Question Answering from Structured Knowledge Sources

Anette Frank [a] Hans-Ulrich Krieger [a] Feiyu Xu [a]
Hans Uszkoreit [a] Berthold Crysmann [a] Brigitte Jörg [a]
Ulrich Schäfer [a]

[a] *German Research Center for Artificial Intelligence, DFKI*
Stuhlsatzenhausweg 3
66123 Saarbrücken, Germany

Abstract

We present an implemented approach for domain-restricted question answering from structured knowledge sources, based on robust semantic analysis in a hybrid NLP system architecture. We perform question interpretation and answer extraction in an architecture that builds on a lexical-conceptual structure for question interpretation, which is interfaced with domain-specific concepts and properties in a structured knowledge base. Question interpretation involves a limited amount of domain-specific inferences, and accounts for higher-level quantificational questions. Question interpretation and answer extraction are modular components that interact in clearly defined ways. We derive so-called *proto queries* from the linguistic representations, which provide partial constraints for answer extraction from the underlying knowledge sources. The search queries we construct from proto queries effectively compute minimal spanning trees from the underlying knowledge sources. Our approach naturally extends to multilingual question answering, and has been developed as a prototype system for two application domains: the domain of Nobel prize winners, and the domain of Language Technology, on the basis of the large ontology underlying the information portal LT World.

Key words: QA, hybrid NLP, multilinguality, RMRS, question semantics, ontology modeling, data base queries

[1] The research reported here has been conducted in the project Quetal, funded by the German Ministry for Education and Research, BMBF, grant no. 01 IW C02. Special thanks go to Bogdan Sacaleanu for implementation of a QA-control server that connects question and answer processing, and to Gregory Gulrajani for advice in the setup of a Sesame server. Günter Neumann and Bogdan Sacaleanu provided major contributions to the realisation of the overall Quetal QA architecture.

1 Introduction

The recent TREC and CLEF evaluation fora have engendered significant progress in the underlying research and the performance of practical QA systems. While these competitions focus on open-domain textual QA on the basis of large document bases or the WWW, there is increasing need for question answering in restricted domains, due to several reasons: First, where open-domain QA exploits the wealth of information on the Web, it is also confronted with the problem of reliability: information on the Web may be contradictory, outdated, or utterly wrong. Second, the utilisation of formalised knowledge in a restricted domain can improve accuracy, since both questions and potential answers may be analysed with respect to the knowledge base (Fleischman et al., 2003). Third, there is a need for accurate specialised information management solutions in both business intelligence and public administration.

QA systems for restricted domains may be designed to retrieve answers from so-called unstructured data (free texts), semi-structured data (such as XML-annotated texts), or structured data (databases). Question answering applied to restricted domains is interesting and challenging in two important respects. Restricted domains tend to be small and stable enough to permit careful knowledge and data modeling in terms of structured knowledge bases, and can therefore serve as certified information sources. Whenever such structured data can be exploited, this offers clear advantages over open text QA. More importantly though, QA in restricted domains requires techniques that crucially differ from the techniques that are currently applied in open-domain textual QA. Since document volumes tend to be small, textual QA techniques cannot exploit data redundancy. Further, both in domain-restricted textual QA and QA from structured knowledge sources, we cannot expect the answer to a given question to follow directly from some textual passage or to be explicitly represented in the knowledge base. Yet, despite a tendency towards deeper analysis, current techniques in QA are still knowledge-lean, in exploiting data redundancy and paraphrasing techniques.

Since the question is the primary source of information to direct the search for the answer, a careful and high-quality analysis of the question is of utmost importance in the area of domain-restricted QA. Most importantly, since we need to answer questions where the answer cannot be directly derived from the underlying document or knowledge base, we need a semantic interpretation of the question that can be tightly connected to the domain knowledge sources and the process of answer extraction.

We present an approach to domain-restricted QA from structured knowledge sources that starts from these considerations. We focus on a high-quality linguistic analysis of the question, with a lexical-conceptual interpretation of the

question relative to the chosen application domain. Our approach extends to multilingual QA scenarios and provides a natural interface to the underlying knowledge bases, enabling flexible strategies for answer extraction.

In Section 2 we give an overview of the architecture and the base components of the system. The system presented in this paper is embedded in a hybrid QA system architecture, the QUETAL QA system. Within this architecture we are using a hybrid NLP platform that integrates a wide range of NL processing tools. We introduce the main aspects of domain modelling for our two application domains: Nobel prizes and Language Technology.
Section 3 describes our approach to question analysis. We start from HPSG analyses of questions, which are enriched with a lexical-conceptual representation that can be further modified by domain-specific inference rules. We show how our account to question interpretation extends to multilingual question answering. Section 4 describes the interface between question interpretation and domain ontologies for query processing. We define a mapping between the lexical concepts used in semantic question interpretation and the concepts in the underlying domain ontology. This mapping is used to extract so-called *proto queries* from the semantic representation of the question. Proto queries are abstract query patterns in a higher-level query language that are translated to concrete database or ontology query language constructs in the answer extraction phase. Section 5 goes into the details of the concrete query construction and explains how a proto query can be (partially) mapped to SQL in the MySQL system and to SeRQL in the Sesame RDF framework.
In Section 6 we perform an evaluation of our current system, and discuss prospects for future developments. Section 7 compares our approach to related work in the field of natural language interfaces to databases. Section 8, finally, concludes with prospects on future work.

2 Architecture

2.1 Overall system architecture

The QA system for structured knowledge sources described below is part of a general QA system architecture, the QUETAL architecture. [2] The hypothesis underlying the QUETAL architecture design is that QA systems perform best if they combine virtues of domain-specialised and open-domain QA, accessing structured, semi-structured, and unstructured knowledge bases.
The core idea is that—instead of providing specific information portals (with system-specific user interfaces)—the QUETAL system offers a single and uni-

[2] See the QUETAL project homepage at http://quetal.dfki.de.

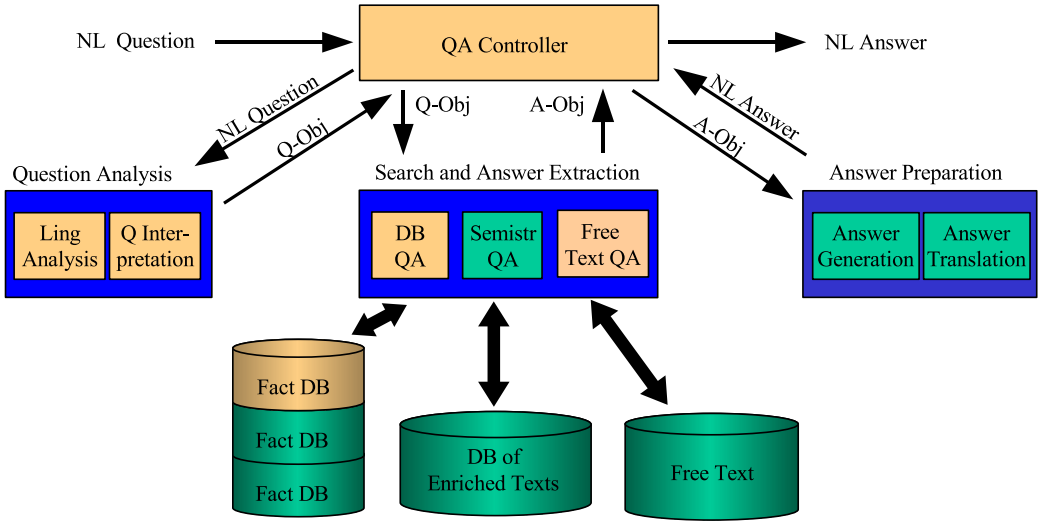

Fig. 1. Hybrid QA system architecture.

form natural language-based QA access to different information sources that exhibit different degrees of structuring.

The QUETAL architecture is hybrid in two senses. The question analysis is hybrid in that shallow and deep NLP are combined to yield both robustness and a rich semantic representation of questions. The answer document base is hybrid in that three types of information sources are employed: (i) *unstructured* text retrieved via Web-based or local full-text search engines and information retrieval systems, (ii) *semi-structured* text that has been enriched offline with IE and NLP techniques, (iii) *structured* fact databases, for instance, ontologies and traditional relational databases that contain domain-specific facts, relations and concepts. [3] As depicted in Figure 1, the QA process starts with linguistic analysis and a subsequent interpretation of the question. After a question type has been identified together with an expected answer type, both represented in a so-called Q(uestion)-Object, one (or more than one) information source is selected to retrieve answer candidates. From the returned A(nswer)-Object, an answer is prepared. As QUETAL supports crosslingual QA, intermediate translation stages after question analysis and answer generation are integrated (Neumann and Sacaleanu, 2003, 2004).

2.2 Architecture for domain-restricted QA

The architecture for domain-restricted QA from structured knowledge sources (Figure 2) is embedded in the general QUETAL architecture. A question is linguistically analysed by the Heart of Gold (HoG) NLP architecture, which flexibly integrates deep and shallow NLP components (Callmeier et al., 2004),

[3] See Neumann and Sacaleanu (2003, 2004) for textual QA in QUETAL. Semi-structured text will be addressed at a later stage of the project.

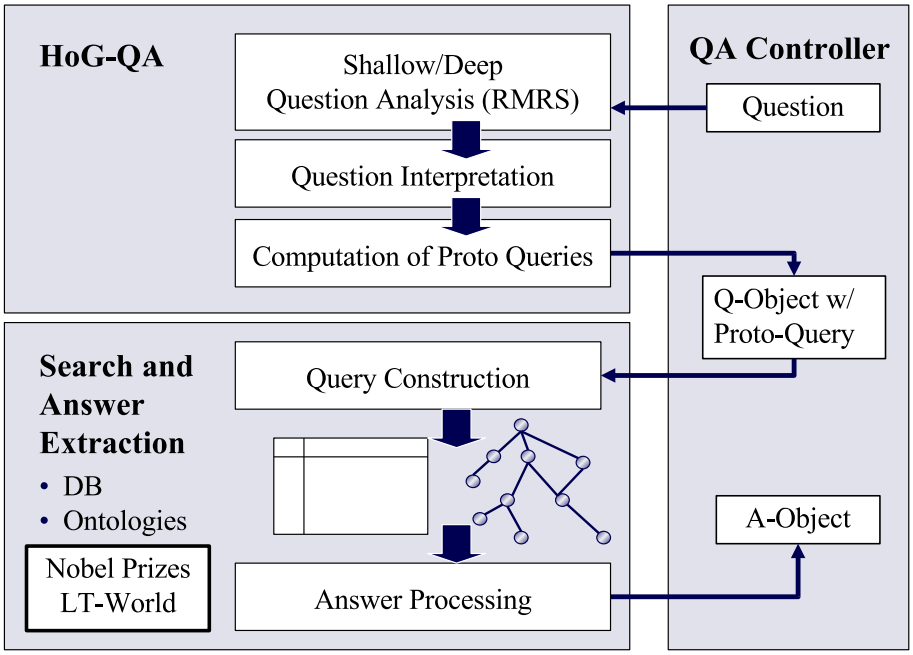

Fig. 2. Architecture of domain-restricted QA from structured knowledge sources.

for instance, PoS tagger, Named Entity recognition and HPSG parser. The semantic representations generated by the Heart of Gold are then interpreted and a question object is generated that contains a proto query. This proto query can be viewed as an implementation-independent, 'higher-level' representation of a database or ontology query. From this, an instance of a specific database or ontology query is constructed. From the result(s) returned by the queried information source, an answer object is generated which forms the basis for subsequent natural language answer generation. The individual stages of this QA process are described in detail in Sections 3 to 5.

2.3 Domain modeling and inference services

This section addresses the modelling of the two domains we have chosen to show the applicability of our approach: the relatively small Nobel prize domain and the larger LT World domain. The main reason for including more than one domain is to be able to show that the modularity of the architecture facilitates portability from one domain to another. Another reason was to demonstrate scalability from a relatively small domain (Nobel prizes) to a considerably more complex one. Even though building up a domain (an ontology/a data base) is a time-intensive knowledge engineering task, these domains were already available and are actively developed (see also the discussion in Section 6.5). Furthermore, our approach shows that a lot of linguistic knowledge (mostly the HPSG grammar) can be reused, independent of the underlying domain. The current tendency that upper- and mid-level ontolo-

gies become standardised and that domain ontologies refer to these general ontologies, will make it even easier to interface our system to a new domain.

2.3.1 The Nobel prize ontology and data base

The domain ontology plays a crucial role in our approach. It is used as the interface between question analysis, answer extraction and knowledge database design. We chose *Nobel Prize* as our initial subject domain, since it is a domain for which both complete records of all awarded prizes in structured formats and thousands of free texts about awards and laureates can be found on the web. [4] Furthermore the data are manageable in size, authoritative and can be used as our gold-standard for evaluation of the QA task. Here we focus on the exploitation of the structured data for QA.

