Cross-Cutting Aspects of Cross-Language Question Answering Systems

Bogdan Sacaleanu
Language Technology Group
DFKI GmbH
Saarbrücken, Germany
bogdan@dfki.de

Günter Neumann
Language Technology Group
DFKI GmbH
Saarbrücken, Germany
neumann@dfki.de

Abstract

We describe on-going work in the development of a cross-language question-answering framework for the open domain. An overview of the framework is being provided, some details on the important concepts of a flexible framework are presented and two cross-cutting aspects (cross-linguality and credibility) for question-answering systems are up for discussion.

1 Introduction

Different projects, different evaluation forums, different tasks, different languages, different document collections, different question types, different answer processing strategies … Anyone familiar with all these concepts knows the complexity and what a daunting prospect of developing a QA-System easily adaptable to ever changing requirements this might be. We have started off with a "prototype-and-go" approach, trying to keep pace with the emergence of new tasks and managing the scarcity of your time and of your resources, to realize later on that what we had is a bunch of prototypes very tuned to their task requirements. Trying to adapt them to new requirements seemed often more difficult then starting off with a new one. Therefore we started looking for an alternative, which should be more flexible and should allow us to cover much more requirements' variations; in other words we were considering putting together a Question Answering framework.

In the rest of the paper we will shortly overview the components of such a framework and will describe the relevant aspects of the solution offered for each of them, aspects that should account for a large variety of question types, document collections and answer processing techniques, as well as for several languages. We will continue with a discussion of two issues that cut across several components of the framework, namely: cross-linguality and answer credibility, and will conclude by shortly naming the domains of usage for the framework and future work.

2 Framework Overview

Based on an existing set of cross-language Question Answering prototypes developed for different requirements, we began by looking for the commonalities among them. Following is a list of `reusable` components that might be considered as a starting point in defining a QA framework (see Figure 1).

Several components along the work-flow of a typical QA system were identified: a `Unit Alignment` component in cross-language environments and a `Query Expansion` component for the Question Analysis task; a `Unit Processor` and a `Query Generator` component for the Information Retrieval task; a `Mention Chain` component for the Answer Extraction task and a `Scoring Strategy` for the Answer Selection task. (see Figure 1)

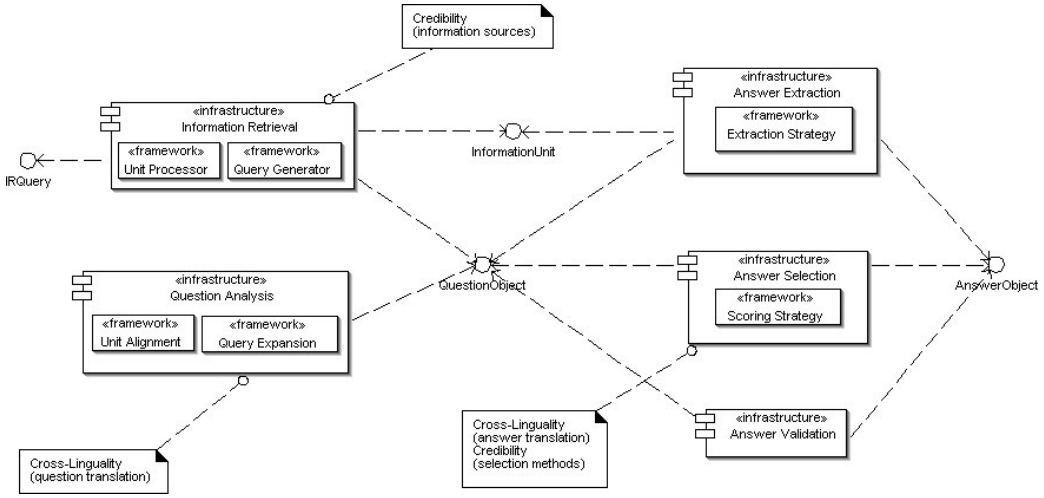

Figure 1: Framework Architecture

Beside the reusability aspect in context of a QA framework, `extensibility` is a further important issue. We have approached it by concentrating on a flexible representation format of the data being passed around in the framework, namely the representation of the `QuestionObject`, the `InformationUnit` and the `AnswerObject`.

