# Extracting Information for Automatic Indexing of Multimedia Material

**Horacio Saggion**
**Hamish Cunningham**
**Diana Maynard**
**Kalina Bontcheva**
**Oana Hamza**
**Christian Ursu**
**Yorick Wilks**

Department of Computer Science
University of Shef£eld
Regent Court
211 Portobello Street
S1 4DP - Shef£eld - England
UK
{saggion,hamish,diana,kalina,oana,cursu,yorick}@dcs.shef.ac.uk

**Abstract**

This paper discusses our work on information extraction (IE) from multi-lingual, multi-media, multi-genre Language Resources, in a domain where there are many different event types. This work is being carried out in the context of MUMIS, an EU-funded project that aims at the development of basic technology for the creation of a composite index from multiple and multi-lingual sources. Our approach to IE relies on a £nite state machinery provided by GATE, a General Architecture for Text Engineering, pipelined with full syntactic analysis and discourse interpretation implemented in Prolog.

## 1. Introduction

The University of Shef£eld is adapting information extraction (Cunningham, 1999; Gaizauskas and Wilks, 1998; Appelt, 1999) technology to produce formal annotations about essential events in football multimedia programme material for the Multimedia Indexing and Searching Environment (MUMIS)[1] project, which aims at the development of basic technology for the creation of a composite index from multiple sources in different languages.

We are working on information extraction from English sources while other project participants are dealing with sources in Dutch (Centre for Telematics and Information Technology, Netherlands) and German (Deutsches Forschungszantrum für Künstliche Intelligenz, Germany). Information extraction is carried out on textual sources, but in future stages of development of this project, the information will also be extracted from transcribed spoken commentaries from radio and television broadcasts. These transcriptions are being produced by the University of Nijmegen (Netherlands). The three IE systems target a shared domain and multilingual lexicon of the football domain developed by ESTEAM (Sweden). As the information is extracted from multiple sources describing the same events in various ways, a merging component is in charge of solving con¤icts and merging information (University of Nijmegen). A menu-based user interface (in Dutch, English, and German) is being developed by the Max-Planck-Institut für Psycholinguistik (Germany), that will allow professional users to query a database of annotations and play video fragments matching the query (e.g., “all goals scored by Owen”). For an overview of the project the reader is referred to (Declerck et al., 2001).

The textual sources used for this project are taken from reports of the Euro2000 Championships: ticker reports that give a minute by minute objective account of the match; match reports that also give a full account of the match but may be subjective; and comments that give general information such as player pro£les. English reports are drawn from a variety of online media sources (BBC-online, Press Association, The Guardian, etc.). These sources report the same events in different ways: as an illustration a source may say “Substitute Westerveld comes on for van der Sar” while another may say “van der Sar (Westerveld 65)” to refer to a substitution event. An example of an English ticker can be seen in Figure 1. The elements to be extracted that are associated with the events are: players, teams, times, scores, and locations on the pitch. An example of the task is presented in Figure 2: the text describes a foul event committed by “Beckham” at the 41 minute of the £rst half. The system extracts the information and produces XML output. The extraction of temporal information is essential to our task because it is the key for locating interesting fragments in the video material.

In this paper, we present our approach to Information Extraction from English text that is based on the use of £nite state machinery pipelined with full semantic analysis and discourse interpretation. The rest of the paper is organised as follows: in the next section we give an overview of the £nite state components of our system and we present its evaluation. In section 3 we discuss our approach to parsing.

[1] `http://mumis.vda.nl/`, funded by the EC’s 5th Framework HLT programme under grant number IST-1999-10651.

England 1-0 Germany
(Charleroi - Att: 30,000)
3 mins: Germany break down the right in the £rst attack of the match but Phil Neville covers well and returns the ball to David Seaman, who clears.
...
41 mins: Beckham is shown a yellow card for retaliating on Ulf Kirsten seconds after he is denied a free-kick.
...
Full-time: England 1-0 Germany

Figure 1: Ticker from the Euro 2000 Championships: Match between England and Germany