We started our specification with an existing general ontology as reference. Two sources have been selected: the knowledge-engineering-based top-level ontology SUMO (Niles and Pease, 2001) and its mid-level specification MILO (Niles and Terry, 2004), and on the other hand the structured thesaurus WordNet (Miller et al., 1993). Since there is a mapping between the artificial concepts in SUMO and the word senses in WordNet (Niles and Pease, 2003), we chose the SUMO ontology as our backbone and define sub-concepts by referring to the mapping between SUMO concepts and WordNet word senses.

The main concepts in our application domain are `prize`, `laureate`, `prize-area`, including domain-independent general concepts, such as `person` or `organization`. Figure 3 lists some of the mappings between domain concepts and SUMO concepts. `laureate` corresponds to the SUMO concept `cognitiveAgent`, inheriting therefore its two subconcepts `human` and `organization`. Most subconcepts of the concept `prize-area`, except for `Peace`, are subconcepts of the general concept `fieldOfStudy`, for example, `Chemistry`. Each concept is further specified by its attributes: `person` is assigned the attributes `firstname` and `surname`. The concepts are organized via hierarchical relations. In addition to the domain-specific relations, such as `nobel-prize-nomination`, we also model some general relations like `person-affiliation`.

2.3.2 The LT World *ontology and data base*

As our second scenario for domain-restricted QA, we have chosen the Language Technology World information portal LT World (http://www.lt-world.org). LT World is an ontology-based virtual information center on the wide spectrum of Human Language Technology, providing information about people,

[4] Peace Nobel Prizes (as many other prizes) can be also awarded to organisations and not just to persons.

type	domain	SUMO
entity	prize	award, ...
entity	laureate	cognitiveAgent
entity	person	human
entity	organization	group
entity	prize-area	fieldOfStudy
event	nobel-prize-winning	unilateralGetting
event	nobel-prize-nomination	declaring, deciding

Fig. 3. Mappings between domain and SUMO concepts.

technologies, products, patents, resources, projects, and organisations in this area. The service is free and is provided by the German Research Center for Artificial Intelligence (DFKI) to the R&D community, potential users of language technologies, students and other interested parties.

Most of the concepts referred to in LT World have a direct counterpart in the underlying ontology (Uszkoreit et al. 2003). For example, people actively working in Language Technology are modelled as instances of the class/concept `Active_Person`. `Active_Person` is a subclass of `Players_and_Teams` which has further subclasses such as `Projects` or `Organisations`. The fact that people coordinate projects is represented by the property/role `hasCoordinated` which maps from `People` $\cup$ `Organisations` (domain) to `Projects` (range). The complex domain `People` $\cup$ `Organisations` of this property indicates that projects might be coordinated not only by `People`, but also by instances of `Organisations`.

The original ontology behind LT World made use of RDF and RDF Schema (Klyne and Carroll 2004; Brickley and Guha 2004). A newer version of the ontology has recently been ported to the Web ontology language OWL (Bechhofer et al. 2004), the new emerging language for the Semantic Web that originates from the DAML+OIL standardisation. OWL still makes use of constructs from RDF and RDFS such as `rdf:resource`, `rdfs:subClassOf`, or `rdfs:domain`, but its two important variants OWL Lite and OWL DL restrict the expressive power of RDFS, thereby ensuring decidability. What makes OWL unique (as compared to RDFS) is the fact that it can describe resources in more detail and that it comes with a well-defined model-theoretical semantics, inherited from description logic (Baader et al. 2003). We have already seen such a new language construct (not available in RDFS): the union operator above, which OWL calls `unionOf`. From description logic, OWL inherits further modelling constructs, such as `intersectionOf`, `equivalentClass`, or cardinality restrictions. The description logic background furthermore provides automated reasoning support such as consistency checking of the TBox and the ABox, subsumption checking, etc. [5] Even though the least expressive

[5] TBox and ABox are terms, introduced in the early days of description logic (or ter-

variant of OWL, viz., OWL Lite has an ExpTime worst-case complexity, optimised implementations based on tableaux algorithms are known (Horrocks et al. 2000), which actually work well for most practical cases and have been implemented in a few systems (see Section 2.3.3).

Our interest in OWL (as compared to RDF/RDFS) is based on the following three practical observations: (i) The standardisation and the propagation of OWL by W3C and the replacement of older frameworks such as (the American) DAML or (European) OIL initiatives forces projects (such LT World) that build up new ontologies to use this new emerging standard. (ii) The modelling primitives in OWL help us to better formulate the intended meaning of the modelled concepts and properties in LT World.[6] (iii) The well-defined semantics of OWL, which is reflected in implemented systems, helps us to reveal modelling errors which could not be observed otherwise. For instance in our original (RDFS-based) LT World ontology, domain and range restriction of properties have not been respected in some instances. These problems could be detected later by consistency checks on the ABox.

2.3.3 Inference services

The new LT World ontology was developed using the OWL plugin of the Protégé knowledge base editor (Horridge 2004; Knublauch et al. 2004). This version of Protégé comes with a partial OWL Lite support by means of the Jena Semantic Web framework (Reynolds 2004). But even though Jena provides incomplete OWL Lite support, a large number of inconsistencies were detected and could thus be eliminated.

The latest version of LT World consists of more than 600 concepts, 200 properties, and 17,000 instances. From an RDF point of view, we have more than 400,000 unique triples. It was confirmed by several tests that querying the ontology through Jena (using RDQL) takes too much time, the main reason being that the OWL reasoner uses the rule engines in Jena for all kinds of

minological logic; Brachman and Schmolze (1985)). TBox refers to the terminological knowledge—knowledge about concepts that are relevant to our domain; for example, that `Active_Person` is a subconcept of `Players_and_Teams`. In that sense, a TBox defines a domain schema. An ABox, however, represents assertions about individuals (of certain concepts), for instance, that *Wolfgang Wahlster* `paticipatesIn` the *SmartWeb* project. Nowadays, the term ontology usually refers to both the TBox and the ABox.

[6] For instance, RDFS provides no means for saying that a concept is exactly the intersection of other concepts. OWL introduces such a concept-forming operator: `owl:intersectionOf`, whereas the `rdfs:subClassOf` construct in RDFS is only a poor approximation. Reasoning within ontologies which miss this and other constructs can lead to quite different results, for instance, during instance retrieval.

inference.

We then looked for implemented description logic systems which view OWL as a mere syntactic variant of the $\mathcal{ALC}$ language family (Horrocks et al. 2000). The FaCT system (Horrocks 1998) seemed to be a good candidate but does not provide much ABox support, which is vital for us (17,000 instances) and other Semantic Web applications. For a long time, we then used the Racer system (Haarslev and Möller 2000). With the help of Racer, many further modelling errors in LT World were uncovered which fell through the "grid" of Protégé/Jena.

During ontology development, the number of instances grew and the complexity of instance descriptions increased. It turned out, though, that the ABox does not really scale up well and that 5,000 instances is the maximum number of instances that Racer can handle for queries to LT World. TBox reasoning (as is the case for the FaCT system) is fine, though.
As other people noticed before, we observed that description logic systems can handle complex TBoxes with several hundreds to thousands of concepts, but break down when it comes to ten thousands or even millions of instances—which will be the case when the vision of the Semantic Web should become a reality. We therefore moved to RDF database systems (see Guo et al. 2004, Haase et al. 2004). The basic idea is that even though we are developing OWL ontologies (LT World) with Protégé, the information that is stored on disk is still RDF on the syntactic level. We are thus interested in RDF DB systems which make sense of the semantics of OWL and RDFS constructs such as `rdfs:subClassOf` or `owl:equivalentClass`. To see why the interpretation of such constructs is of main importance, imagine we want to know the subclasses of class `c1`. From

$$\{\langle \texttt{c2 rdfs:subClassOf c1}\rangle, \langle \texttt{c3 rdfs:subClassOf c2}\rangle\}$$

a pure RDF database system can only return {`c2`}, but by knowing that `rdfs:subClassOf` is a transitive and reflexive relation, we get the right result, viz., {`c1`, `c2`, `c3`}. Similar considerations hold for `equivalentClass`.

We currently solved the scalability problem by porting the ontology to Sesame (http://www.openrdf.org/), an open-source middleware framework for storing and retrieving RDF data (Broekstra et al. 2002). Sesame partially supports the semantics of RDFS and OWL constructs via entailment rules that compute "missing" RDF triples in a forward-chaining style at compile time. Consider, for instance, the `equivalentClass` construct in OWL. From an RDF triple

$$\langle \texttt{Author owl:equivalentClass Writer}\rangle$$

a new triple

⟨Writer owl:equivalentClass Author⟩

is computed. The (slightly simplified) entailment rule which is responsible for this behaviour looks like this:

```
<rule name="owl-equivalentClass-inverse">
  <prem> <subj "c1"/> <pred "equivalentClass"/> <obj "c2"/> </prem>
  <cons> <subj "c2"/> <pred "equivalentClass"/> <obj "c1"/> </cons>
</rule>
```

These predefined rules can be altered and the XML rule file can be extended, according to the users' needs.[7] LT World originally consists of about 200,000 of RDF triples, resulting from the 17,000 instances. The closure computation adds almost the same number of new entailed triples, so that Sesame must handle in the end 404,767 statements. Closure computation is fast and takes only a few seconds of real time on a mid-size Linux machine.

Since sets of RDF statements represent RDF graphs, querying information in an RDF framework means to specify path expressions. Sesame comes with a very powerful query language, SeRQL, which includes (see Aduna B.V. 2004): (i) generalised path expressions, including multi-value nodes and branches, (ii) a restricted form of disjunction through optional matching, (iii) existential quantification over predicates, and (iv) Boolean constraints.
We will see in Section 5.2 that all of the above features, even predicate quantification (which gives us some decidable second-order expressiveness here) are needed to arrive at a SeRQL query which retrieves the right objects in LT World. Speaking in terms of Codd's relational model, we note that SeRQL is not relationally complete, but clearly supports the basic algebraic operations *selection*, *projection*, and *product* (see also Haase et al. 2004).

Sesame has been tested with several 100,000 instances (Guo et al. 2004). Its storage model can be configured by either using an existing database system (PostgreSQL, MySQL, or Oracle) or by going for a pure in-memory representation of the data. We have opted for the latter version in LT World to speed up query time. The system scales up very well, giving satisfactory performance. The memory footprint ranges from 70 to 200 MBytes.

The following sections are devoted to the core of our architecture for domain-restricted QA (see Figure 2). Question processing in the HoG comprises question analysis and interpretation (Section 3), the mapping to domain concepts,

[7] Such a closure computation, as known from expert systems, is guaranteed to terminate, since new URIs are not generated by the construction rules at compile time. Only new triples are computed and there are only finitely-many possibilities here, assuming that the number of predicates stays constant (which is the case). Given u URIs and p predicates, at most $p \times u^2$ triples can be generated.

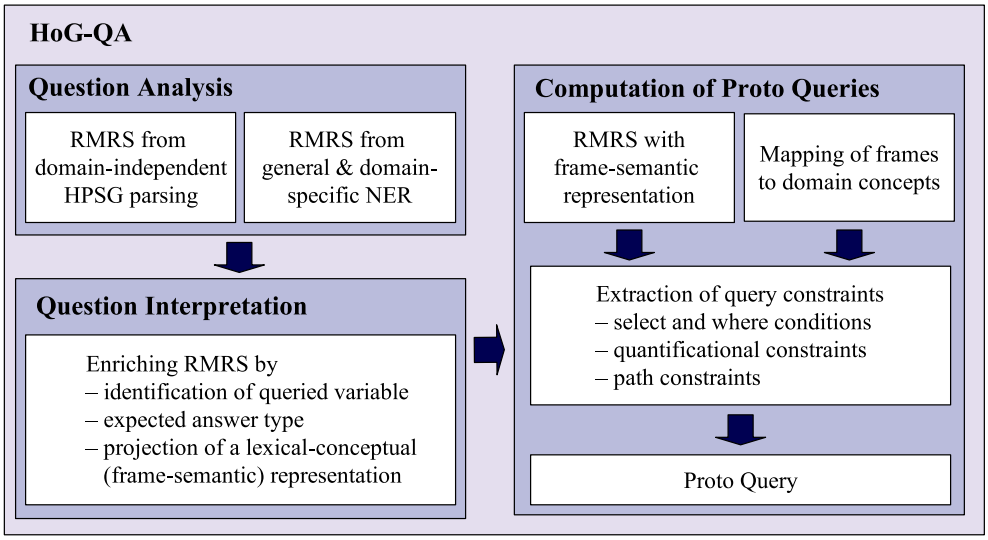

Fig. 4. Internal Architecture of HoG-QA.

and the construction of proto query terms, to be handed over to the answer extraction modules (Section 4). Section 5 explains the details of query construction and answer extraction for our concrete application domains.