2.1 Reusability

"Reusability is the likelihood a segment of structured code can be used again to add new functionalities with slight or no modification". (Wikipedia)

In a cross-language setting of a QA system, which is crossing the language barrier at the user end rather then at the document end, there are two approaches for getting the formalized user information need (`QuestionObject`) to the documents' language: either creating it based on the question translation or analyzing the original question and translating the formalized result. This last approach is convenient when machine readable dictionaries, part-of-speech taggers and other bilingual lexical resources (i.e. WordNets) for both languages are available. For this purpose a `Unit Alignment` Component was designed that produces an alignment of simple (words) and complex (phrases) syntactic units from the source to target language.

`Query Expansion` is another component present among the support structures of our framework. Backed by lexico-semantic resources as EuroWordNet [V98] it can be used for all languages supported by these resource. For a given input word, it can return synonyms, hyponyms and hypernyms according to the following algorithm:

```
if (trgtWord_is_not_ambig)
      return Information;
else if (trgtWord_is_ambig)
{
TRANS:
      1. translate Question
      2. do Unit Alignment
      3. if (transTrgtWord_is_EWN_aligned)
            a. if (alignment_is_not_ambig)
                  return Information;
            b. else if (alignment_is_ambig)
                  save Alignments;
                  goto TRANS;
}

intersection = intersect_saved_alignments();
if (intersection.size == N)       // strict N=1
      return Information_for_intersection;
return NULL;
```

An example for the above algorithm up to the stage 3.a is given in Section 3.1. This represents the ideal case, when our input can be disambiguated using the alignments of a question translation. But more often it is the case of advancing over this stage to 3.b, when the ambiguous EuroWordNet alignments are being saved and a new translation of the question through other online translation services is attempted. The idea behind this expansion method is that lexical diversity of different translations could narrow down the ambiguity of a word to a desired level (N).

To select an initial set of information units relevant to the question, traditional search engines are being used to scale down the search space. Two important aspects in this process are: the level of detail at which the indexation units are chosen and the granularity of the information units (be it document, paragraph, sentence or phrase). Two subcomponents are important at this stage: the `Unit Processor` and the `Query Generator`.

The `Unit Processor` covers the above-mentioned aspects: it takes as input an `InformationUnit` (i.e., a raw text document) and it either reduces it to a set of new units (i.e., sentences), or it annotates the unit at different levels (i.e., named entities, grammatical relations). This way, by chaining different `Unit Processors`, you can both reduce the information unit size and generate new indexing units.

The `Query Generator` relies on an abstract description of the processing method to accordingly generate the `IRQuery` to make use of the advanced indexation units. For example, when named entities were annotated during the `Unit Processor` chaining and used as indexing units, the `Query Generator` will adapt the `IRQuery` so as to search for an additional field (`neType`) that might reflect the expected answer type for the question considered. We consider the `Query Generator` as being the mediator between the question analysis result `QuestionObject` (answer type, constraints) and the search engine serving the retrieval component with information units (documents, sentences). Even more, the `Query Generator` relies on an abstract description of the search engine, too, and can adapt the `IRQuery` according to either a boolean or a ranked search engine.

A `Mention Chain` component in the Answer Extraction task provides an ease of burden for the Selection task by computing answer reference chains. This is very helpful for factoid questions on the Web, and not only, where redundancy of an answer candidate is a good hint for its potential selection and credibility. A mention chain contains all answers sharing a common normalized representation, determined either through the string similarity of the answers only or by additionally employing context entailment measures.

The `Scoring Strategy` component builds on the mathematical graph theory and reduces the answer candidate scoring issue to a shortest path problem on lexical graphs. In most of the cases the answer "suitability" could be scaled down to computing a distance metric for the answer and some information in the question (i.e., keywords, focus). Both a simple textual distance measure and another one based on dependency structures were implemented on these graph structures, with slight variations making use of weight and cost properties for graph edges.

Based on available web search API (i.e., Google, Yahoo) the `Answer Validation` component computes a total frequency count of co-occurrence for pairs of question and answer, assuming that the right answer shares more contexts with the question than any other candidate and that the considered answers are semantic independent and insensitive with respect to the timeline preferred by the search engines.