41 mins: Beckham is shown a yellow card for retaliating on Ulf Kirsten seconds after he is denied a free-kick.

```
<event_entry>
  <event_type>yellow card</event_type>
  <event_ID>16</event_ID>
  <event_time>41</event_time>
  <original_doc_name>/Z:/mumis/src/mumis/resources/text/sources/
England-Germany/bbc-england-germany-ticker.txt</original_doc_name>
  <player_1>Beckham</player_1>
  <team_player_1>England</team_player_1>
  <player_2></player_2>
  <team_player_2></team_player_2>
  <score>0:0</score>
 </event_entry>
<event_entry>
  <event_type>foul</event_type>
  <event_ID>17</event_ID>
  <event_time>41</event_time>
  <original_doc_name>/Z:/mumis/src/mumis/resources/text/sources/
England-Germany/bbc-england-germany-ticker.txt</original_doc_name>
  <player_1>Beckham</player_1>
  <team_player_1>England</team_player_1>
  <player_2>Kirsten</player_2>
  <team_player_2>Germany</team_player_2>
   <score>0:0</score>
  </event_entry>
```

Figure 2: Example of the IE Task and Output of the System.

Then, in section 4 we present our work on discourse interpretation and £nally, in section 5 we close with conclusions and future work.

## 2. Overview of the English Information Extraction System

An analysis of the domain has revealed that there are 31 types of events in a football match (kick-off, substitution, goal, foul, red car, yellow card, etc.). Some events, such as “yellow card”, require only one player while others, like “substitution”, require two players to be extracted. All events require temporal and location information to be extracted from the text.
We have produced semi-formal dictionary de£nitions for each event in the domain (e.g. “Corner: An offensive player kicks the ball from stationary at a corner”) but these de£nitions are not enough to produce an operational system. A careful analysis of textual sources was done to de£ne each event formally in order to allow its identi£cation and extraction from texts by automatic means. We have used WordSmith (Scott, 1996) as the main tool for corpus analysis.

Our system is conceptualised as a *Java front-end system* based on £nite state transduction followed by a *Prolog back-end system* for inference over a classi£cation hierarchy implemented in SICStus Prolog. The system architecture is shown in Figure 3.

The £nite state machinery is based on ANNIE, a free IE system available as part of GATE, a General Architecture for Text Engineering (Cunningham et al., 2002) (see `http://gate.ac.uk/`).

The input to the process is a document (plain text, HTML, SGML, XML, RTF, or EMAIL) that is automati-

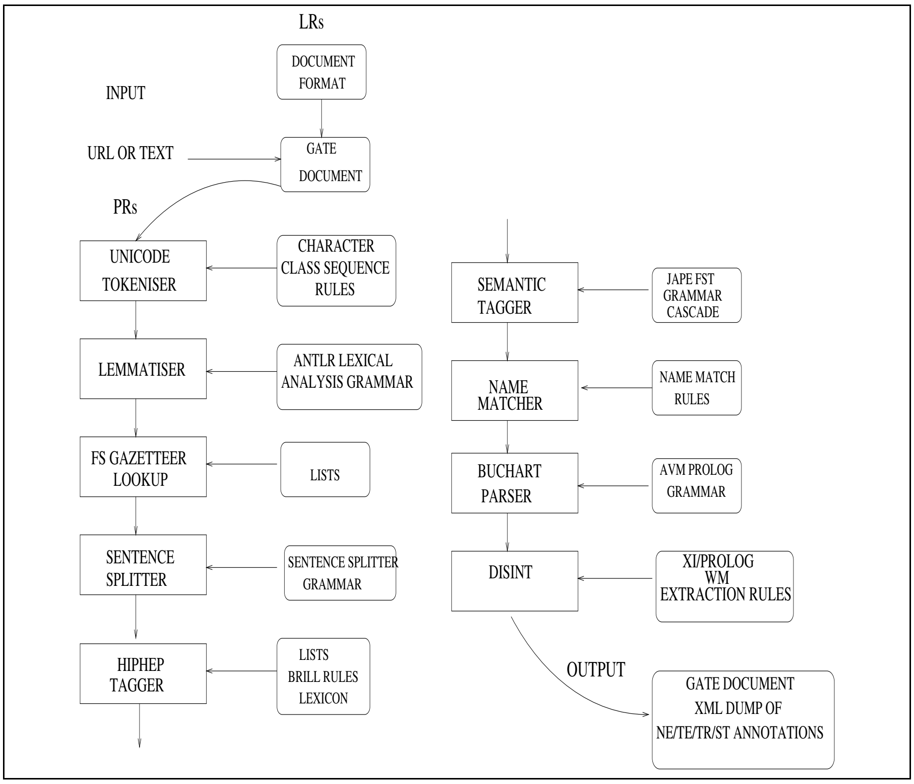


Figure 3: English Information Extraction System Architecture

cally transformed by a text structure analyser into a GATE document: a structure containing the "text" of the original input and a number of annotation sets. Each component in the pipeline adds new information to the document in the form of annotations. An annotation has a type, a pair of offsets, and a set of feature-values that allow encoding orthographic, lexical, syntactic, and semantic information.