3 Question Analysis and Interpretation

Question processing starts with generic syntactic and semantic analysis on the basis of HPSG parsing in a hybrid shallow/deep parsing architecture. This is described in Section 3.1. The parser output is a logical form of the question in the formalism of (Robust) Minimal Recursion Semantics (RMRS). On the basis of the RMRS output structures, a special question interpretation process identifies further semantic information that is relevant for question processing: the queried variable and its associated expected answer type (see Section 3.2). The refined RMRS is further enriched by a lexical-conceptual structure, following the framework of Frame Semantics. On the basis of this enriched representation, we define domain-specific inference rules (see Sections 3.3 and 3.4). In Section 3.5 we discuss how our approach to question interpretation provides a natural account for multilingual and crosslingual QA scenarios. Figure 4 gives a detailed overview of the question interpretation process, including the construction of proto queries, to be discussed in Section 4.

3.1 Hybrid NLP for question analysis

For question analysis we employ deep HPSG syntactic and semantic analysis. HPSG parsing is efficiently performed using the PET parser of (Callmeier,

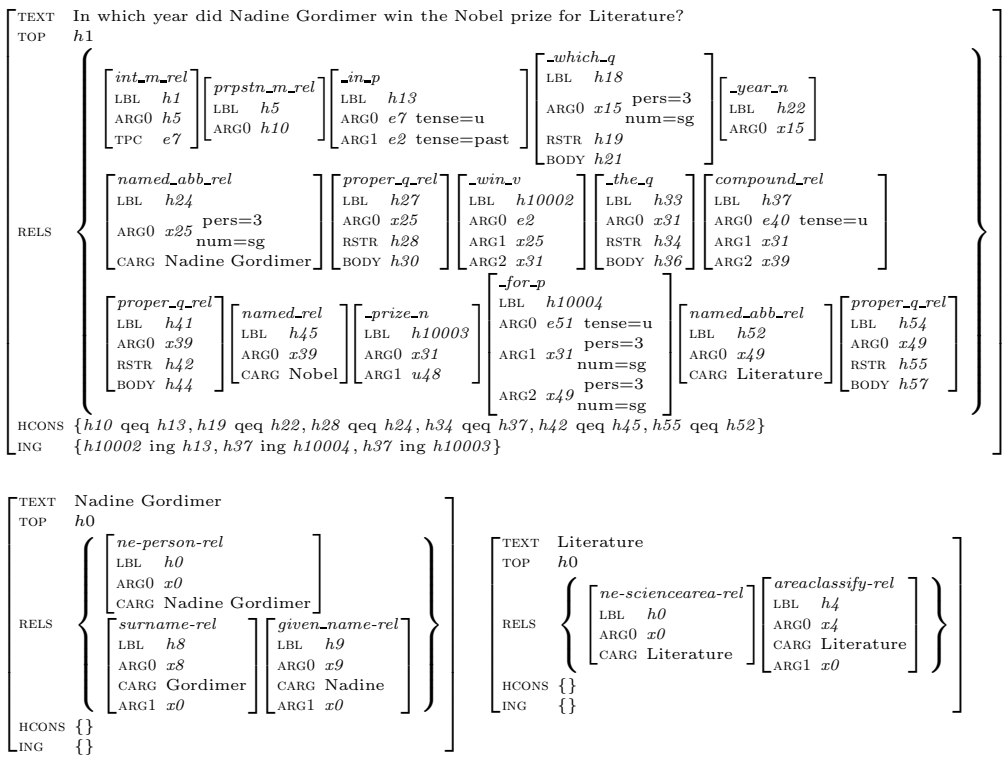

Fig. 5. RMRS of HPSG analysis (top) and SProUT NE recognition (bottom)

2000). For increased robustness, the parser is embedded in an NLP processing platform for integrated shallow and deep analysis, the Heart of Gold (HoG) architecture (Callmeier et al., 2004). Within this architecture, HPSG parsing is seamlessly integrated with the Information Extraction system SProUT (Drozdzynski et al., 2004). SProUT performs named entity recognition (NER) on the basis of unification-based finite-state transduction rules and gazetteers. It provides structured representations both for general named entity classes and domain-specific terms and named entities. The Heart of Gold architecture is designed for integration of NLP components for multiple languages.

In our QA application we are using wide-coverage HPSG grammars for English and German. [8] Both grammars are integrated with shallow NE recognition. HPSG parsing delivers semantic representations in the formalism of Minimal Recursion Semantics (MRS) (Copestake et al., 2005). MRS is designed for the underspecification of scope ambiguities and uses a flat, non-recursive semantic representation format. The basic units of semantic formulas are so-called elementary predications (EPs). As depicted by the example in Figure 5, EPs consist minimally of a relation; for example, a relation *_win_v*, associated with a distinguished label (LBL), and an individual (x) or event (e) variable (ARG0). Argument relations are represented by (abstract) role features (ARG1–ARGN),

[8] The open-source English Resource Grammar (Copestake and Flickinger, 2000; Baldwin et al., 2004) and the German HPSG grammar developed at DFKI (Müller and Kasper, 2000; Crysmann et al., 2002; Crysmann, 2003, 2005).

numbered according to their relative obliqueness. Basic predicate-argument structure is then expressed by coindexation of the argument's inherent variable (or, in the case of propositional or scopal arguments, its label) with the appropriate role feature of the predicate; for example, in Figure 5, the ARG1 of *_win_v* is identified with the individual variable (*x25*) introduced by the proper name "*Nadine Gordimer*". Scope ambiguities are represented compactly by means of subordination (qeq) constraints in the HCONS list. For example, quantifiers (such as *_which_q* in Figure 5) specify their restrictor argument RSTR to outscope their head noun via a qeq relation (here, *h19* qeq *h22*, with *h22* the label of the noun relation *_year_n*), but remain underspecified with respect to their scope argument BODY (here, *h21*).

Figure 5 can, roughly, be read as an interrogative proposition (*int_m_rel*) with a wh-quantified modifier "*in which year*",[9] where the modified event *e2* is a winning relation, its logical subject ARG1 refers to an individual *x25*, with proper name "*Nadine Gordimer*",[10] and whose logical object *x31* is represented as a definite quantified (*_the_q_rel*) compound noun (*compound_rel*) composed of a head noun relation *_prize_n* and a proper name relation ("*Nobel*"). The former is modified by the PP "*for Literature*", where the preposition's ARG1 refers to the variable of the modifié (*x31*), and its ARG2 to the variable for *"Literature"*, which is recognized, by NE recognition, as a proper name in the domain of Nobel prizes. As seen in the bottom structures, NE recognition delivers EPs for the main NE relation types (here, *ne-person-rel* and *ne-sciencearea-rel*), together with more fine-grained information, such as *surname* and *given_name* relations. The latter are represented as modifiers, taking the ARG0 variable of the main relation as value of their ARG1 argument.

Although MRS had mainly been designed to address computational issues, such as tractability and order-independence, there is now a substantial body of theoretical work in HPSG using MRS as meaning representation language (see, for example, Yatabe (2001), Bonami (2002), Kordoni (2003)).

A variant of MRS, Robust Minimal Recursion Semantics (RMRS) has recently been designed in Copestake (2003). The RMRS formalism facilitates the integration of deep semantic analyses with *partial* semantic structures, as produced by shallow NLP components, such as chunkers or robust PCFGs. Within the Heart of Gold architecture, RMRS constitutes the interchange format for all the different NLP components, including NE recognition. Figure 5 displays an example of the RMRS analysis produced by HPSG parsing, along with the RMRS representations provided by named entity recognition. The

[9] See Section 3.2 below for the details of the MRS representation of interrogative sentences.

[10] Note that several relations can be predicated over the same ARG0 variable, as is the case for "*Nadine Gordimer*", represented as a proper name quantifier *proper_q-rel* and a named relation with constant argument value CARG *Nadine Gordimer*.

RMRSs of the SProUT NER component are available as highly structured, IE-like NE representations, decomposing, for instance, a complex person name into *surname* and *given_name* relations. The identified NE classes are further mapped to coarse-grained HPSG NE-types (see *named_abb_rel*), which are directly delivered to the HPSG parser to enhance robustness. Both these highly structured RMRS representations and the coarse-grained HPSG types are automatically generated by XSLT stylesheets that are compiled from the XML output structure specifications of SProUT NE types (see Schäfer (2004) for an earlier approach).

3.2 Question interpretation

The RMRS analysis of questions as delivered by HPSG parsing marks the proposition with the semantic relation *int_m_rel*, for interrogative message type. [11] In wh-questions, interrogative pronouns introduce sortal relations for the queried constituent, such as *person_rel* (who), *thing_rel* (what), *time_rel* (when), etc. For wh-phrases with nominal heads, the semantic relation introduced by the noun constrains the semantic type of the queried constituent (see *_year_n* in Figure 5). Imperative sentences such as *"List all persons who work on Information Extraction."* introduce an imperative message type *imp_m_rel*.

While the RMRS representation of the question encodes important semantic information for question interpretation, such as the message type and the marking of wh-phrases, this representation must be further enriched in order to derive concise queries for answer extraction from structured knowledge sources. This we perform in a special semantic interpretation process that takes as input the RMRSs provided by "general purpose" HPSG parsing. The minimal information we need to identify is the *queried variable (q_var)* in the RMRS for the question, along with sortal information for this queried variable, the *expected answer type (EAT)*. This information is usually employed in textual QA systems, but can also be effectively used for answer extraction from structured knowledge sources, as will be discussed in Section 5 below.

The question interpretation module takes as input the RMRS representations of the question as delivered by hybrid analysis in the Heart of Gold: the RMRS produced by the English or German HPSG parsers, and the RMRSs for recognised named entities. [12] We apply interpretation rules that refer to

[11] In Figure 5, *int_m_rel* (labeled *h1*) embeds, as its ARG0, the generic proposition relation *prpstn_m_rel* (for *proposition_message*) with label *h5*. The proposition labelled *h5*, in turn, embeds the event variable *e2* of the verb relation, with intervening qeq constraint (*h10* qeq *h13*) and temporal modification by the PP *"in which year"*, labeled *h13*, which embeds *e2* as its ARG2 argument.

[12] For the entire question interpretation process depicted in Figure 4 we are currently using the term rewriting system of Crouch (2005). To this end, the input RMRSs

(partial) argument structures in the RMRS in order to identify and mark the queried variable *q_var* of the question. We further determine the ontological type of the queried variable, which provides important semantic constraints for answer extraction. Pronominal wh-phrases introduce a semantic relation for the queried variable, such as *person*, *location*, or *reason*. For these general concepts, as well as for wh-phrases headed by common nouns, we perform a concept lookup, either by selecting an ontological class from SUMO, by way of its WordNet lookup facility, or else by directly mapping the lexeme to its corresponding domain concept. [13] For the example displayed in Figure 5, this yields the additional semantic constraints: *q_var(x15)* and *EAT(x15, 'year')*, with *x15* the variable corresponding to *"year"*. These additional constraints are encoded in the RMRS by way of elementary predications (EPs) *q_focus* and *EAT_rel*, as seen below. In both EPs the value of the ARG0 feature identifies the queried variable. *EAT_rel* in addition encodes the feature SORT, which takes as value the sortal type determined for the queried variable.

$$\begin{bmatrix} \text{REL} & q\_focus \\ \text{ARG0} & x15 \end{bmatrix} \begin{bmatrix} \text{REL} & EAT\_rel \\ \text{ARG0} & x15 \\ \text{SORT} & year \end{bmatrix}$$

The RMRS as logical form of the question now explicitly encodes the queried variable, along with ontological restrictions as additional sortal constraints. The remaining EPs in the RMRS define relational constraints on the requested information. In our example, we are looking for the time when a Nobel prize was won, by a person named "*Nadine Gordimer*", where the area was "*Literature*". These are the key relational constraints that need to be satisfied when retrieving the answer from the underlying knowledge base.

It is the task of question interpretation to identify these relational constraints on the basis of the semantic representation of the question. These constraints can then be translated to a search query in the formal query language of the underlying knowledge base. We perform this task in three steps: We first enrich the RMRS with a frame-based lexical-conceptual representation (Section 3.3). On the basis of a pre-defined set of domain-relevant frames and roles we extract from this enriched representation relational constraints for query construction. These relational constraints, defined in a so-called *proto query*, can then be translated to a search query with corresponding domain-specific concepts and properties, to retrieve the requested information from the knowledge base.

The motivation for this approach is two-fold: First, the projection of a frame-based lexical-conceptual structure yields a normalised semantic representation that naturally accounts for linguistic variants, or paraphrases of questions.

---

are translated to a corresponding term representation.

[13] In our current prototype system concept lookup is encoded manually. In future work we will experiment with automated methods for concept lookup using WordNet-based word sense disambiguation (Burchardt et al., 2005a,b).

