific usage of some words, which are treated as terms in this domain.

The main innovative aspects of FRET are:

- usage of the specific temporal features in the domain texts. Scenarios are matched to paragraphs discussing certain important moments. This simplifies the choice of sentences to be parsed in order to fill in a template;
- clear and sound logical definitions of notions like "template filling", allowing application of higher-order logic;
- elaborated inference mechanisms which provide relatively deep NL understanding but only in "certain points". The de-facto fragmentation of the knowledge base into *scenario-relevant* and *scenario-irrelevant* facts allows relatively simple and very effective inference. Note that only scenario-relevant relations between events are linked in the inference chains;
- attempts for domain-specific treatment of the negation.

However, many difficulties in the implementation are due to our decision to present sentences into pure logical forms. One of them, that we plan to work on, is a more precise resolution of negations' scope. We hope to improve FRET performance in the next months when an extensive evaluation with further unknown texts is planned.

The implementation of presented version of FRET is in Prolog to make it clear and comprehensible. Another advantage of the logical programming language is easier realization of inferences and knowledge representation. The next challenge is to rewrite the system in Java, which is not a trivial task. The reason of following this direction is a better co-operation with GATE system and faster performance in case of growing, real-scale linguistic and knowledge resources.

The presented approach can easily be adapted to a new domain, because it uses just a few domain dependent resources: data structures and template description. However we should keep in mind that this approach is tailored only for text with a specific temporal structure. In our further work we plan to test FRET system behaviour on another domains of such type and we expect similar results.

References

[1] Boytcheva, Sv., A. Strupchanska and G.Angelova. (July 2002), "*Processing Negation in NL Interfaces to Knowledge Bases*" In Proceedings of. ICCS-2002, pp.137-150

[2] Chinchor. N. (1993), "*The statistical significance of the MUC-5 results*", In Proceedings of MUC-5, pp. 79-83. Morgan Kaufmann,.

[3] Chinchor N., (1998), "*Overview of MUC-7*", In Proceedings of MUC-7, http://www.muc.saic.com/

[4] Cunningaham, H., D. Mayard, K. Boncheva, V. Tablan, C. Ursu and M. Dimitrov (2002) "*The GATE User Guide*". http://gate.ac.uk/.

[5] Dimitrov, Marin (2002), "*A light-weight Approach to Coreference Resolution for Named Entities in Text*", MSc thesis, Sofia University

[6] Grishman, Ralph (1997), "*Information Extraction: Techniques and Challenges*", International Summer School, SCIE-97

[7] Grishman, R. and B. Sundheim. (1996), "*Message Understanding Conference – 6 : A Brief History*". In Proceedings of COLING-96, pp. 466--471.

[8] Lehnert, W., C. Cardie, D. Fisher, E. Riloff, and R. Williams, (May 1991), Ò*University of Massachusetts: MUC-3 Test Results and Analysis*, in Proceedings of MUC-3, Morgan Kaufmann, pp. 116-119.

[9] Lehnert, W., D. Fisher, J. McCarthy, E. Riloff, and S. Soderland, ÒUniversity of Massachusetts: MUC-4 Test Results and Analysis, in *Proceedings of MUC-4* (June 1992), Morgan Kaufmann,. pp. 151-158.

[10] Neumann, G., R. Backofen, J. Baur, M. Becker, C, Broun (1997) "*An Information Extraction Core System for Real World German Text Processing*"

[11] Soderland, Stephen (1997) "*Learning to Extract Text-based Information from the World Wide Web*"

[12] Wilks, Yorick (1997) "*Information Extraction as a Core Language Technology*", International Summer School, SCIE-97.