The £nite state components we have developed are:

- Unicode Tokeniser: it is a £nite state transducer based on regular expressions over Unicode Character Classes that splits the text into very simple tokens such as numbers, punctuation and words of different types;
- Gazetteer Lookup process: it identi£es and classi£es key words related to particular entity types in a particular domain. For MUMIS, we have collected information in order to identify players, trainers, referees, time markers, as well as stadiums, and sites. Features such as the af£liation and the position of each player in the Championships are particularly important for discourse interpretation (e.g., the team af£liation and position helps during entity coreference).
- Semantic Tagging: it identi£es and classi£es more complex sequences of tokens in the source document. We use JAPE (Java Annotation Pattern Engine) (Cunningham et al., 2002), a pattern-matching engine implemented in Java and based on the Common Pattern Speci£cation Language (Appelt, 1996), to identify and annotate regular expressions over annotations. JAPE grammars are sets of rules which act on annotations assigned in earlier phases, in order to produce annotated entities. The rules are separated into two parts: the left hand side (LHS) of the rule performs pattern-matching; the right hand side (RHS) of the rule describes the annotation to be assigned. On the LHS, the pattern is described in terms of the annotations already assigned while the RHS of the rule contains information about the annotation to be produced, including attributes and their corresponding values. Distinctive characteristics of JAPE grammars are the possibility of specifying context for the pattern, control strategies and priority for triggering of the rules, and Java code that is executed whenever the LHS of the rule matches the annotations allowing for a deeper level of analysis. JAPE uses a compiler that translates grammar rules into Java objects that target the GATE API (and a regular expression library). JAPE grammars were also used to develop the rule-based sentence splitter used in the system.
- Orthographic Name Matcher: it looks for association between named entities by verifying a set of rules (e.g., "D. Beckham" and "David Beckham"). The information produced by this module is used during discourse interpretation.

The £nite state step is essential to identify as early as possible typical expressions and domain jargon that can be semantically interpreted without relying on full parsing or semantic interpretation. For example, an expression like "van der Sar (Westerveld 65)" indicates a substitution, and can be identi£ed with gazetteer look up and a regular

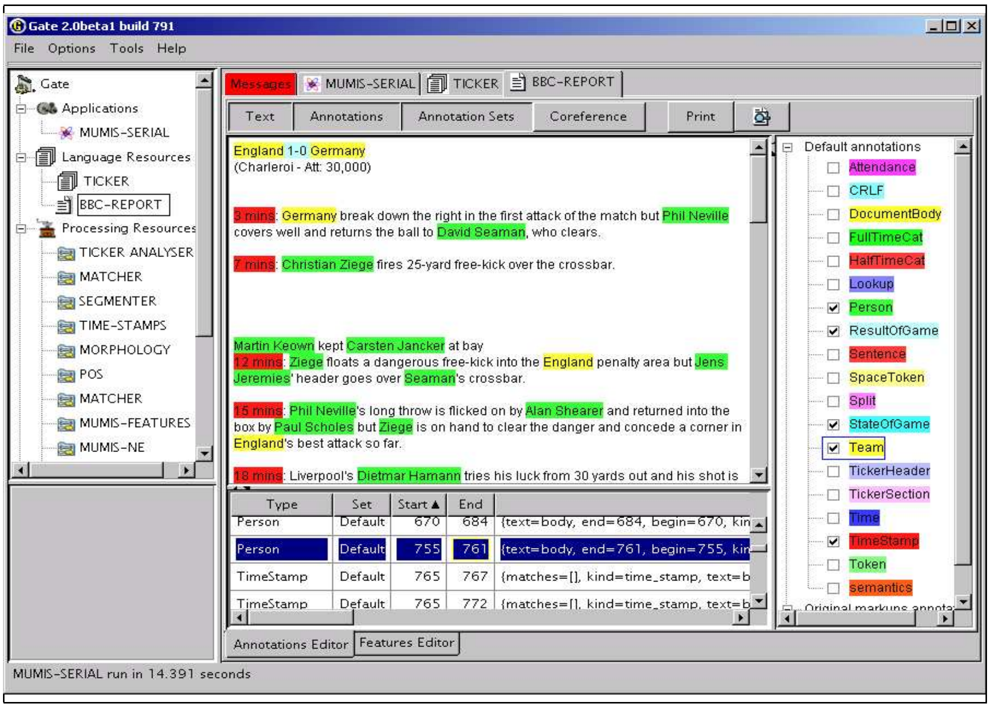


Figure 4: MUMIS IE System with the GATE User Interface

grammar.