It further constitutes a natural approach for multi-lingual and cross-lingual question answering in restricted domains. Second, by defining a set of domain-relevant frames and roles we can establish a modular interface between the linguistically determined lexical-conceptual representation of the question and the concepts of the underlying knowledge bases. On the basis of a mapping between domain-relevant frames and corresponding concepts in the domain ontologies, we can efficiently identify and extract the domain-relevant constraints from the semantic representation of the question. These constraints are encoded in a proto query that is handed over to the answer extraction process. The use of abstract proto queries gives us a clean interface that abstracts away from the syntax and functionality of the backend query languages.

3.3 Projection of a frame-semantic representation

We enrich the RMRS with a lexical-conceptual projection, following the theory of Frame Semantics, as pursued in the FrameNet project (Baker et al., 1998). FrameNet is building a lexical database of frame-semantic descriptions for English verbs, nouns, and adjectives. A *frame* models a conceptual situation with a number of concept-specific roles that identify the participants in the situation. Each frame lists a number of *lexical units* that can evoke the corresponding frame. In addition, FrameNet defines peripheral or extra-thematic roles, such as MANNER or TIME. An example is given in (1).

(1) A business man making an investment would know that [he $_{RECIPIENT}$] would $get_{GETTING}$ [his money $_{THEME}$] back [within four years $_{TIME}$].

An important motivation for using a frame-semantic projection is that—due to their design as lexical-conceptual semantic structures—frames account very naturally for the normalisation of paraphrases. For illustration, consider the semantically equivalent paraphrases in (2.a), which are all very typical expressions for requesting information from a database about Nobel prizes.
HPSG semantic representations in terms of (R)MRS, however, are tailored to account for structural semantic properties such as quantifier scoping and predicate-argument structure, and thus still reflect the various different argument structures involved, as illustrated in (2.b).
Following related work in Frank and Erk (2004), we enrich the RMRS representation with a frame-semantic projection, by mapping the different argument structures of verbs or nouns to their corresponding frame structure, which states the name of the frame and its frame-specific roles. An example of such a frame assignment rule is given in (2.c). [14] (2.d) displays the frame-semantic

[14] We abstract here from the concrete syntax of the rewrite system we are using.

representation obtained for the partial RMRS variants in (2.b).

(2) a. *(win / be awarded / obtain / get / be winner of) a prize*

b. Different argument structures in RMRS representation

$$\left(\begin{bmatrix} \text{REL} & \textit{win/get/obtain} \\ \text{ARG0} & \textit{e1} \\ \text{ARG1} & \textit{x1} \\ \text{ARG2} & \textit{x2} \end{bmatrix} \vee \begin{bmatrix} \text{REL} & \textit{award} \\ \text{ARG0} & \textit{e1} \\ \text{ARG1} & \textit{u1} \\ \text{ARG2} & \textit{x2} \\ \text{ARG3} & \textit{x1} \end{bmatrix} \vee \begin{bmatrix} \text{REL} & \textit{winner} \\ \text{ARG0} & \textit{x1} \\ \text{ARG1} & \textit{x2} \end{bmatrix}\right) \begin{bmatrix} \text{REL} & \textit{prize} \\ \text{ARG0} & \textit{x2} \end{bmatrix}$$

c. RMRS-based frame assignment rule

$$\begin{bmatrix} \text{REL} & \textit{win} \\ \text{ARG0} & \textit{e1} \\ \text{ARG1} & \textit{x1} \\ \text{ARG2} & \textit{x2} \end{bmatrix} \begin{bmatrix} \text{REL} & \textit{prize} \\ \text{ARG0} & \textit{x2} \end{bmatrix} \Rightarrow \begin{bmatrix} \text{GETTING} & \textit{e1} \\ \text{SOURCE} & \textit{u1} \\ \text{THEME} & \textit{x2} \\ \text{RECIPIENT} & \textit{x1} \end{bmatrix} \begin{bmatrix} \text{AWARD} & \textit{x2} \\ \text{LAUREATE} & \textit{x1} \\ \text{DOMAIN} & \textit{u3} \end{bmatrix}$$

d. Conceptual (frame-semantic) representation

$$\begin{bmatrix} \text{GETTING} & \textit{e1} \\ \text{SOURCE} & \textit{u1} \\ \text{THEME} & \textit{x2} \\ \text{RECIPIENT} & \textit{x1} \end{bmatrix} \begin{bmatrix} \text{AWARD} & \textit{x2} \\ \text{LAUREATE} & \textit{x1} \\ \text{DOMAIN} & \textit{u3} \end{bmatrix}$$

3.4 Inferences

The frame-semantic representations can be further enriched by applying simple forward-chaining inference rules.

Frames define a number of *core* frame elements, which can be understood to be existentially quantified even in cases where the role is not overtly realised. Thus, we can introduce non-instantiated argument variables for unexpressed frame elements (see SOURCE in the GETTING frame in (2.d)).

We further define domain-specific inference rules that can be derived from inherent semantic relations between frames. The rule in (3), for example, defines that whenever there is an AWARD frame where the role LAUREATE refers to some variable in the logical form, this variable in turn projects a frame LAUREATE, with its own specific core semantic roles, such as NAME, etc. By application of rule (3), we extend the frame representation in (2.d) with an additional frame LAUREATE, bound to the variable x1. Inferences of this type turn out to be very effective to obtain maximally connected representations in the frame semantic projection. [15]

[15] Tightly connected frame representations in turn are useful for the extraction of concept-relating paths in the query construction phase (see Sections 4 and 5 below).

(3) $\begin{bmatrix}\text{AWARD} & x2\\ \text{LAUREATE} & x1\end{bmatrix} \Rightarrow \begin{bmatrix}\text{LAUREATE} & x1\\ \text{NAME} & u5\end{bmatrix}$

In addition to such forward-chaining inferences we define a number of inference rules that are crucial to bridge mismatches between the representation that is generated on the basis of the linguistic analysis and the structure of the underlying knowledge base, that is, the concepts or relations in the underlying ontology or database. In example (4), the linguistic analysis of the question determines a frame-semantic structure where the temporal modifier of the winning event is mapped to the TIME role of the GETTING frame. The underlying domain ontology, however, does not encode a concept that corresponds to the GETTING frame. Instead, this temporal information is encoded as a property of the award. Mismatches of this type can be accounted for by inference rules, as illustrated in (4.c). The rule states that if there is a GETTING frame where the THEME is an AWARD, and its TIME role refers to some temporal variable, the AWARD frame inherits the value of this TIME role. This corresponds to an inference according to which the time of receiving an award is equal to the time (attribute) of the award.

(4) a. *When did Marie Curie win the Nobel prize for Physics?*

b. Partial RMRS and frame-semantic projection

$\begin{bmatrix}\text{REL} & \textit{sciencearea}\\ \text{ARG0} & \textit{x3}\\ \text{CARG} & \textit{physics}\end{bmatrix}\begin{bmatrix}\text{REL} & \textit{person}\\ \text{ARG0} & \textit{x1}\\ \text{CARG} & \textit{Marie Curie}\end{bmatrix}$

$\begin{bmatrix}\text{REL} & \textit{q\_focus}\\ \text{ARG0} & \textit{t1}\end{bmatrix}\begin{bmatrix}\text{REL} & \textit{eat\_rel}\\ \text{ARG0} & \textit{t1}\\ \text{SORT} & \textit{time}\end{bmatrix}$

$\begin{bmatrix}\text{GETTING} & \textit{e1}\\ \text{SOURCE} & \textit{u1}\\ \text{THEME} & \textit{x2}\\ \text{RECIPIENT} & \textit{x1}\\ \text{TIME} & \textit{t1}\end{bmatrix}\begin{bmatrix}\text{AWARD} & \textit{x2}\\ \text{LAUREATE} & \textit{x1}\\ \text{DOMAIN} & \textit{x3}\\ \text{TIME} & \textit{u2}\end{bmatrix}\begin{bmatrix}\text{LAUREATE} & \textit{x1}\\ \text{NAME} & \textit{Marie Curie}\\ \text{AFFILIATION} & \textit{u6}\end{bmatrix}$

c. Inference rule

$\begin{bmatrix}\text{GETTING} & \textit{e1}\\ \text{THEME} & \textit{x2}\\ \text{TIME} & \textit{t1}\end{bmatrix}\begin{bmatrix}\text{AWARD} & \textit{x2}\\ \text{TIME} & \textit{u2}\end{bmatrix} \Rightarrow \begin{bmatrix}\text{AWARD} & \textit{x2}\\ \text{TIME} & \textit{t1}\end{bmatrix}$

Inferences of this type allow us to map linguistically determined frame-semantic represention to the structure of the underlying domain ontology, and thus, to extract appropriate query constraints for the answer extraction process. This will be discussed in more detail in Section 4.

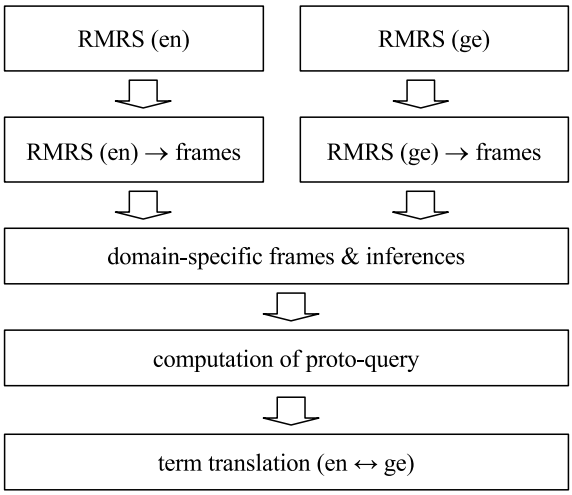

Fig. 6. Multilingual QA: frame-semantic representations as interlingua.

3.5 Multilinguality

Our approach to question interpretation naturally extends to multilingual and crosslingual QA scenarios. Since frames are defined as lexical-conceptual structures, they are to a large extent language independent. [16] Thus, question interpretation in terms of a frame-semantic representation effectively implements a kind of 'interlingua' approach for question answering: the frame-semantic representations serve as a language-independent interface to the underlying knowledge bases. [17]

As illustrated in Figure 6, HPSG grammars for different languages—in our scenario, English and German—provide semantic structures in a uniform formalism, (R)MRS. The language specific relations in these semantic forms are translated by language- and lexeme-specific frame projection rules to a common, language-independent frame-semantic representation.

The remaining parts of the question interpretation and answer extraction processes are then uniform across languages. Domain-specific inference rules refer to the common frame-semantic representations, thus they are applied to the same type of intermediate structures in question interpretation, irrespective of whether they were produced by German or English HPSG grammars. Simi-

[16] This is currently explored by international activities around FrameNet (Erk et al., 2003; Subirats and Sato, 2004).

[17] We don't present any claims to the effect that frames could be used as an interlingua in MT. Our architecture differs from interlingua approaches as pursued, for example, in Rosetta (Rosetta, 1994), which assumes a close correspondence between source and target language grammars, in that the translation relation needs to observe a *principle of compositionality*. In our approach, frame representations serve as a coarse-grained language-independent conceptual representation that is constructed independently by language-specific rules for the individual languages.

larly, the subsequent rules for the extraction of proto queries uniformly operate on the language-independent frame-semantic representations (see Section 4). For crosslingual QA from structured knowledge sources, we perform term translation for instances (named entities) and domain-specific terms of the knowledge base that can appear as values in search queries constructed from the question's representation.
As expected, we experienced very moderate development effort when porting the question processing module from English to German. [18]

4 Interfacing Question Interpretation and Domain Ontologies

The process of question interpretation yields a lexical-conceptual frame structure on top of the RMRS obtained from HPSG parsing. On the basis of predefined mappings from domain-relevant frames and roles to concepts in the underlying knowledge bases, we extract query constraints from the frame-semantic representation of the question, which are composed to so-called *proto queries*—abstract query patterns that can be translated to concrete ontology or database query languages of the underlying domain knowledge bases. This is discussed in Sections 4.1 and 4.2. Section 4.3 explains the construction of proto queries for more complex questions involving quantification. Finally, we discuss the extraction of path constraints that can connect otherwise unrelated frames in individual query constraints of a proto query.

