A Question-driven Information System adaptable via Information Extraction techniques

Emili Sapena, **Manuel Gonzalez**, **Lluís Padró** and **Jordi Turmo**
TALP Research Center
Universitat Politècnica de Catalunya
Barcelona, Spain
{esapena, manuelg, padro, turmo}@lsi.upc.edu

November 6, 2007

1 Introduction

This paper explores the application of NLP technology to build information systems oriented to the development of videogames. The scenario is a videogame where the user can interact with an agent in the game which manages lots of information about a certain domain. The user can ask or dialogue with the agent in the course of the game to achieve the desired goals.

The presented system is a demonstrator of the viability of this technology, as well as a proposal of an architecture upon which information systems for other domains or games can be more easily developed.

2 The Information System

The presented system is composed by two main parts: The Information Extraction Process, which crawls the web and extracts all relevant knowledge that is then stored in the database, and the Question Processor which interprets user questions and obtains a SQL statement that can be executed on the database.

The Information Extraction Process is executed in development time, the knowledge base is built, and the database frozen and exported to the exploitation (gaming) platform. The extraction process can be executed regularly, in order to release updated versions of the database. The architecture of the system is shown in figure 1.

In the following two sections both subsystems are described with a short explanation of each module. Next, section 5 describes the demonstration case developed on football (soccer) domain. Finally, some conclusions and future work are discussed.

3 The Information Extraction Process

The information extraction (IE) process consist of several tasks. Some of them extract the template elements (TE), template relations (TR) and scenario tem-

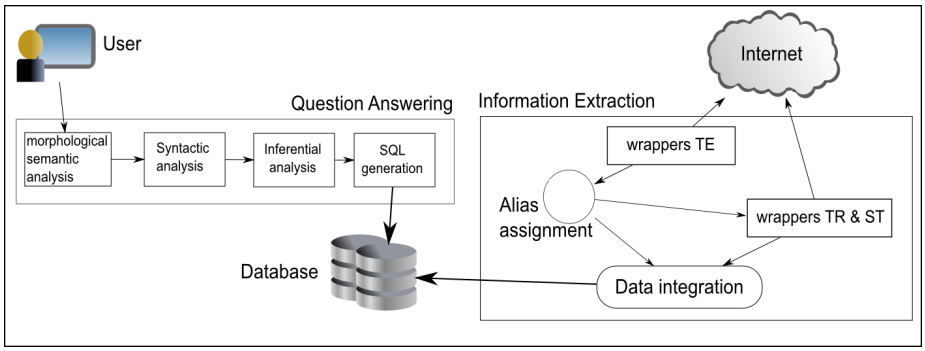

Figure 1: Global architecture of the system

plates (ST). The others keep the consistency of the data deleting repetitions or other impurities on the one hand, and integrating information on the other. Following, there is a description of each stage of the process:

- **Stage 1. Extract TE**. In a first stage, a set of wrappers crawl selected websites containing information about the domain. Each wrapper detects and recognize named entities (NE) and their attributes (TE). A wrapper can be as sophisticated as the website structure requires. Some wrappers only use a few handwritten regular expressions to get the data, while others need Augmented Transition Networks (ATN). Depending on the source structure, a wrapper should be designed in one way or another.

- **Stage 2. Alias Assignment**. When information is extracted from different sources, it usually has some redundancy. Moreover, each source may use different names to refer to the same entities. Variations in named entity expressions are due to multiple reasons: use of abbreviations, different naming conventions (for example "Name Surname" and "Surname, N."), aliases, misspellings or naming variations over time. In order to determine when two sources are referring to the same entities, a second stage, called alias assignment, is required. Alias assignment task decides if a mention in one source can be referring to one or more entities extracted from other sources.

- **Stage 3. Extract TR and ST**. After alias assignment, a disambiguation procedure will be required to decide which real entity among the possible ones is the alias pointing to in each context. The disambiguation procedure, however, is done at the end of the IE process because it will be relations between entities that can help to decide it. Therefore, in the third stage the set of wrappers extract relations (TR and ST) between elements already extracted but now considering that some of them can be alias of others. Nevertheless, some relations will remain as ambiguous pointing to a group of elements.