The JAPE rule bellow allows to identify such construction:

```
Rule: Substitute2
(
 {Person.kind == player}
 (SPACE)
 {Token.string == "("}
 {Person.kind == player}
 (SPACE)
 {Token.kind == number}
 {Token.string == ")"}
 (SPACE)?
):sub

-->

:sub.Substitution =
{rule = Substitute2}
```

The LHS is a regular pattern over annotations "Token" (produced by the tokeniser) and "Person" (produced by other JAPE rules). The rule also refers to SPACE that is a macro we have used in order to specify space tokens. The RHS speci£es a "Substitution" annotation type to be added into the annotation set. Further processing using Java code in JAPE is done by other grammar rules to extract semantic features from the annotation produced by this rule (the player that goes in "Westerveld", the player that goes out "van der Sar", and the time of the event "65").

The GATE GUI is shown in Figure 4. It shows the document under analysis and the annotations produced by the system.

### 2.1. Evaluation of the Semantic Tagger

We evaluated the performance of our semantic tagger in the named entity recognition task using precision and recall (Firmin and Chrzanowski, 1999). We used GATE's automated precision and recall evaluation tool AnnotationDiff (Cunningham et al., 2002). As our training corpus refers to matches where 'England', 'Germany' or 'Holland' participated, we are relying on unseen texts reporting other events to evaluate our named entity recognition component. We have manually annotated with category 'Person', the set of all BBC reports about matches of the Group B of the Euro2000 championships: these constitute our gold standard for evaluation. In group B none of the above mentioned countries participated. Our semantic tagging component for the 'Person' (e.g., players) category was measured at 91% precision, 76% recall, and a combined F-measure of 82% (when precision and recall are of equall weight). This compares favourably with results obtained using the 'default' GATE named entity recognition system that also uses gazetteer and grammar rules. The baseline system obtained 91% precision, 30% recall, and 45% F-measure.

## 3. Parsing

Ticker reports are in essence dynamic texts: a verbal account of events over time stamps, and this fact is taken into account during analysis: scores change as well as the players leaving or entering the game. These texts also have a speci£c text structure that we take into account when parsing. We have developed a simple pre-processing step that identi£es 'ticker header', where information about lists of players and the result of the game is usually stated, and 'ticker sections' grouping together sentences describing events under single time stamps. These sections are different from traditional paragraphs because they can either span multiple paragraphs or even a paragraph can contain multiple ticker sections. Other textual sources (e.g., match reports and comments) do not provide such rich temporal information, nevertheless they contain complete event descriptions ("David Beckham - a muted force in attack - was shown a yellow card for a late challenge on Kirsten"). March reports and comments have paragraph structure that is identi£ed by the GATE document structure analyser.

Part of speech tagging is done with an implementation of the "independence and commitment" approach to POS tagging (Hepple, 2000), based on (Brill, 1995): the novelty of this approach is the fact that no rule interaction is considered, and so the speed of the learning process is improved while the tagging accuracy is maintained. For MUMIS, we have enriched the default lexicon produced by the learning step with our own vocabulary. The lexicon of the domain was obtained from the corpus and appropriate part of speech tags were produced with the help of the CELEX database (Burnage, 1990).

We have also implemented a rule-based lemmatiser that produces for each noun and verb in the input text an af£x and root. The root is the entry of the word in the dictionary and the af£x is a normalised form that represents the word ending ("ed" for all past participle forms and "s" for plurals). The external lemmatiser program is implemented as a set of regular expressions speci£ed in ¤ex and translated into C code. These patterns represent both morphological analyses of the input and exception rules. The exception rules were derived from WordNet (Miller, 1995), but also revised by the analysis of a number of English corpora. In this sense, the lemmatisation process is domain independent.

We are using an implementation of the Bottom-up chart parsing described in (Gazdar and Mellish, 1989), enriched with semantic rules that construct a naive semantic of each sentence in £rst order logical form. The parser is complete in the sense that every analysis licensed by the grammar is produced, though there is a mechanism to control this. On completion a "best parse" algorithm is run to select a single analysis of the sentence, which may be partial if no tree spanning the whole sentence can be constructed.