4.1 Mapping frames to domain concepts and properties

As a basis for the extraction of queries to the domain knowledge bases we define a set of *domain-relevant* frames and roles, for which the domain models specify corresponding concepts and properties. This is done by simple clauses, as illustrated below for the frame AWARD from the Nobel prize domain. [19]

[18] The main body of frame projection rules could be carried over from English to German by adapting the language-specific predicate names, while differences in argument structure required the definition of new projection rules. According to the architecture as displayed in Figure 6, German and English frame projection rules are defined as interchangeable sub-modules. In the same way, the sets of rules that define domain-specific inferences and the mapping of frames to domain concepts (see Sections 4.1 and 4.2) are defined as interchangeable sub-modules for the different application scenarios, yet independently from the language parameter.
[19] In the Prolog-like notation of the rewrite system of Crouch (2005), arguments starting with underscore are anonymous variables, those beginning with uppercase letters introduce named variables, and lowercased arguments refer to constants. The clauses state facts that can be matched by subsequent rewrite rules, as in (5) below.

```
frame_role2domain(award,laureate,_,_).
frame_role2domain(award,domain,_,_).
frame_role2domain(award,time,_,_).
```

Besides the *identification* of domain-relevant frames, we can further specify a *mapping* to corresponding concepts and properties in the underlying ontology. This option we pursued in the LT-World scenario, where the clauses in addition state the target concepts and properties in the LT-World ontology.

```
frame_role2domain(project,leader,'ActiveProject',coordinatedBy).
frame_role2domain(project,name,'ActiveProject',projectNameVariant).
```

On the basis of this information we extract domain-relevant concepts from the semantic representation of the question, and turn them into abstract query terms that are then translated to concrete database or ontology queries. [20]

4.2 Construction of proto queries

A basic distinction for the construction of structured query terms is the distinction between queried vs. constraining concepts and attributes. For the extraction of *queried concepts* in (5.a), we select those domain-relevant frames and roles that correspond to the queried variable (*q_var*) in the logical form, represented as the ARG0 argument of the *q_focus* relation (see Section 3.2). We further extract the corresponding ontological restrictions encoded as the expected answer type in *EAT_rel*. In (5.b) we extract all remaining (non-queried) domain-relevant frames and roles, which provide additional constraints on the queried concepts. Again, we extract ontological restrictions, here in terms of their named entity type, as encoded by the RMRS structures provided by NE recognition in the HoG. Subsequent rules further identify the value of the constraint, in general the main predicate (relation) or CARG (constant name) associated with the role's variable, such as 'Marie Curie' in (5.b), or time constants for temporal constraints.

(5) a.
```
    q_focus(Qid,Y),
    frame(Frame,X), role(Role,X,Y), EAT_rel(Y,Sort),
    frame_role2domain(Frame,Role,_,_)
=> select_cond(Qid,Frame,Role,Sort).
```

[20] We already mentioned, in Section 3.5, footnote 17, that the rules for identification and mapping of frames to domain-specific concepts are defined as interchangeable sub-modules, which are independent of the language parameter and the general rule set for proto query construction.

b.
```
-q_focus(Qid,Y),
frame(Frame,X), role(Role,X,Y), ne_type(Y,NE),
frame_role2domain(Frame,Role,_,_)
=> where_cond(Qid,Frame,Role,NE).
```

By this method we extract so-called *proto queries* from the frame-semantic structures, as illustrated below. (6.c) shows the instantiated query terms that are extracted by rules like (5). The values of `Frame` and `Role` in the `select_cond` and `where_cond` terms are recorded as values of the attributes `rel` and `attr` of the proto query (6.d). [21]

(6) a. *In which areas did Marie Curie win a Nobel prize?*

b. Question interpretation

$$\begin{bmatrix}\text{REL} & \textit{q\_focus}\\ \text{ARG0} & \textit{x10}\end{bmatrix}\begin{bmatrix}\text{REL} & \textit{EAT\_rel}\\ \text{ARG0} & \textit{x10}\\ \text{SORT} & \textit{FieldofStudy}\end{bmatrix}\begin{bmatrix}\text{REL} & \textit{person}\\ \text{ARG0} & \textit{x17}\\ \text{CARG} & \textit{Marie Curie}\end{bmatrix}$$

$$\begin{bmatrix}\text{GETTING} & \textit{e2}\\ \text{THEME} & \textit{x21}\\ \text{RECIPIENT} & \textit{x17}\end{bmatrix}\begin{bmatrix}\text{AWARD} & \textit{x21}\\ \text{LAUREATE} & \textit{x17}\\ \text{DOMAIN} & \textit{x10}\end{bmatrix}\begin{bmatrix}\text{LAUREATE} & \textit{x17}\end{bmatrix}$$

c. Extracted `select_cond` and `where_cond` terms:

```
select_cond(0,award,domain,'FieldofStudy')
where_cond(0,award,laureate,person,'Marie Curie')
```

d. Proto Query

```
<PROTO-QUERY id="1">
  <SELECT-COND qid="0" rel="award" attr="domain" sort="FieldofStudy">
  <WHERE-COND qid="0" rel="award" attr="laureate" netype="person" val="Marie Curie">
</PROTO-QUERY>
```

4.3 Quantificational questions

QA from structured knowledge bases is particularly well suited to answer questions for which the answer is not explicitly represented in the document or knowledge base, but must be inferred from the available information. Prime examples are cardinality, quantificational or comparative questions, as in (7).

(7) a. *How many researchers won a Nobel prize for Physics before 1911?*
b. *Which institution has published most papers between 2000 and 2004?*
c. *Which nation has won more Nobel prizes in Physics than the U.S.?*

[21] Proto queries may be complex, that is, may be decomposed into individual sub-queries with specially marked dependencies. Therefore, all conditions that pertain to a single sub-query are marked by a common sub-query index (`qid`). The main query index is associated with the questions's q_focus, as seen in (5). The rules that process complex quantificational queries, as in Figure 7, introduce new `qid` indices for the generated sub-queries.

```
<PROTO-QUERY id="1">
  <SELECT-COND qid="0" rel="award" attr="laureate" sort=""/>
  <WHERE-COND qid="0" rel="award" attr="time" netype="" valfunc="before" valarg="1911"/>
  <WHERE-COND qid="0" rel="award" attr="domain" netype="sciencearea" val="Physics"/>
  <OP-COND oprel="card" domain-type="answer" domain-id="0"/>
</PROTO-QUERY>
```

Fig. 7. P-Q for *How many researchers won a Nobel prize for Physics before 1911?*

```
<PROTO-QUERY id="8">
  <SELECT-COND qid="0" rel="laureate" attr="origin" sort="?"/>
  <QUANT-COND qid="1" quantrel="foreach" domain-type="answer" domain-id="0"/>
  <SELECT-COND qid="1" rel="award" attr="" sort=""/>
  <WHERE-COND qid="1" rel="laureate" attr="origin" valfunc="answer_of" valarg="0"/>
  <WHERE-COND qid="1" rel="award" attr="domain" val="Physics"/>
  <OP-COND oprel="max_card" domain-type="answer" domain-id="1" />
</PROTO-QUERY>
```

Fig. 8. P-Q for *Which nation has won most Nobel prizes for Physics?*

To account for these quantificational aspects, we employ special proto query conditions `OP-COND` and `QUANT-COND`. These constructs go beyond the formal power of database query languages like SeRQL, but can be translated to special post-processing operations in the answer extraction phase.

The quantificational conditions are strongly determined by the semantic representation of the question. It is for this reason that the computation of proto queries is tightly integrated with question interpretation (see Figure 4). Cardinality questions are marked by operators such as *how many* that range over the queried variable. For such configurations we generate a condition `OP-COND` which specifies the operator relation `op-rel` that corresponds to the semantics of the quantifier. Since the quantification ranges over the queried variable, the domain of computation is defined as the answer for the sub-query for the queried variable (`domain-id="0"` in Figure 7).

In quantificational and comparative questions of the types illustrated in (7.b,c) the quantification ranges over a non-queried variable. In these cases we perform query decomposition. We compute conditions for a base query that retrieves instances for the domain of quantification (for example, *"nation"* in Figure 8). The quantifier condition `QUANT-COND` defines that for each instance in this domain we perform a sub-query for the queried variable and the non-queried relational constraints (select- and where-conditions), by referring to each instance of the quantifier domain. An operator condition encodes the quantifier-specific relation (for example, `max-card` for *most*) that is to be computed over the retrieved data records.

```
<PROTO-QUERY id="1">
  <SELECT-COND qid="0" rel="award" attr="time" sort=""/>
  <WHERE-COND qid="0" rel="laureate" attr="name" netype="person" val="Marie Curie"
                      path="[award,laureate,laureate]"/>
  <WHERE-COND qid="0" rel="award" attr="domain" netype="sciencearea" val="Physics"/>
</PROTO-QUERY>
```

Fig. 9. P-Q for *In which year did Marie Curie win the Nobel prize for Physics?*

4.4 Extraction of concept-relating paths

The rules for the extraction of proto queries (5.a,b) only consider local frames and roles to define relational constraints for proto queries. Thus, the concepts that appear in the individual `SELECT`- and `WHERE`-conditions may be unconnected. An example is given in Figure 9. Here, the `WHERE` and `SELECT` condition are about the frames `laureate` and `award`, yet it is not clear whether or how these relate to each other. [22]

The frame-semantic representation of the question does, however, often specify connecting paths between these frames and roles. In part, these connections are determined by the linguistic structure, in part by domain-specific inferences (see Section 3.4). In the frame-semantic representation for the example in Figure 9, the frames `laureate` and `award` are connected due to an inference rule that concludes, from two frames `award` and `laureate` that participate as role fillers in a single `getting` frame, that the value of an unfilled role `laureate` of the obtained `award` is identical to the `laureate`, i.e. the `recipient` of the `getting` frame:

$$(8)\ \begin{bmatrix}\text{GETTING} & e1\\ \text{THEME} & x1\\ \text{RECIPIENT} & x2\end{bmatrix}\begin{bmatrix}\text{AWARD} & x1\\ \text{LAUREATE} & u1\end{bmatrix}\Big[\text{LAUREATE}\quad x2\Big] \Rightarrow u1 = x2$$

Thus, for all pairs of `SELECT` and `WHERE` conditions of a proto query that refer to distinct frames, we extract, from the question's frame-semantic representation, the connecting paths that lead from the frame and role of the `WHERE` condition to the frame and role of the `SELECT` condition, and record these connecting paths as a PATH attribute in the query object conditions, as seen in Figure 9. This path information is used in the answer extraction phase to further specify the connections between the individual partial search constraints (see Section 5.2).

[22] The frame/role pairs `laureate/name` and `laureate/domain` are triggered by the verb *win* and its ARG1 argument, and the compound *Nobel prize* and its modifier PP *for Physics*, respectively. The frame/role pair `award/time` is obtained from the temporal verb modifier and the inference rule (4.c) described in Section 3.4.

5 Answer Extraction

5.1 Answer extraction from Nobel prize knowledge resources

The instances of domain relations are stored in the relational database MySQL. We store the Nobel prize winners in two separate tables: one for persons and one for organisations, since the two concepts *person* and *organization* are associated with different attributes. In the following examples, we call these *nobel-prize-winner-person* and *nobel-prize-winner-organization*.

The first step to be taken in answer extraction is to translate proto queries provided by question interpretation to SQL queries. Proto queries as introduced in Section 4 identify: (i) the answer type concept, which corresponds to the value of the SQL `SELECT` command, (ii) additional concepts and their values, which constrain the answer type value (these concepts will fill the SQL `WHERE` conditions), and (iii) dependencies between elementary questions, if a question is complex and needs to be decomposed into several simple questions.

For example, for the simple fact-based question *"Who won the Nobel Prize in Chemistry in 2000?"* question analysis returns the following proto query:

```
<PROTO-QUERY id="q13" type="sql">
  <SELECT-COND qid="0" rel="award" attr="laureate" />
  <WHERE-COND qid="0" rel="award" attr="domain" val="Chemistry"/>
  <WHERE-COND qid="0" rel="award" attr="time" val="2000"/>
</PROTO-QUERY>
```

The task of SQL query translation is to first identify the tables where the requested concepts can be found, and second, the relevant table fields which can match the values given in the proto query. We have defined mapping rules between FrameNet frames and their roles and their corresponding database tables and their fields. In a special field `event-dependent` we further mark concepts that are events. Below we list some examples of table entries.

Rel	Attr	val-concept	DBTable	DBField	event-dependent
award	laureate	person	nobel-prize-winner-person	name	yes
award	laureate	organization	nobel-prize-winner-organization	name	yes
award	domain	prize-area	nobel-prize-winner-person	area	no
award	domain	prize-area	nobel-prize-winner-organization	area	no
award	time	date time	nobel-prize-winner-person	year	yes
award	time	date time	nobel-prize-winner-organization	year	yes

The `SELECT-COND` in the example above only mentions the frame-semantic `rel` and `attr` attributes `award` and `laureate`. Yet, there is no direct mapping to a table for `laureate`. In such cases we make use of our ontology and

discover that `laureate` corresponds to `cognitiveAgent` which has two subconcepts: `human` and `group`. Their corresponding domain concepts are `person` and `organization`. We thus expand `laureate` to `person` and `organization` and find their corresponding tables. In the same way, we identify the tables for the `WHERE-COND`. In this example, `SELECT-` and `WHERE-COND` require access to the same tables. Thus, we generate the following two SQL queries:

```
SELECT name FROM nobel-prize-winner-person
  WHERE year="2000" AND area="chemistry"
SELECT name FROM nobel-prize-winner-organization
  WHERE year="2000" AND area="chemistry"
```

The final answer is obtained by merging their results. While the example just considered involves concept expansion, we also perform concept disambiguation. This is illustrated by the example *In which year did Nadine Gordimer win the Nobel prize for Literature?*, with the proto query

```
<PROTO-QUERY id="1">
  <SELECT-COND rel="award" attr="time" sort="Year" />
  <WHERE-COND rel="award" attr="domain" netype="prize-area" val="Literature" />
  <WHERE-COND rel="award" attr="laureate" netype="person" val="Nadine Gordimer" />
</PROTO-QUERY>
```

Again, both the `SELECT-COND` and the first `WHERE-COND` identify the two tables `nobel-prize-winner-person` and `nobel-prize-winner-organization`. However, in the second `WHERE-COND`, the linguistic analysis recognises that the entity type of `laureate` is `person`. We can use this information for table disambiguation and choose the table `nobel-prize-winner-person`.
The SQL query for this question then is:

```
SELECT year FROM nobel-prize-winner-person
  WHERE area="Literature" AND name="Nadine Gordimer"
```

Finally, we distinguish queried entities that are independent of individual prize winning events from event-dependent entities. Consider the two questions:

(9) *How many areas are there for the Nobel Prize?*

(10) *How many Nobel Prize winners has France produced?*

In the first case, every area in which a person or organisation has won a Nobel prize is only counted once. For answering the second question, we could also count every person once, even if the person has been awarded two prizes, such as Marie Curie. However, in line with counting tourists visiting Paris or customers of Harrod's, we decided to make the cardinality of recipients event-dependent.