2.2 Extensibility

"Extensibility is a system design principle where the implementation takes into consideration future growth. … The central theme is to provide for change while minimizing impact to existing system functions." (Wikipedia)

Extensibility can be approached by two methods during framework design: through software design patterns and through a common extensible data representation format. While we have used some patterns through the development of reusable components like the chaining aspect in the Unit Processor, the normalization function in the Mention Chain and the graph design in the Scoring Strategy, we have concentrated more on an extensible representation format for data being passed around through the framework: the `QuestionObject`, the `InformationUnit` and the `AnswerObject`. For this purpose we have used XML, XML Schema and data binding methods (JAXB) to guarantee component life on an evolving data format. The primary benefit of allowing extensibility in a format is that it enables a format to evolve without requiring central control of the format. A secondary benefit is that it allows the format to stay focused and simple by pushing specialized cases and complex solutions into optionally supported extensions. W3C XML Schema provides two features that promote extensibility in XML vocabularies: the wildcards `xs:any` and `xs:anyAttribute` are used to allow the occurrence of elements and attributes from specified namespaces into a given format, and the `xsi:type` attribute that can be placed on an element in an XML instance document to change its type to a more refined subtype. That

is, according to user's need, the data exchanged among the framework's components can be extended, without changing the framework functionality.

3 QA Cross-Cutting Aspects

In a question answering framework there are aspects that does not directly relate to the core concerns, but are needed for a proper design. Moreover, they can hardly be pinned down to a component, as they cross several core components of the system. We are talking about concepts like cross-linguality and credibility – system credibility reflected in answer credibility.

3.1 Cross-Linguality

There are three traditional approaches that count for cross–linguality in context of information management systems:

- translating the queries into the target language,
- translating the document collection into the source language or
- translating the queries and the documents into an intermediate representation (inter–lingua).

Two types of translation services are well known within this context which are based on

- lexical resources (e.g., dictionaries, aligned wordnets), or
- machine translation (e.g., example–based translation).

The only feasible approach when dealing with huge amounts of data, as is the case for question answering systems, is translating the question into the language of the document collection and the related issue of back-translating the answers into the language of the user.

We are using two different methods for responding questions asked in a language different from the one of the answer-bearing documents. Both employ online translation services (Altavista, FreeTranslation, etc.) for crossing the language barrier, but at different processing steps: before and after formalizing the user information need into a `QuestionObject`.

The `before-method` translates the question string in an earlier step, resulting in several automatic translated strings, of which the best one is analyzed by the Question Analysis component and passed on to the Information Retrieval component. This is the strategy we use in an English–German cross-lingual setting. To be more precise: the English source question is translated into several alternative German questions using online MT services. Each German question is then parsed with SMES [NP02], our German parser. The resulting query object is then weighted according to its linguistic well–formedness and its completeness wrt. query information (question type, question focus, answer–type).

The assumption behind this weighting scheme is that "a translated string is of greater utility for subsequent processes than another one, if its linguistic analysis is more complete or appropriate."

The `after-method` translates the formalized result of the Query Analysis Component by using the question translations, a language modeling tool and a word alignment tool for creating a mapping of the formal information need from the source language into the target language. We illustrate this strategy in a German–English setting along two lines (using the following German question as example: `In welchem Jahrzehnt investierten japanische Autohersteller sehr stark?`):

- translations as returned by the on-line MT systems are being ranked according to a language model

```
In which decade did Japanese
automakers invest very
strongly? (0.7)
In which decade did Japanese
car manufacturers invest
very strongly? (0.8)
```

- translations with a satisfactory degree of resemblance to a natural language utterance (i.e. linguistically well-formedness), given by a threshold on the language model ranking, are aligned based on several filters: back-propagation dictionary filter - based on MRD (machine readable dictionaries), PoS filter - based on statistical part-of-speech taggers, and cognates filter - based on string similarity measures (dice coefficient and LCSR (lowest common substring ratio)).

```
In: [in:1]
welchem: [which:0.5]
Jahrzehnt: [decade:1]
investierten: [invest:1]
```

```
japanische: [Japanese:0.5]
Autohersteller:
    [car manufacturers:0.8,
      auto makers:0.1]
sehr: [very:1]
stark: [strongly:0.5]
```

The evaluation gives evidence that both strategies are comparable in results, whereby the last one is slightly better, due to the fact of not being forced to choose a best translation, but working with and combining all the translations available. That is, considering and combining several, possible different, translations of the same question, the chance of detecting a translation error in an earlier phase of the work–flow becomes higher and avoids error propagations through the whole system.