The parser uses two grammars: the £rst is a domain dependent semantic grammar used to produce logical forms for the entities of the football domain; the second is a context-free phrasal grammar of English enriched with features and values that has been used in many Information Extraction projects (Cunningham et al., 1999). It consists of a sequence of sub-grammars for: noun phrases (NP), verb phrases (VP), prepositional phrases (PP), relative phrases(R) and sentences (S).

The semantic rules produce unary predicates for entities and events (e.g., *player(e1)*, *save(e2)*) and binary predicates for properties (e.g. *lsubj(e1,e2)*). Constants (e.g., $e1$, $e2$) are used to represent entity and event identi£ers. First order predicate names are: (i) the citation forms obtained during lemmatisation; (ii) forms used to code syntactic information (e.g. $lsubj$ for the logical subject of a given verb); or (iii) speci£c predicate names being used for domain modelling (e.g., *player_of*). Instantiated values of properties attached to events are used as the slot £llers in the template representation the system is producing (e.g., the name of the scorer in a goal event).

As an illustration, the semantic representation for the sentence "41 mins: Beckham is shown a yellow card for retaliating on Ulf Kirsten seconds after he is denied a free-kick" is shown in Figure 5. Note that while some information in the semantic representation is explicit in the input sentence (e.g., the verb "retaliating" used to produce the predicate $retaliate$) other information is domain dependent and semantic in nature (e.g., the predicate $player$). This latter information is produced by the semantic grammar.

The parser is written in SICStus Prolog (version 3.6.8). In order to call the parser and to obtain the output back we rely on Jasper, a bi-directional interface between Java and SICStus (SIC, 2000).

## 4. Discourse Interpretation

The discourse interpreter is based on a World Model representing the ontological (or hierarchical) knowledge about a particular domain. The interpreter works by mapping the information produced by the parsing and semantic interpretation into an evolving Discourse Model of the input text. The World Model contains rules allowing the deduction of new knowledge from the "explicit" information found on the text, and also the connection between new and old instances mentioned in the input text (co reference).

Based on the corpus study we have speci£ed a world model of the football domain. We have adopted the XI Knowledge Representation Language (Gaizauskas and Humphreys, 1996b), a formalism that allows the user to code and operate with symbolic knowledge. XI is compiled into Prolog, making it possible to mix procedural knowledge with the basic declarative formalism's constructs. We have chosen XI because it has been proved successful in other domains for information extraction (Humphreys et al., 1998; Humphreys et al., 2000).

### 4.1. Knowledge Coding

XI provides basic language construct to specify hierarchical relations. Individuals, classes of individuals, inclusion relations between classes of individuals and multiple inheritance hierarchies can be de£ned and attribute-values may be associated with classes or with individuals.

*time_stamp(e416)*, *time_count(e416,'41')*, *player(e419)*,
*name(e419,'Beckham')*, *player_team_name(e419,'England')*,
*mid£elder(e419)*, *be(e418)*, *time(e418,present)*, *aspect(e418,simple)*,
*voice(e418,active)*, *adj(e418,e421)*, *adj(e421,shown)*, *lsubj(e418,e419)*,
*card(e422)*, *number(e422,sing)*, *adj(e422,yellow)*, *det(e422,a)*,
*retaliate(e423)*, *time(e423,none)*, *aspect(e423,simple)*, *voice(e423,active)*,
*player(e425)*, *name(e425,'Ulf Kirsten')*, *player_team_name(e425,'Germany')*,
*forward(e425)*, ...

Figure 5: Partial Semantic Output

In XI, classes are represented as unary predicates and individuals as atoms. An attribute or property is a binary predicate, the £rst argument of which is the class/individual the attribute is assigned to and the second being the value of this attribute. The value can be a £xed term or a term that became instantiated in appropriate situations when new knowledge is deduced during processing. Figure 6 gives an overview of our ontology of the football domain.