Thus, the answer to the first question will be: *Six areas: Chemistry, Physics,*

Peace, Literature, Medicine, Economics, although all areas occur more than once in award-winning events. We treat *area* as event-independent, generating an SQL query with a `DISTINCT` condition:

```
SELECT DISTINCT area FROM table
```

The answer to the second question will be: *Three winners: Marie Curie (2) and Pierre Curie (1).* Here, the person in the *award* relation is handled as event-dependent. In this case we generate the SQL query

```
SELECT person FROM table WHERE country="France"
```

5.2 Answer extraction from the LT World ontology

In this section, we show how a proto query can be mapped to an expression in the query language SeRQL of Sesame.
Based on the mapping from domain-specific frames and roles in the proto query conditions to domain concepts and properties (see Section 4.1), we first perform a translation of the values of `rel, attr`, and `path` attributes to the corresponding domain concepts and attributes of the LT World ontology. Thus, each relation (value of `rel`) now denotes a concept in the ontology and each attribute (value of `attr`) denotes an OWL property.

In a SeRQL query, instances of a concept are identified by variables in the subject position of an RDF triple. The concept itself is stated in the object position, and subject and object are connected by `rdf:type`—this is exactly the way how instances of a specific concept are represented in the RDF base of Sesame. For example,

```
<SELECT-COND qid=".." rel="Organisations" attr="locatedIn" ... />
```

leads to the introduction of the following RDF triple (`_r` is a fresh variable, `ltw` the LT World namespace):

```
{_r} rdf:type {ltw:Organisations}
```

Since attributes like `locatedIn` refer to properties of a concept, we obtain a further triple:

```
{_r} ltw:locatedIn {_q}
```

The property `locatedIn` connects instances of the main concept `Organisations` via the root variable `_r` with the queried information. The queried information is bound to a new question variable `_q` that will be returned. It is marked by the SELECT clause in a SeRQL query:

(1) for each `SELECT-COND` and `WHERE-COND`
  – each relation denotes a concept
  – each attribute denotes a property
  – each unique relation introduces a new *root* variable
(2) each `SELECT-COND` introduces a new *query* variable
(3) each `WHERE-COND` introduces a new *local* variable
(4) guarantee that the RDF triples form a connected graph
  – if path constraints are specified, link the root variables
  – otherwise, introduce new *property* variables linking the roots
(5) finally apply `OP-COND` to the result table (if needed)

Fig. 10. Principles for transformation of proto queries to SeRQL-Queries.

```
SELECT {_q}
FROM {_r} rdf:type {ltw:Organisations},
     {_r} ltw:locatedIn {_q} ...
```

In Figure 10 we give an overview of the main principles of the transformation from proto queries to SeRQL queries.
In order to illustrate the transformation principles, let us consider the question *"Who is working in the Quetal project?"*, with its (simplified) proto query that contains a `SELECT` and a single `WHERE` condition:

```
<PROTO-QUERY>
  <SELECT-COND rel="Active_Person" attr=""/>
  <WHERE-COND rel="Active_Project" attr="projectName" val="Quetal"/>
</PROTO-QUERY>
```

Given this proto query the following SeRQL query is generated:

```
SELECT DISTINCT _q0
FROM {_r1} rdf:type {ltw:Active_Person},
     {_r2} rdf:type {ltw:Active_Project},
     {_r1} ltw:name {_qo},
     {_r2} ltw:projectName {_l3},
     [ {_r1} _p4 {_r2} ],
     [ {_r2} _p5 {_r1} ]
WHERE (NOT (_p4 = NULL) AND (_p5 = NULL)) AND (_l3 LIKE "Quetal")
```

Query construction comprises three main aspects. First, information that is requested must be encoded by variables following the starting `SELECT` clause. We make use of the keyword `DISTINCT` to rule out duplicate occurrences in case no `OP-COND` condition is specified in the proto query. However, if an operator condition is present, `DISTINCT` should not be added because duplicates must be taken into account for arithmetic operations in quantificational questions (for instance in *"Who led most projects in Information Extraction?"*).

Second, RDF triples are collected in the FROM clause, separated by commas, which implicitly express logical conjunction. A restricted form of disjunction is available at this point due to the optionality operator [ ] which expresses information that need not be matched.
Additional restrictions on variables can be formulated in the WHERE clause, including equality (=), inequality (!=) and string matching (LIKE). These restrictions can be combined using the Boolean connectives AND, OR, and NOT.

Returning to the example above, according to principle (1) of Figure 10, two root variables _r1 and _r2 are introduced and linked to concepts Active_Person and Active_Project via rdf:type. Principle (2) leads to the query variable _q0 which is linked to _r1 via the property name, the default property in case the attribute value in a SELECT-COND is empty (= ""). From (3), we get a variable _l3 which binds the value of attribute val ("Quetal") in the proto query above. The value itself is specified in the WHERE clause of the SeRQL query.

The most interesting aspect in the query construction process is how to account for *partially connected* concepts (remember principle (4); for illustration, see Figure 11). In our example, two relations/concepts _r1 and _r2 are introduced in the proto query. Since we want to retrieve information related to _r1 (via _q0), it is important that _r1 and _r2 are *connected*. Otherwise the information from _r2 (through rdf:type and ltw:projectName) can not be incorporated into the search of the RDF database. Put differently, if we did not account for connecting concepts/properties between _r1 and _r2, the above SeRQL query would simply retrieve all instances of concept Active_Person.

It is the last two clauses of the FROM condition and the first WHERE clause that account for this problem. Since _r1 and _r2 are not connected and no information is given regarding a connecting *property* and the *direction* of such a connecting property (from _r1 to _r2, or vice versa?), we let Sesame "guess" this information. Firstly, in order to guess the property, we use *property* variables in the predicate position of an RDF triple. Secondly, in order to guess the direction, we need some kind of disjunction on the FROM level. Here the optionality operator comes into play. Notice that there may be several different properties, connecting _r1 and _r2, even properties from _r1 to _r2 *and* from _r2 to _r1 at the same time. In order not to rule out both optionality statements, we have to formulate further constraints, specifying that the two property variables should not be NULL at the *same* time.

Looking at this from a graph-theoretical perspective, we are interested in *constraints*, characterising directed *minimum spanning trees* (Garey and Johnson, 1979, p. 130). The nodes in such a tree are exactly the root nodes representing the concepts, and the edges which connect the nodes represent the missing properties. The missing edges are either specified via path expressions, or represented by property variables which need to be instantiated by Sesame

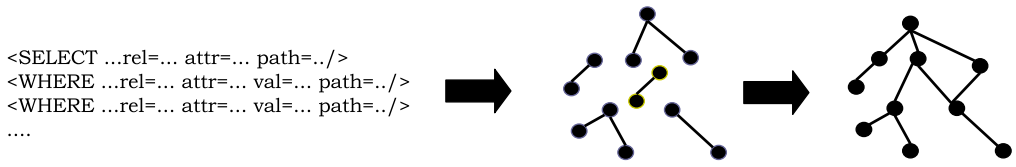

Fig. 11. Given a proto query, a set of partial graphs is determined. In a final step, missing links are added to make the set a connected graph. These links are either added by the proto query transformation process in case path constraints are specified (see Section 4.4) or computed by Sesame through database inspection.

through a DB search. The basic idea is illustrated in Figure 11. Specifying path constraints and adding them as further constraints to the `WHERE` clause of the SeRQL query is clearly superior in terms of run time performance against a pure DB search, where concrete properties are replaced by property variables. The elimination of a single property variable by a concrete property speeds up query performance by more than a factor of 100. Eliminating several property values then results in a multiplication of the single speedup factors. [23]

To see how path constraints can simplify a SeRQL query, consider the following three relations `Active_Person`, `Active_Project`, and `Active_Company`, so that we obtain three root variables overall. Three minimum spanning trees (MSTs) are possible here (actually six, if we take directedness into account), plus, of course, a graph where each node is linked to every other node. In order to characterise the MSTs in a SeRQL query, we introduce two property variables for each pair of root variables. The constraints specified in the `WHERE` clause of the SeRQL query then guarantee that the resulting graphs are at least minimum spanning trees, meaning that they are connected. Considering the three relations above, we see that such a query becomes quite complicated:

```
SELECT DISTINCT _q0
FROM {_r1} rdf:type {ltw:Active_Person},
     {_r2} rdf:type {ltw:Active_Project},
     {_r3} rdf:type {ltw:Active_Company},
     [{_r3} _p6 {_r1}], [{_r1} _p7 {_r3}],
     [{_r3} _p8 {_r2}], [{_r2} _p9 {_r3}],
     [{_r1} _p10 {_r2}], [{_r2} _p11 {_r1}],  ...
WHERE (((NOT (_p10 = NULL AND _p11 = NULL)) AND
        (NOT (_p8 = NULL AND _p9 = NULL))) OR
       ((NOT (_p10 = NULL AND _p11 = NULL)) AND
        (NOT (_p6 = NULL AND _p7 = NULL))) OR
       ((NOT (_p6 = NULL AND _p7 = NULL)) AND
        (NOT (_p8 = NULL AND _p9 = NULL))))  ...
```

[23] A further argument why property guessing can only be the fallback position is that it is in principle possible that there exists more than one property connecting two roots. Guessing would apply all properties, leading to wrong results (too many candidate answers). Fortunately, we have not encountered such a behaviour.

Clearly, the existence of path constraints is extremely important here in order to reduce the DB search space. That is, in cases where we extract concept-relating paths from the frame-semantic representation of the question (see Section 4.4), we can use these path constraints as additional contraints in the SeRQL query. If we assume that at least one `WHERE` condition in the proto query specifies a non-empty path constraint, for example,

```
<WHERE-COND qid="..." rel="Active_Project" attr="..." val="..."
   path="['Active_Person','participatedIn','Active_Project']"/>
```

the complexity of the above SeRQL query is considerably reduced:

```
SELECT DISTINCT _q0
FROM {_r1} rdf:type {ltw:Active_Person},
     {_r2} rdf:type {ltw:Active_Project},
     {_r3} rdf:type {ltw:Active_Company},
     {_r1} ltw:participatedIn {_r2},
     [{_r3} _p6 {_r1}], [{_r1} _p7 {_r3}],
     [{_r3} _p8 {_r2}], [{_r2} _p9 {_r3}],  ...
WHERE (((NOT (_p6 = NULL AND _p7 = NULL)) AND
        (NOT (_p8 = NULL AND _p9 = NULL))))  ...
```

Assuming further that another `WHERE` condition comes with the path constraint

```
path="['Active_Company','hasOrganized','Active_Project']"
```

the SeRQL query collapses to

```
SELECT DISTINCT _q0
FROM {_r1} rdf:type {ltw:Active_Person},
     {_r2} rdf:type {ltw:Active_Project},
     {_r3} rdf:type {ltw:Active_Company},
     {_r1} ltw:participatedIn {_r2},
     {_r3} ltw:hasOrganized {_r2}, ...
WHERE ...
```

Our implementation handles all these cases, along with further optimisations.

5.3 Answer preparation

The answers returned by both MySQL and Sesame are grouped in a result table. Since certain queries require post-processing by arithmetic operations (*"How many people are working at ISI?"*) and since communication between the different components are realised via XML, we do not directly return the result table. Instead, the answers are encoded in a structured *answer object*, similar to the query object that embodies the original proto query (Figure 2).

An answer object refers to the query id of the query object and distinguishes between potential conflicting answers (several `VALUES` tags) and list-based answers (a single result, consisting of several pieces; several `VALUE` tags). Similar to question objects, the answer objects serve as XML exchange structures in the QA architecture. That is, the same type of structure is returned by MySQL for the Nobel prize domain and Sesame in the LT World domain.