The related issue of back-translating is explored by looking for parallel data to the answer's context or metadata, and extracting translation candidates based on their context and string surface similarity. For example, in a CLEF setting, for a German question having as English answer "Yasser Arafat" we have extracted the time-stamp of the answer's context (19.07.1994), collected all the data with a time-stamp of 07.1994 in the source language, extracted the named entities of type PERSON and then aligned "Jassir Arafat" based on its string surface similarity to the initial answer.

The translations and their alignment to the original question, according to the above-mentioned `after-method`, have also a positive side-effect, namely: some of the aligned words may have several ranked translations. As it is the case of the "Autohersteller", a word might consider the best ranked alignment ("car manufacturers") as its direct translation and the remaining ones as its expanded words. As such, given a reliable alignment method, cross-linguality can prove supportive even for `Query Expansion`. Moreover, another method of usage can confirm the added value of cross-linguality for `Query Expansion`, as described below.

For this task we are using the German and the English wordnets aligned within the EuroWordNet [V98] lexical resource. Our goal is to extend the formalized information need `Question Object` with synonyms for the words that are present in the wordnet.

Considering the ambiguity of words, a WSD module is required as part of the expansion task. For this purpose we are using both the original question and its translations, leveraging the reduction in ambiguity gained through translation. Our devised `pseudo-WSD algorithm` works as following:

1. look up every word from the word-translation alignment (see example above) in the lexical resource;

2. if the word is not ambiguous (which is, for example, the case for `Japanese`) then extend the Question Object with its synonyms (e.g., `[Japanese, Nipponese]`);

3. if the word is ambiguous (e.g., invest) then

(3a) for every possible reading of it, get its aligned German correspondent reading (if it exists) and look up that reading in the German original question , e.g.,

```
1351398: adorn-clothe-invest (EN)
1351223: invest-vest (EN)
1400771: empower-endow-endue-gift-
indue-invest (EN)
1350325: induct-invest-seat (EN)
1293271:
```

- commit-invest-place-`put (EN)`
- anlegen-investieren (DE)

(3b) if an aligned reading is found (e.g., Reading-`1293271`) retain it and add the English synonyms of it to the Question Object, i.e., expand it with:

```
commit, place, put
```

Following the question expansion task, the Question Object has been enriched with new words that are synonyms of the un–ambiguous English words and by synonyms of those ambiguous words, whose meaning(s) have been found in the original German question. Thus our expanded example has gained several more expanded words as follows:

{`Nipponese,commit,place,put`}

3.2 Answer Credibility

In the ideal case, a Questions Answering System (QAS) will deliver correct answers *and* knows that they are correct, i.e., it can deliver a proof of the correctness of the answers. However, at least for the case of open-domain textual QA applications, this is out of reach with current technology. Thus, current QAS can only deliver answers with certain *trustworthyness*. Since, a receiver of an answer usually assumes, that a QAS tries to identify the best answer possible (at

least for cooperative language games), the QAS should assign a credibility measure to each selected answer. The underlying decisions made, can then also be used for explaining, how the answer's credibility was determined.

We view answer credibility as an additional process to answer extraction and selection that determines the quality of identified answer candidates by checking the plausibility of the answer sources and context on basis of *meta* information. For example, useful document and web page information might be:[1]

- The name of the author of this activity;
- Textual fingerprints of authority, e.g., "official web page of US government";
- E-mail address of the contact person for this activity;
- When was this webpage last updated and links were checked (also: is there an regular update);
- The name of the host school or organization;
- The link structure of the document, e.g., link IN/OUT density, links to relevant people, other authorities, clusters / hierarchies of authorities;
- Text structure, e.g., textual coherence or style;

Another important source of meta information relates to ontological knowledge, e.g., the consistency of contextual information with respect to a domain ontology that defines the scope of the answer candidates. By this we mean the following:

- Given an answer candidate A of a question Q (e.g., an instance of a concept in question) determined by a web-based QAS.
- Check the textual context of A concerning the mentioning of other relevant facts/concepts, that can be determined via access to an external (domain) ontology using relevant terms from Q and A.

For example, if the following request is sent to a web-based QAS: *Name IE-systems that are based on statistical methods.* Assume that for this question, the QAS identifies a list of names of IE-systems from textual sources as answer candidates. Assume further, that the QAS has access to an ontology-based meta-store about Language Technology terms, cf. [JU05]. Then, answer credibility checking can use the query terms *IE-system* and *statistical method* for extracting relevant facts from this LT-store, e.g., names of statistical methods, IE experts, IE subtasks or properties of the concept *information extraction*. Next, for each answer candidate, check the textual context of the answer for the mentioning of these terms and their mutual relationship. Then a possible credibility heuristics might decide that the more relevant domain-knowledge can be identified in the context of the answer the higher is the credibility of this answer.