Clauses $A \implies B$ are used to specify 'inclusion' relations (all $B$ is an $A$). Clauses $I \longleftarrow C$ are used to specify 'is a' relations between individuals and classes ($I$ is a $C$). Operators $\vee$ and & indicate disjunction and conjunction respectively. In addition to the declarative operators, a number of constructs can be used during deduction: $A \Rightarrow B$ is used to verify class inclusion (every $B$ is a $A$), $A \leftarrow B$ is used to verify if $A$ 'is a' $B$, and $hasprop(I, P)$ is used to verify if $I$ 'has property' $P$ (also, $nodeprop(I, P)$ can be used to verify properties but only at the instance level). Properties can be attached to individuals or classes conditionally on the actual state of the world. Conditional properties are speci£ed with the "if" operator ($:-$).

Properties of individuals and classes are speci£ed through the $props$ predicate in the form:

$$props(Class|Individual, [Properties])$$

where $Properties$ can be unconditional or conditional on the current state of the world.

We have created our world model by de£ning the three types of properties required by the discourse interpreter:

(i) rules that prevent entity coreference (e.g., a player who is playing cannot corefer with a player who is no longer playing); these are speci£ed using the reserved predicate $distinct$;

(ii) rules that allow the modi£cation of the discourse model with presuppositions events and entities have (e.g., a substitution event presupposes a player that is playing and a player that is on the bench); these are compiled using the reserved predicate name $presupposition$,

(iii) rules allowing the modi£cation of the discourse model with consequences for events and entities (e.g., a goal event has as consequence a change in the state of the game); these are de£ned using the reserved predicate $consequence$. These rules also allow for completing the partial parse that the parsing process might be produced.

More than 300 rules for discourse interpretation have been manually created for the MUMIS project.

Knowledge processing is done in the following way:

1. The naive semantics of each sentence produced during parsing is mapped into an evolving discourse model of the text. Each element in the semantics is mapped into a node in the hierarchy by consulting a conceptual dictionary developed from the multilingual-lexicon developed for MUMIS. Ambiguous predicates (e.g. goal for the event "to score a goal" and goal for the object in the pitch "the two goal-posts and the crossbar") are dealt in with discourse interpretation rules that take into account the context of the predicate. Predicates not found in the dictionary are mapped into objects or events depending on their syntactic category (noun or verb). Attributes are associated to instances using $props$ constructs.

2. Presuppositions are added to the model. This causes the creation of new hypothesised instances.

3. Object coreference is applied to solve instances produced by the parser or by presupposition rules. The coreference mechanism will try to solve a hypothesis using the current sentence, any unresolved hypothesis in this step will be removed from the model.

4. Consequences are added to the model, any hypotheses will remain active because they might be solved later as new information is processed.

5. Event coreference is applied mainly to merge partially instantiated events.

We model the dynamic nature of tickers by maintaining the "dynamic state of the game", a logical structure that records among other elements: the teams playing, the time stamp under consideration, the players participating in the game, the £nal score, and the score at particular time stamps. This structure is the basis for many of the deductions the system has to make: for example when the system

*entity(X)* $\Longrightarrow$ *object(X)* $\vee$ *event(X)* $\vee$ *attribute(X)*
*object(X)* $\Longrightarrow$ *time(X)* $\vee$ *person(X)* $\vee$ *soccer_artifact(X)*
*person(X)* $\Longrightarrow$ *player(X)* $\vee$ *of£cial(X)* $\vee$ *player_collective(X)*
*player(X)* $\Longrightarrow$ *goalkeeper(X)* $\vee$ *mid£elder(X)* $\vee$ *...*
*soccer_artifact(X)* $\Longrightarrow$ *ball(X)* $\vee$ *goalmouth(X)* $\vee$ *goalpost(X)* $\vee$ *crossbar(X)*
*event(X)* $\Longrightarrow$ *substitution_event(X)* $\vee$ *goal_event(X)* $\vee$ *save_event(X)* $\vee$ *red_card_event(X)* $\vee$ *yellow_card_event(X)* $\vee$ *shot_on_goal_event(X)* $\vee$ *...*
*attribute(X)* $\Longrightarrow$ *functional(X)* $\vee$ *relational(X)*
*player_of* $\longleftarrow$ *functional(X)*
*trainer_of* $\longleftarrow$ *functional(X)*

Figure 6: Partial Ontology for the Football Domain

needs to £nd the time of a particular event or when it needs to decide which team scored the goal. The dynamic state of the game is updated as the text is processed sentence by sentence. Note that in tickers, expressions like “England 2 - 1 Germany” clearly indicate the state of the game, nevertheless, out of context they do not provide enough information about the team that scored the goal, actually the previous state of the game could be “England 1 - 1 Germany” or “England 2 - 0 Germany”. The “dynamic state of the game” provides the information or context to disambiguate. For match reports and comments discourse interpretation rules hipothesise temporal information that is solved by the coreference resolution algorithm.