As an example, the (simplified) answer object to the question *"Who is working in the Quetal project?"* is

```
<AOBJ id="id18" msg="answer" query-id="Q01" lang="EN">
  <ANSWER type="complex" score="1.0">
    <VALUES> <VALUE>Bogdan Sacaleanu</VALUE>
             <VALUE>Günter Neumann</VALUE>
             ... </VALUES>
  </ANSWER>
</AOBJ>
```

With respect to natural language answer generation, we are currently exploring two alternative approaches: template-bassed shallow generation using TG2 (Busemann, 1996) as well as MRS-based HPSG surface generation. While a template-based system should appear more than sufficient to provide natural language output for restricted domains, it will certainly require considerable duplication of language engineering efforts. It therefore makes perfect sense to exploit the reversability of the German and English HPSG grammars for multilingual answer generation. The required MRS input for the generator can then be partially derived from the MRSs obtained in question analysis, with requested information being filled in from the answer objects.

6 Evaluation

We have performed an initial evaluation of our prototype system for domain-restricted QA from structured knowledge sources. A system-internal evaluation assesses the quality and efficiency of question interpretation and answer extraction. In addition we performed a comparative evaluation of our domain-restricted system to the web-based open-domain textual QA system AnswerBus (Zheng, 2002). This, in conjunction with a detailed classification of question types, allows us to assess the added value of a specialized, domain-restricted QA component in a hybrid system architecture. [24]

[24] The textual QA system of Quetal obtained the best results in the 2004 cross-lingual CLEF task (Neumann and Sacaleanu, 2004). However, it is not yet extended to web-based QA. Since we do not yet have access to appropriately large document bases for our two domains, it seemed most appropriate to choose an independent open-source web-based QA system and to perform the comparative evaluation in

question type	in %	expected answer type	in %
factual	58	time	13
list	15	person, organisation	54
definition	2	currency	3
cardinality (how many)	22	prize area	14
quantificational (most)	24	nation	12
event quantification	2	achievement	1
embedded questions	17		

Fig. 12. Distribution of question types and expected answer types.

The comparative evaluation to AnswerBus restricts us to questions in English; we further chose the Nobel prize domain, as information about this domain is appropriately covered by the WWW. We compiled a set of 100 English questions about the Nobel prize domain, in part adapted or inspired from the FAQ sections of Nobel prize web portals.

6.1 Question classification

The question types in our test set range from factual and list questions to different types of cardinality and quantificational questions. Figure 12 gives a detailed overview of the different question types and their distribution over the sample set, along with a classification of the questions' expected answer types, again with quantitative distribution.[25] The questions are varied in terms of paraphrases (verbal and nominal paraphrases, interrogative, non-interrogatives or embedded questions, such as *"Give me a list of..."*, *"Could you tell me in which year..."*), and according to different types of constraints to be used in question interpretation and answer extraction, such as (relational) temporal constraints (*in/before/since/after 1999*), gender (*female prize winners*), prize areas, as well as countries, locations, and affiliations.

6.2 Question processing and interpretation

For the 100 questions the average runtime (real time) per question was 3.74 seconds for full online processing from text input to answer object output, on a Intel Xeon 2.5 GHz Linux machine. Answer extraction alone took 125 milliseconds per query object on average. For four questions the linguistic analysis failed. The question sample contains 18 questions that instantiate two types of event quantification which are not yet accounted for by the question interpretation module. For the qualitative evaluation we accordingly distinguish between the full question sample as basis for evaluation, displayed in the first

the Nobel prize domain, for which enough information can be found on the Web.
[25] The types are overlapping, so the figures do not sum up to 100%.

	correct pq in n^{th} parse (in %)			nb of correct pq's per q (in %)			overall proto query results (with voting)			
	1st	2nd	3rd	3	2	1	corr.	uncert.	wrong	no pq
full sample	46.0	41.0	32.0	18.0	24.0	17.0	58.0	3.0	15.0	24.0
in-scope only	56.1	50.0	39.0	22.0	29.3	20.7	69.5	2.4	7.3	19.5

Fig. 13. Evaluation figures for question interpretation (pq: proto query).

row of Figure 13, while the figures in the second row are computed on the basis of the 82 questions that can currently be considered as in-scope phenomena.

Our HPSG grammars are equipped with stochastic models for parse selection (Oepen et al., 2002). In the current set-up, the question interpretation module applies to the three highest-ranked semantic HPSG analyses, and delivers a separate question object for each of them. Figure 13 gives an overview of the distribution of correct proto queries over the highest-ranked parses (columns 2-4), as well as the overall number of correct proto queries across the three highest-ranked analyses (columns 5-7). Of the overall set of 100 questions, 46% return the correct proto query for the highest-ranked parse, 41% and 32%, respectively, for the second and third ranks; restricted to the in-scope phenomena, the figures raise to 56.1%, 50% and 39%, respectively.

In many cases, question interpretation extracts a correct proto query from more than one of the three best parses: For 18% of all questions (22% of the in-scope samples) we obtain the correct proto query from all three parses considered; 24% (29.3%) return two correct proto queries; for 17% (20.7%) we obtain a single correct proto query from the three highest-ranked analyses.
Since we are considering the three best parses, we can apply a voting scheme to determine which one of possible alternative proto queries to send to the answer extraction module. In cases of non-conflicting multiple results, voting is not necessary. However, we often obtain proto queries that are partial, or less specific than another proto query result for the same question, which hence could lead to wrong answers. In those cases where the partial query is subsumed by both alternative analyses, or by a single alternative analysis out of two resulting proto queries, we ignore the partial query, in favour of the more specific one. In 67,9% of all cases that involve partial queries (28 on the full corpus), this strategy yields a correct proto query. In cases where a proto query is subsumed by only one of two alternative proto queries, we mark the result as uncertain. This occurs in 3% (2.4%) of cases. For 24% (19.5%) of the questions, all analyses return an empty proto query, and are thus to be regarded as out of system coverage. These cases are either due to problems in the semantic analysis (failed or wrong parses or parse selection), or in the question interpretation process.

	correct	incorrect	no answer
abs. #	43	4	11
in %	74.1	6.9	19.0

Fig. 14. Evaluation figures for answer extraction (based on 58 correct proto queries).

question type	in % (in-scope)	expected answer type	in %
factual	53.4 (64.6)	time	46.2
list	40.0 (42.0)	person, organisation	40.7
definition	100.0 (100.0)	currency	0.0
cardinality (how many)	18.2 (22.0)	prize area	42.9
quantificational (most)	25.0 (54.5)	nation	58.3
event quantification	100.0 (100.0)	achievement	0.0
embedded questions	47.1 (47.1)		

Fig. 15. Distribution of correct answers over question and expected answer types.

As seen in Figure 13, the overall ratio of correct proto queries that result from the voting and filtering process is 58% (69.5%). With 15% (7.3%), we achieve a moderate error rate, opposed to a higher rate of cases where the system signals that it is uncertain (3%/2.4%) or unable to answer the question (24%/19.5%).

Overall, then, the system features relatively high precision that is balanced against a low error rate and reduced recall. This tendency is especially welcome for a domain-restricted QA system that is confronted with high user expectations regarding the reliability of the answers delivered. Another outcome of the evaluation is that a high percentage of the unanswered questions failed because the correct parse was not promoted high enough by stochastic parse selection. This situation will be improved in future work, by retraining the stochastic disambiguation model on typical question samples.

6.3 Answer extraction

We evaluated the answer extraction module on the basis of the 58 correct proto queries that were selected by the voting procedure. Figure 14 presents the results: for 74.1% of the proto queries the correct answer was returned; in 6.9% the answer was wrong; for 19%, finally, no answer was returned. Error analysis for the 4 incorrect answers yielded a single minor cause of error (wrong answer type identification). For missing answers we identified several causes that need to be adjusted: mismatches of concept-database mappings, wrong table selection and out of scope phenomena.
Figure 15 details the distribution of correct answers over different question types, with restriction to in-scope phenomena in parantheses.

	correct for top n results				correct for question types			
	1st	2nd	3rd	overall	fact	card	quant	embedded
in %	9.0	8.0	8.0	15.0	22.4	4.5	4.2	5.9

Fig. 16. Distribution of correct answers (AnswerBus).

6.4 Comparison to AnswerBus

In order to assess the added value of a domain-restricted QA component, we compare the results of our current prototype system to the results delivered by the open-domain textual QA system AnswerBus (Zheng, 2002). We collected the three highest-ranked answers returned by AnswerBus, and evaluated the returned answers (Figure 16). The coverage on our 100 question sample is rather poor: for only 15% of the questions it delivered a correct answer within the first three ranks. Detailed analysis of the distribution of results over question types shows that AnswerBus fares moderately well for factual questions, but shows poor performance for other question types, such as cardinality, quantificational, or embedded questions. Of the remaining question types, none could be answered.

6.5 Discussion of results

Since our system is still at the stage of a prototype, the evaluations carried out above have to be taken with care. In the current phase where concept lookup and frame assignment are not yet automated, it is evident that the extension of coverage to novel types of structures has to be performed manually. However, most of the necessary adjustments can be amended in a very short time, and the gain of coverage persists, in particular for those constructions that are domain-independent, such as various types of (direct and indirect) question variants, or quantificational structures, which can thus be ported to new domains without further ado. For domain-dependent concept and frame projection we will investigate automation techniques as described in Burchardt et al. (2005b,a). More work also needs to be invested in order to reduce the current system's error rate. Here, we will investigate heuristics to identify out-of-coverage questions, using sortal restrictions, or generation of natural language questions (or answers) from the (answer-enriched) query objects, to detect inconsistencies or "misinterpretations" of the question analysis.
The comparison to AnswerBus is not to be taken as a comparison between equals: it is intended to highlight the gain that results from more knowledge-intensive approaches, in particular in the context of a hybrid system architecture, where questions that cannot be answered by the domain-specialized

QA system can be passed on to a textual (open-domain) QA component. [26] The results from our comparison experiment strongly suggest that in those application domains where databases and ontologies are maintained and actively used, a carefully designed QA system in a hybrid architecture pays off when it comes to support targeted and reliable information access through the medium of natural language.

7 Comparison to Related Work

The approach outlined here stays close to past and present work on natural language interfaces to databases (NLIDB) (see overview in Copestake and Sparck Jones, 1990; Androutsopoulos et al., 1995). Compared to the early systems in that area, such as Lunar (Woods et al., 1972) or CO-OP (Kaplan, 1984), not surprisingly, we build on more general and advanced resources and techniques in both linguistic and knowledge modeling.
Like most of these systems, we make use of a modular, layered architecture that clearly separates syntactic analysis and the construction of a semantic interpretation for the question, which is then translated to a database query, based on a mapping from linguistic expressions and predicates to database concepts (relations, attributes and values). This architecture ensures a high degree of portability, in that we use general-purpose linguistic analysers that are not tied to specific domains.

Probably the closest predecessor to our approach is the CLE system (Alshawi, 1992), which is also built on sophisticated linguistic components, including large-scale unification-based parsing and highly advanced semantic representations (QLF). [27] However, our approach goes beyond this and other traditional systems, in that we introduce two further abstraction layers, which greatly enhance the aspect of portability.
Firstly, by introducing an additional layer of lexical-conceptual frame struc-

[26] In such a system, the user can be informed about the sources and techniques the hybrid system has been using, to help the user assess the reliability of the answers.
[27] Although differences between that system and ours are subtle, there are some aspects in which modules of our system extend and improve on the earlier system. The linguistic groundedness of the CLE effort is taken a step further by adopting a unification-oriented grammatical theory as the basis of syntacto-semantic processing, namely HPSG. Grounding in a universal linguistic theory and semantic formalism also constitutes one of the cornerstones of our approach to cross-linguality. In CLE, this feature had to be addressed by grammar porting, a technique which is far more limited in scope, since it presupposes a certain degree of typological and structural uniformity of the languages. A further novel aspect is the hybrid linguistic processing architecture, which provides seamless integration of deep and shallow components utilising a unified semantic representation formalism.

ture, we achieve a higher degree of abstraction from surface-linguistic realisations and paraphrases, which reduces the number and complexity of mapping rules to database concepts. Furthermore, due to the language-independent character of frame-semantic representations, we obtain portability across languages by projecting semantic structures obtained from language-specific HPSG processing to a common representation (see Figure 6). As a consequence of this, the construction mechanisms for queries to the database or ontology can be ported to other languages without any further effort.
Secondly, the usage of an additonal layer of abstract query terms (proto queries) offers an important gain in portability across different target knowledge bases and their associated query languages (here: MySQL and SeRQL).
Finally, contrary to traditional NLIDB approaches, our architecture uses ontologies as the interface between question analysis, answer extraction and knowledge engineering.

Newer systems also adhere to the traditional, layered architecture, but add further aspects and/or use special strategies to circumvent the construction of a full-blown semantic representation as input to query construction. We briefly discuss two systems that exhibit especially interesting features.