These examples demonstrate that a wide variety of metadata from different levels can be and should be exploited for determining the credibility of answers. Since answer credibility is a new research area, it is still unclear which level of information is best suited for which kind of questions and answers. In order to be able to investigate many possible credibility scenarios, we consider answer credibility as a complex abstract data type or QAS credibility model, **QAS-CM** for short. The QAS-CM allows the definition of different kinds of credibility parameters, which are related to corresponding meta data and are orthogonal to the component-oriented view of the QAS. The values of the parameters might be computed and evaluated by different components, even in parallel. Thus, the credibility of an answer is a complex value determined through composition of component-related credibility values. We are distinguishing two types of metadata:

- **Static metadata**: are available directly through the textual source of an answer candidate and are represented in form of annotations, e.g. HTML/XML tags.
- **Dynamic metadata**: are computed online via the components of the QAS, e.g., linguistic entities, semantic relations, textual entailment, text structure and coherence, topic linkage.

[1] For information on the topic of Web Credibility, cf. http://credibility.stanford.edu/ . This URL links to the Web Credibility Project of the Stanford Persuasive Technology Lab. Although they do not consider credibility under a strict QA perspective as we do, their work and results are a rich source of inspiration.

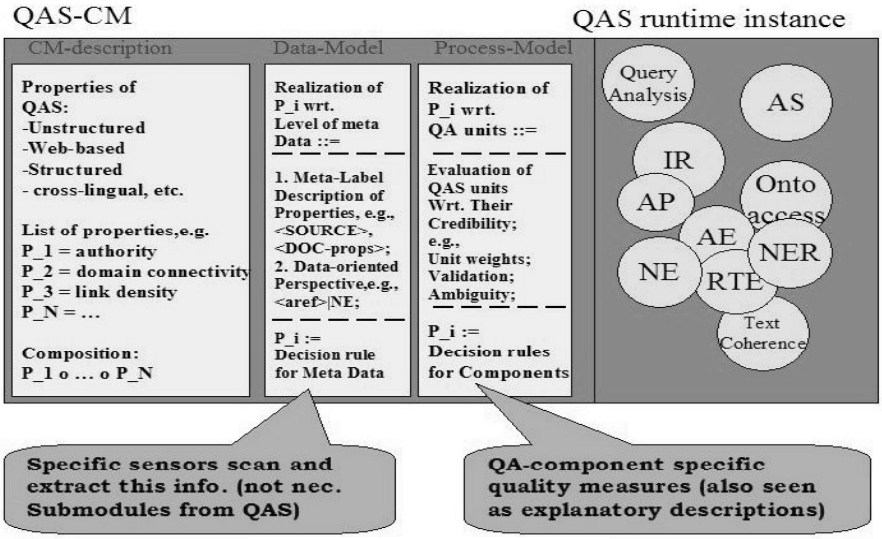

Figure 2. Credibility Model

Following this perspective, we propose the following structure for the QAS-CM (see Figure 2). We distinguish two major subsystems: the QAS-CM itself and the QAS runtime instance. The QAS-CM is further subdivided into three modules:

1. CM-description
2. Data-Model
3. Process-Model

The **CM-description** module defines the credibility properties in a declarative way independently from the specific QAS instance used. It defines the name space of the credibility properties that are to be implemented and the way they are composed. We assume that each individual credibility property returns a real number, i.e., we consider it as a *utility function*, cf. [RN03]. Thus, the composition rules define how a complex value is computed, e.g., by summation or multiplication. The CM-description also contains specification about the actual QAS instance, e.g., the type of document source or natural language to be processed. The CM-description serves as a blueprint for the other two modules, the Data-Model and the Process-Model.

The **Data-Model** implements credibility properties as decision rules over instantiated meta information taking into account either static or dynamic meta data. For example a rule like

```
entails(<SOURCE>,"official
web page of <LOCATION>") →
authority:=αI
```

assigns the weight α_I to the credibility property with name `authority` if the document source of the answer candidate has a tag `<source>` which contains a substring "`official web page <LOCATION>`", where `<LOCATION>` contains a location name which corresponds to one mentioned in the Wh-question. For a question like "Name all German chancellors after World War II.", then an answer with a web page source, that contains something like `<SOURCE> … official web page of Germany … </SOURCE>` will receive a credibility value α_I.