Event and entity coreference is the basic mechanism that allows the system to £ll in properties for each event in the domain. Coreference is an uni£cation-based process that is based on: (i) the notion of distance between nodes in the ontological hierarchy; (ii) a measure of similarity between entities and events computed using the values of associated properties (for example two player can be made coreferent if they were found similar by the name matcher process); and (iii) rules that constrain coreference. We use the basic coreference algorithm described in (Gaizauskas and Humphreys, 1996a). It is a general algorithm that requires careful coding of the semantic properties of events and entities of the domain. We approach coreference resolution in a symbolic way by specifying a number of rules that constrain coreference. Coreference rules are mainly negative in the sense that they specify when entities should not corefer. A typical case is when two players participate in the same event, the two players should be different. Another typical case would be an event whose semantics indicate two players of the same team, so the coreference mechanism should avoid coreference between players that participate in that event if they are from different teams. The rule below states that *PLAYER1* and *PLAYER2* are different if they participate on the same *GOAL* event.

$$\begin{array}{l}
props(player(\mathit{PLAYER1}),[\\
(distinct(\mathit{PLAYER1},\mathit{PLAYER2}):-\\
\mathit{GOAL}\leftarrow goal_event(_),\\
hasprop(\mathit{GOAL},player_1(\mathit{GOAL},\mathit{PLAYER1})),\\
hasprop(\mathit{GOAL},player_2(\mathit{GOAL},\mathit{PLAYER2})))\\
]).
\end{array}$$

Presuppositions allow hypothesis of entities essential to complete the semantics of a particular event. A typical case is when the parser is unable to £nd the logical subject or logical object of a domain verb, in which case they need to be presupposed and passed to the discourse interpreter to complete the meaning of the domain verb. Presuppositions are hypotheses local to the sentence under analysis. The rule below says that if the concept “substitution” is found, then presuppose a “substitution” event and hypothesise two players where the second player is introduced by preposition ‘for.’

$$\begin{array}{l}
props(substitution(\mathit{SUBS}),[\\
(presupposition(\mathit{SUBS},\\
[substitution_event(E),player(\mathit{PLAYER1}),\\
player(\mathit{PLAYER2}),team(\mathit{TEAM}),\\
player_of(\mathit{PLAYER1},\mathit{TEAM}),\\
player_of(\mathit{PLAYER2},\mathit{TEAM}),\\
player_1(E,\mathit{PLAYER1}),\\
player_2(E,\mathit{PLAYER2})]):-\\
hasprop(\mathit{PLAYER2},for(\mathit{FOR},\mathit{PLAYER2})))\\
])
\end{array}$$

We also use presuppositions to dealt with ambiguous items.

Consequences also allow hypothesis of entities and events but in a more general way, because they can be instantiated with entities coming from remote parts of the document or not yet processed. The rule below indicates that $PLAYER$ plays for a given $TEAM$ with name $NAME$ (where $NAME$ is knowledge coming from the gazetteer look-up step on the £nite state machinery).

$$\begin{array}{l}
props(player(\mathit{PLAYER}),[\\
(consequence(\mathit{PLAYER},\\
[team(\mathit{TEAM}),player_of(\mathit{PLAYER},\mathit{TEAM}),\\
name(\mathit{TEAM},\mathit{NAME})]):-\\
hasprop(\mathit{PLAYER},team_name(\mathit{PLAYER},\mathit{NAME})))\\
])
\end{array}$$

Event coreference is carried out to merge partially instantiated events. Consider for example the text “Goal! England 1 - 0 Germany”: Both expressions “Goal!” and “England 1 - 0 Germany”, are used to indicate a goal event, if two goal events are deduced from the text, the event coreference mechanism should merge them using constraints such as the time stamp of the events and the participating players and teams. This is carried out with speci£c Prolog code.

A template extraction algorithm was implemented that explores the discourse model, extracting and sequencing in temporal order instances of the 31 event types. For each event instance, values from properties attached to them in the discourse model are extracted and used to instantiate event templates. A template writing algorithm is used to read the list of templates and produce XML output.

## 5. Conclusions and Future Work

In this paper, we have presented our approach to information extraction from football reports for the MUMIS project. We have been successful in the integration of two complementary paradigms: £nite state machinery implemented in Java with full syntactic analysis and discourse interpretation implemented in Prolog. The system is complete and operative and we are in the process of producing complete XML annotations for the full corpus.