- **Stage 4. Data Integration**. Once all information from different sources has been extracted, the data may contain some repetitions, errors, ambigu-

ous relations and some relations broke down that can be integrated. Last stage solve all this uncleanliness executing a set of procedures.

- **Data Cleaning**. Some internal pages of the source websites have formatting errors or special fields that wrappers can not understand. This situation sometimes fall in erroneous elements or not sense relations between elements in the data. A *Data Cleaning* process is executed to find and delete them.
- **Time Interval Simplification**. Some relations between elements have an initial and a final date defining the time interval during which the relation exists. Reading this information from different documents results in a set of relations between the same elements, either consecutive or overlapped in time. This process finds this groups of relations and generates the minimum relations that express the same information.
- **Disambiguation and Deduplication**. Thanks to the alias assignment done in the second stage, we know that some elements can refer to the same real entity. Moreover, after TR and ST extraction, we have relations between them. Consequently, on one hand we have ambiguous relations and, on the other hand, we have groups of entities that can be the same. *Disambiguation and Deduplication* process uses algorithms of energy function optimization to finally integrate the data such as relax (Genís, 1989) or ants (Comellas and Ozon, 1998).

4 The Question Processor

The question processor is mainly composed by four modules: morphological-semantic analysis (executed in Freeling), syntactic analysis, inferential analysis and SQL generation. The input, questions written in natural language, is sequentially processed by these modules to become finally a SQL statement. Afterworks, the SQL statement is executed against the database to obtain the answers. Currently, the processor can handle simple sentences with temporal structures. Following there is an explanation of the main modules:

- **Morphologic-Semantic analysis**. First module uses Freeling (Atserias et al., 2006) to morphologically analyze the input sentence. Later, a syntactic analysis is done in the following module. Freeling has been modified to also output a set of λ *expressions* which enables to formalize each word as a quasi-logical form. For instance, *singer* = λ`(X)(singer(X), arg(X, person))`.
- **Syntactic and semantic analysis** is performed by using a DCG with the information obtained from previous step. A logical expression is generated, of the form: `interrogative(X), functor(X, condition1, condition2, ...)`. It is a typical structure where `X` is the answer to the question. The `interrogative` defines the kind of question (what, where, who, ...) and `functor` is the category of the target, i.e. the kind of answer we are looking for.

- **Inferential Analysis** performs simple inferences (e.g. related to temporal concepts) required for some complex questions.

- **SQL Generation**. The last module is an SQL statement generator. It builds a sentence in SQL that corresponds to the input logic expression. It uses a set of rules that relate every *functor* in the logic expression, the conditions and its relations with the appropriate element in the database in order to enable their translation to SQL statements. Changing this set of rules, the generator can be adapted to new databases or new domains.

5 Case of example

We have evaluated the system customizing it to the football domain. IE process gets information about players, clubs, championships, and so on crawling several websites. This information is integrated and stored in a database. The question processor uses this database to find the answers.

The wrappers used in this case have specific regular expressions for each website and have been manually developed. They use 5 different sources and the data offered in most of these websites is semi-structured. However, for a statistic site with a huge list of matches played with results and scorers, was more appropriate an Augmented Transition Network (ATN) to extract the data. It is because the information shown there is typed by hand by some volunteer and their format change depending on the author. The ATN developed is composed by about 70 states.

Once the first stage is finished, we have a lot of template elements that could refer to the same real entity. In this stage, we don't have yet enough information to definitely join or not these elements but we can assign some of them as a candidate alias of others. To assign some names like "Man Utd" or "J. Saviola" to "Manchester United Football Club" and "Javier Pedro Saviola" respectively, we train a Support Vector Machine pairwise classifier where each pair alias-entity is represented as a vector of features as explained in (Sapena, Padró, and Turmo, 2007). Some feature functions are domain-dependent and they need to be tuned if the system is used in other domains.