The **Process-Model** is related to the components of the QAS that have been used for the processing of a question/answer pair. We assume that each component can assign a utility measure for its determined output. In doing so, a component should also take into account the size and utility measure of its input and the ambiguity degree of its output etc. For example, assume that the question analysis component (cf. sec. 3) can recognize the syntactic quality of a parse tree of a Wh-question, as being `complete`, `partial` or `null`, cf. [NS05a]. Then a possible decision rule for the Process-Model might be

```
syntactic_quality(<QUESTION>
, COMPLETE) → QueryAna-
lyser:=αI=1
```

in case, the parser of the question analysis component was able to completely parse the Wh-question `<QUESTION>`. In a similar way, the web-validator mentioned in sec. 2.1 can also be integrated as a credibility function into our QAS-CM using the following decision rule:

`validate(Web, <QUESTION>, <CurrAnsw>) → WebValuator:=`α_I

Note that in case statistical components are used in QAS, e.g., a stochastic NE-recognizer, then the probabilities of the those components can be used for defining the utility values for the representative decision rules. Furthermore note, that each decision rule of the Process-Model corresponds to a single component. The compositon of several such rules are defined in the CM-description module.

We are currently implementing QAS-CM following the same object-oriented framework as described in sec. 3, which eases integration of the components of the QAS. The data and the process model are implemented as a set of production rules using the RuleML language, cf. http://www.ruleml.org/. Currently, we define the values of α_I manually. This is more easier to implement but more tricky to maintain because of potential mutual interdependencies between individual values. Therefore, in the next development cycle of our QAS-CM, we will use a statistical approach for automatically acquiring optimal parameter settings. Starting point will be a question/answer/document corpus. This corpus can directly be used for training the Data-Model. In the case of the Process-Model, a Bayesian Network will be dynamically trained following ideas from IE-research, cf. [PP03]. Since in this case we also need output from all major QAS components, we are integrating an exhaustive tracing mechanism into our QA framework as a basis for automatically determining initial training material.

4 Results

The presented QA framework has been used both in mono-lingual and cross-lingual scenarios for closed document collections (CLEF collection) and open document collections (World Wide Web). In terms of reusability and extensibility, the framework allowed for up to two weeks of work for building fully functional QA systems for different use scenarios. In terms of time performance for systems build upon the framework, the following figures apply: for systems used to query the Web a response time of up to 20 seconds in mono-lingual settings could be measured; those querying the CLEF document collection in a mono-lingual setting (German only) registered a latency of up to 3 seconds and for a cross-lingual setting of up to 15 seconds. The qualitative performance has been measured only on closed document collection and the best results for 200 questions of the CLEF 2005 evaluation campaign in different use scenarios, according to [NS05b], were as follows:

	Right		**Wrong**	**Inexact**
DeDe	87	43.5%	100	13
DeEn	51	25.5%	141	8
EnDe	46	23%	141	12

Reference

[JU05] B. Jörg and H. Uszkoreit. The Ontology-based Architecture of LT World, a Comprehensive Web Information System for a Science and Technology Discipline. Leitbild Informationskompetenz: Positionen - Praxis - Perspektiven im europäischen Wissensmarkt. 27. Online Tagung, 2005.

[NP02] G. Neumann and J. Piskorski. A shallow text processing core engine. Computational Intelligence, 18(3):451–476, 2002.

[NS05a] G. Neumann and S. Sacaleanu. Experiments on robust NL-question interpretation and multi-layered document annotation for a cross-language question/answering system. In Clef 2004, volume 3491. Springer-Verlag LNCS, 2005.

[NS05b] G. Neumann and B. Sacaleanu. DFKI's LT-lab at the CLEF 2005 Multiple Language Question AnsweringTrack. In Working Notes for the CLEF 2005 Workshop, 21-23 September, Vienna, Austria, 2005.

[PP03] L. Peshkin and A. Pfefer. Bayesian Information Extraction Network. In proceedings of IJCAI, 2003.

[RN03] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Englewood Cliffs, NJ, 2nd edition, 2003.

[V98] Vossen, P. (eds) 1998 *EuroWordNet: A Multilingual Database with Lexical Semantic Networks*, Kluwer Academic Publishers, Dordrecht.