A stimulating aspect of this project was the domain modelling of a complex event such as a football match where 31 events, their participants and relationships need to be accurately identi£ed in plain English texts. This is a challenging situation for information extraction.

Our work in progress involves formal evaluation of the information extraction task that is being carried out using gold standards produced by professional annotators. Future work will address information extraction from transcribed spoken commentaries.

## Acknowledgements

We would like to thank Rob Gaizauskas and George Demetriou for their help with the discourse interpreter. We also thank Wim Peters for his help with the CELEX database. This work is funded by the EC's 5th Framework HLT programme under grant number IST-1999-10651.

## 6. References

D.E. Appelt. 1996. The Common Pattern Speci£cation Language. Technical report, SRI International, Arti£cial Intelligence Center.

D. Appelt. 1999. An Introduction to Information Extraction. *Arti£cial Intelligence Communications*, 12(3):161–172.

Eric Brill. 1995. Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging. *Computational Linguistics*.

G. Burnage. 1990. *CELEX: A Guide for users*. Centre for Lexical Information, Nijmegen.

H. Cunningham, R.G. Gaizauskas, K. Humphreys, and Y. Wilks. 1999. Experience with a Language Engineering Architecture: Three Years of GATE. In *Proceedings of the AISB'99 Workshop on Reference Architectures and Data Standards for NLP*, Edinburgh, April. The Society for the Study of Arti£cial Intelligence and Simulation of Behaviour.

H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan, and C. Ursu. 2002. *The GATE User Guide*. http://gate.ac.uk/.

H. Cunningham. 1999. Information Extraction: a User Guide (revised version). Research Memorandum CS–99–07, Department of Computer Science, University of Shef£eld, May.

T. Declerck, P. Wittenburg, and H. Cunningham. 2001. The Automatic Generation of Formal Annotations in a Multimedia Indexing and Searching Environment. In *Workshop on Human Language Technology and Knowledge Management*, Toulouse, France. http://www.elsnet.org/acl2001-hlt+km.html.

T. Firmin and M.J. Chrzanowski. 1999. An Evaluation of Automatic Text Summarization Systems. In I. Mani and M.T. Maybury, editors, *Advances in Automatic Text Summarization*, pages 325–336. The MIT Press.

R. Gaizauskas and K. Humphreys. 1996a. Quantitive Evaluation of Coreference Algorithms in an Information Extraction System. In *DAARC96 - Discourse Anaphora and Anaphor Resolution Colloquium*. Lancaster University.

R. Gaizauskas and K. Humphreys. 1996b. XI: A Simple Prolog-based Language for Cross-Classi£cation and Inhetotance. In *Proceedings of the 7th International Conference in Arti£cial Intelligence: Methodology, Systems, Applications*, pages 86–95, Sozopol, Bulgaria.

R. Gaizauskas and Y. Wilks. 1998. Information Extraction: Beyond Document Retrieval. *Journal of Documentation*, 54(1):70–105.

G. Gazdar and C. Mellish. 1989. *Natural Language Processing in Prolog: An Introduction to Computational Linguistics*. Addison-Wesley Publishing Company.

Mark Hepple. 2000. Independence and commitment: Assumptions for rapid training and execution of rule-based POS taggers. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL-2000)*, Hong Kong, October.

K. Humphreys, R. Gaizauskas, S. Azzam, C. Huyck, B. Mitchell, H. Cunningham, and Y. Wilks. 1998. Description of the LaSIE system as used for MUC-7. In *Proceedings of the Seventh Message Understanding Conference (MUC-7)*. http://www.itl.nist.gov/iaui/894.02/-related_projects/muc/index.html.

K. Humphreys, G. Demetriou, and R. Gaizauskas. 2000. Two applications of information extraction to biological science journal articles: Enzyme interactions and protein structures. In *Proc. of Paci£c Symposium on Biocomputing (PSB-2000)*, Honolulu, Hawaii.

G. Miller. 1995. WordNet: a Lexical Database for English. *Communications of the ACM*, Volume 38(Number 11), November.

M. Scott. 1996. *WordSmith Tools Manual*. Oxford University Press.

SICSTUS. Intelligent Systems Laboratory. Swedish Institute of Computer Science, PO Box 1263. SE-164 29 Kista, Sweden, 2000. *SICStus Prolog User's Manual*, May.