Kind of entity	Number of elements extracted	Different alias
People (players, coach, presidents and referees)	83.156	84.680
Clubs	25.105	25.320
Stadiums	772	-
Competitions	3675	167
Awards	2126	-
Matches	307.707	-

Table 1: Elements extracted and alias. Data before final disambiguation process

The disambiguation process uses the information of the relations between elements. The algorithm learns which relations help to the disambiguation and which not. A little knowledge about the domain and database structure is needed in order to tune and execute the process. Table 2 shows the results obtained with different algorithms in a small testing database for club and

person entities. The performance of the algorithms is presented in terms of the F-score between the purity and inverse purity of the groups found, as frequently used to evaluate clustering techniques.

Algorithm	Clubs	People
Baseline	54.5%	60.9%
Relax	66.3%	62.2%
Ants	79.3%	92.8%

Table 2: F1 measure for purity and inverse purity in groups of candidate alias at disambiguation process

When dealing with a different domain, the knowledge used by the Question Processor has to be adapted. Concretely, the lexicon has to be tuned for the domain, and the same happens with the inferential rules. The SQL generation module also needs to be adapted to the database structure.

The grammar is language dependent, thus moving to a new language would also require adjusting it. In our case of study, the questions are written in Spanish.

6 Future Work

Currently, we are working in the improvement of two different tasks. First, in the disambiguation process of the knowledge gatherer, we are studying some algorithms for the entity matching problem. Our goal is to achieve an information gathering system able to extract data from other sources incrementally. Second, we are working in the inferential analyzer to analyze more complex questions and improve the selectional restriction handling performed by the module.

References

Atserias, Jordi, Bernardino Casas, Elisabet Comelles, Meritxell Gonzlez, Llus Padr, and Muntsa Padr. 2006. Freeling 1.3: Syntactic and semantic services in an open-source nlp library. In *Proceedings of the fifth international conference on Language Resources and Evaluation (LREC 2006), ELRA*. Genoa, Italy.

Comellas, F. and J. Ozon. 1998. An ant algorithm for the graph colouring problem.

Genís, Carme Torrasi. 1989. Relaxation and neural learning: points of convergence and divergence. *J. Parallel Distrib. Comput.*, 6(2):217–244.

Sapena, Emili, Lluís Padró, and Jordi Turmo. 2007. Alias assigment in information extraction. In *Proceedings of SEPLN-2007*. Sevilla, Spain.

Stage	Result
User	*Quién marcó 1 gol en el partido* *Futbol Club Barcelona - Milan del 2 de Noviembre de 2004?*

Logic Expression	`pregunta(`
	`quien(_G1552),`
	`marcar(_G1552, goles(1),`
	`partido(_G1566,`
	`equipoLocal(futbol_club_barcelona), equipoVisitante(milan),`
	`intervalo(2/11/2004, 2/11/2004))))`

SQL final statement	`Select distinct TE_people1.name`
	`From ST_plays as ST_plays2, TE_people as TE_people1,`
	`ST_plays as ST_plays1, TE_match as TE_match3,`
	`TE_teams as TE_teams4, TR_playLocal as TR_playLocal3,`
	`TE_teams as TE_teams5, TR_playVisitor as TR_playVisitor3`
	`Where ST_plays2.goals='1' and TE_people1.id=ST_plays2.idPerson`
	`and TE_people1.id=ST_plays1.idPerson`
	`and TE_match3.id=ST_plays1.idMatch`
	`and TE_teams4.name='futbol club barcelona'`
	`and TE_teams4.id=TR_playLocal3.idTeam`
	`and TE_match3.id=TR_playLocal3.idMatch`
	`and TE_teams5.name='milan'`
	`and TE_teams5.id=TR_playVisitor3.idTeam`
	`and TE_match3.id=TR_playVisitor3.idMatch`
	`and '2004-11-2'<=TE_match3.date`
	`and '2004-11-2'>=TE_match3.date`
	`and TE_match3.id=ST_plays2.idMatch`

Answer	*Andriy Mykolayovych Shevchenko* *Samuel Etoo Fils* *Ronaldo Assis de Moreira*

Table 3: Example of the process followed by a question