Aqualog is a portable QA system that exclusively relies on the chunker of the GATE platform (`http://gate.ac.uk`) for linguistic analysis. The authors claim that one can do away with much of the parsed information and used a so-called *relation similarity service* (RSS) to produce ontology-compliant queries. What distinguishes this system from others is that the RSS is interactive in that it asks the user for help when it is not capable to disambiguate between, say, two possible relations between concepts. The RSS is using WordNet synsets to map natural language expressions to database relation names. In addition, if no relation candidate can be found by direct mapping, string matching is used to determine the most likely candidates.
Our system shares similarities with this approach, in using lexical-semantic resources to establish the mapping to database relations. However, by using FrameNet resources, we achieve a further abstraction level that is neutral across languages. We could build in interaction with the user to further enhance our system in several ways: the user could choose among distinct question interpretations, or could select the intended relations in the case of unconnected query contraints (see Section 5.2).

The main assumption underlying the PRECISE NLIDB of Popescu et al. (2004) is that "...people are unwilling to trade reliable and predictable user interfaces for intelligent but unreliable ones". The authors argue that for a broad class of so-called "semantically tractable" NL questions, PRECISE is guaranteed to find the right DB query. Semantically tractable questions are characterized as those questions for which all semantically relevant tokens can be mapped to database elements, and where all tokens that refer to database elements stand

in appropriate syntactic relationships in the analysis of the question, which is produced by the Charniak parser. This test for tractability is performed before query construction, and in case of failure, the user is requested to re-phrase the question. In addition, the system deals with ambiguities in the mapping from tokens to database elements, which is reduced to a graph-matching problem: the input to query construction are flow networks that involve the relations, attributes and values corresponding to tokens; disambiguation is performed by finding a maximum flow through the networks.
The most attractive feature of this system is its high precision in recognizing out-of-scope questions, by imposing strict constraints on the mapping of question terms to database elements. Our approach is similar to PRECISE in that only those surface terms appear in frame-concept mappings that stand in approriate syntactic-semantic relations. In order to detect semantic tractability in the sense of PRECISE, we could add a similar check on the completeness of this mapping, by further checking whether all content words are mapped to frames, roles or their values, and to reject the question in case this is not fulfilled.
In our approach we haven't yet encountered ambiguities in the assignment of frames, or the mapping of frames to knowledge bases, that is, we obtain a single proto query for each HPSG parse result. To account for this type of ambiguities, we could produce proto queries in disjunctive normal form, and apply structural disambiguation by query subsumption checks (see Section 6.2) or further structural constraints, such as uniqueness of attribute values.[28] As in PRECISE, remaining ambiguities could be presented to the user for disambiguation.

8 Conclusion and Future Work

We presented an approach for domain-restricted QA from structured knowledge sources, building on deep semantic analysis of the question with a modular interface between linguistically motivated semantic representations and domain-specific models, in terms of ontologies or domain databases. The architecture embodies a flexible interface to various types of knowledge storage devices and their corresponding query languages.

Our system achieves a high degree of portability, by a modular, layered architecture. The large-scale grammars we use are not tailored to a specific domain. They deliver a semantic representation (RMRS) that is uniform across languages and is shared with shallow NE recognition grammars. The frame-semantic layer accounts for multi- and crosslingual QA, by capturing linguistic

[28] The latter seems to correspond to finding a maximum flow through the network in PRECISE.

variation and paraphrases of semantically equivalent questions.[29] It offers a modular interface for the mapping of lexical concepts to domain-dependent ontologies that can be systematically adapted to new domains and languages. Moreover, the frame-semantic structures could be employed for textual QA tasks, in open or restricted domains, by matching enriched question and answer candidate analyses. Answer extraction builds on the linguistic question analysis, performing query expansion and disambiguation using ontological constraints. DB searches compute minimal spanning trees among query concepts, and efficiently integrate concept-relating paths extracted from the question representations.

Our prototype system is still small, but has been tested on a variety of question types (wh-, yes/no-, imperative, definition and quantificational questions). Future work will focus on adding further question types, such as relational questions and questions involving coordination and negation, as well as research into automation techniques for concept lookup.
An issue to be solved for the envisaged integration of open text unrestricted domain QA and restricted QA on structured, semi-structured and unstructured data is the selection of the returned response in cases where the individual searches yield different results. The strategy to be chosen might not only depend on the reliability and durability of the knowledge sources but also on the measured relative confidence of the respective search modules. Our initial experiments suggest that even in the absence of an empirically validated selection strategy, a mere preference for the restricted domain response would improve overall accuracy.

References

Aduna B.V. (2004). *User Guide for Sesame*. Online manual at http://www.openrdf.org/doc/users/userguide.html, covers Sesame r. 1.1.

Alshawi, H., editor (1992). *The Core Language Engine*. ACL-MIT Press Series in Natural Language Processing. MIT Press.

Androutsopoulos, I., Ritchie, G. D., and Thanisch, P. (1995). Natural language interfaces to databases—an introduction. *Journal of Natural Language Engineering*, 1(1):29–81.

Baader, F., Calvanese, D., McGuinness, D., Nardi, D., and Patel-Schneider, P. (2003). *The Description Logic Handbook*. Cambridge University Press.

Baker, C. F., Fillmore, C. J., and Lowe, J. B. (1998). The Berkeley FrameNet project. In *Proceedings of COLING-ACL 1998*, Montréal, Canada.

[29] Automatic frame assignment for different languages (currently, German and English) is an active research area (Gildea and Jurafsky 2002; Baldewein et al. 2004).

Baldewein, U., Erk, K., Pado, S., and Prescher, D. (2004). Semantic role labeling with similarity-based generalisation using em-based clustering. In *Proceedings of Senseval'04*, Barcelona.

Baldwin, T., Bender, E., Flickinger, D., Kim, A., and Oepen, S. (2004). Road-testing the English Resource Grammar over the British National Corpus. In *Proceedings of LREC-2004*, Lisbon, Portugal.

Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D. L., Patel-Schneider, P. F., and Stein, L. A. (2004). OWL web ontology language reference. Technical report, W3C. 10 February.

Bonami, O. (2002). A syntax-semantics interface for tense and aspect in French. In Eynde, F. V., Hellan, L., and Beermann, D., editors, *The Proceedings of the 8th International Conference on Head-Driven Phrase Structure Grammar*, pages 31–50, Stanford. CSLI Publications.

Brachman, R. J. and Schmolze, J. G. (1985). An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9:171–216.

Brickley, D. and Guha, R. V. (2004). RDF vocabulary description language 1.0: RDF Schema. Technical report, W3C.

Broekstra, J., Kampman, A., and van Harmelen, F. (2002). Sesame: A generic architecture for storing and querying RDF and RDF schema. In *Proceedings ISWC 2001*, pages 54–68. Springer.

Burchardt, A., Erk, K., and Frank, A. (2005a). A WordNet Detour to FrameNet. In *Proceedings of the 2nd GermaNet Workshop*.

Burchardt, A., Frank, A., and Pinkal, M. (2005b). Building Text Meaning Representations from Contextually Related Frames – A Case Study. In *Proceedings of the Sixth International Workshop on Computational Semantics, IWCS-06*, Tilburg, The Netherlands.

Callmeier, U. (2000). PET – a platform for experimentation with efficient HPSG processing techniques. *Natural Language Engineering*, 6(1):99–108.

Callmeier, U., Eisele, A., Schäfer, U., and Siegel, M. (2004). The DeepThought core architecture framework. In *Proceedings of LREC-2004*, pages 1205–1208, Lisbon, Portugal.

Copestake, A. (2003). Report on the Design of RMRS. Technical Report D1.1a, University of Cambridge, University of Cambridge, UK.

Copestake, A. and Flickinger, D. (2000). An open-source grammar development environment and broad-coverage English grammar using HPSG. In *Proceedings LREC-2000*.

Copestake, A., Flickinger, D., Sag, I., and Pollard, C. (2005). Minimal Recursion Semantics. To appear in: *Research on Language and Computation*.

Copestake, A. and Sparck Jones, K. (1990). Natural language interfaces to databases. *Knowledge Engineering*, 5(4):225–249.

Crouch, R. (2005). Packed rewriting for mapping semantics to KR. In *Proceedings IWCS*, Tilburg, The Netherlands.

Crysmann, B. (2003). On the efficient implementation of German verb placement in HPSG. In *Proceedings of RANLP*, pages 112–116.

Crysmann, B. (2005). Relative clause extraposition in German: An efficient

and portable implementation. To appear in: *Research on Language and Computation*.

Crysmann, B., Frank, A., Kiefer, B., Müller, S., Neumann, G., Piskorski, J., Schäfer, U., Siegel, M., Uszkoreit, H., Xu, F., Becker, M., and Krieger, H.-U. (2002). An Integrated Architecture for Deep and Shallow Processing. In *Proceedings ACL*.

Drozdzynski, W., Krieger, H.-U., Piskorski, J., Schäfer, U., and Xu, F. (2004). Shallow processing with unification and typed feature structures — foundations and applications. *Künstliche Intelligenz*, 1:17–23.

Erk, K., Kowalski, A., Padó, S., and Pinkal, M. (2003). Towards a Resource for Lexical Semantics: A Large German Corpus with Extensive Semantic Annotation. In *Proceedings of the ACL 2003*, pages 537–544.

Fleischman, M., Hovy, E., and Echihabi, A. (2003). Offline strategies for online question answering: Answering questions before they are asked. In Hinrichs, E. and Roth, D., editors, *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 1–7.

Frank, A. and Erk, K. (2004). Towards an LFG syntax—semantics interface for Frame Semantics annotation. In Gelbukh, A., editor, *Computational Linguistics and Intelligent Text Processing*. LNCS, Springer.

Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability. A Guide to the Theory of NP-Completeness*. W.H. Freeman, New York.

Gildea, D. and Jurafsky, D. (2002). Automatic labeling of semantic roles. *Computational Linguistics*, 28(3).

Guo, Y., Pan, Z., and Heflin, J. (2004). An evaluation of knowledge base systems for large OWL datasets. In *Proceedings of ISWC 2003*. Springer.

Haarslev, V. and Möller, R. (2000). High performance reasoning with very large knowledge bases. In *Proceedings of the International Workshop in Description Logics*.

Haase, P., Broekstra, J., Eberhart, A., and Volz, R. (2004). A comparison of RDF query languages. In *Proceedings ISWC 2003*, pages 502–517. Springer.

Horridge, M. (2004). A practical guide to building OWL ontologies with the Protégé-OWL plugin. Technical report, University of Manchester.

Horrocks, I. (1998). *FaCT Reference Manual*. http://www.cs.man.ac.uk/~horrocks/FaCT/.

Horrocks, I., Sattler, U., and Tobies, S. (2000). Reasoning with individuals for the description logic SHIQ. In *Proceedings of CADE-17*. Springer.

Kaplan, S. J. (1984). Designing a portable natural language database query system. *ACM Transactions on Database Systems*, 9(1):1–19.

Klyne, G. and Carroll, J. J. (2004). Resource description framework (RDF): Concepts and abstract syntax. Technical report, W3C.

Knublauch, H., Musen, M. A., and Rector, A. L. (2004). Editing description logic ontologies with the Protégé OWL plugin. In *Proceedings of the International Workshop in Description Logics*.

Kordoni, V. (2003). Valence Alternations in Modern Greek: an MRS Analysis. In Kim, J.-B. and Wechsler, S., editors, *The Proceedings of the 9th In-*

ternational Conference on Head-Driven Phrase Structure Grammar, pages 129–146, Stanford. CSLI Publications.

Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., and Miller, K. (1993). Five papers on WordNet. Technical report, Cognitive Science Laboratory, Princeton.

Müller, S. and Kasper, W. (2000). HPSG analysis of German. In Wahlster, W., editor, *Verbmobil: Foundations of Speech-to-Speech Translation*, pages 238–253. Springer, Berlin.

Neumann, G. and Sacaleanu, B. (2003). A Cross-language Question/Answering System for German and English. In *Proceedings of the CLEF-2003 Workshop*, Trondheim.

Neumann, G. and Sacaleanu, B. (2004). Experiments on Robust NL Question Interpretation and Multi-layered Document Annotation for a Cross-Language Question/Answering System. In *Proceedings of the Working Notes for the CLEF-2004 Workshop*, Bath, UK.

Niles, I. and Pease, A. (2001). Origins of the Standard Upper Merged Ontology: A proposal for the IEEE standard upper ontology. In *IJCAI-2001 Workshop on the IEEE Standard Upper Ontology*.

Niles, I. and Pease, A. (2003). Linking lexicons and ontologies: Mapping WordNet to the suggested upper merged ontology. In *Proceedings of the 2003 International Conference on Information and Knowledge Engineering*.

Niles, I. and Terry, A. (2004). The MILO: A general-purpose, mid-level ontology. In *2004 International Conference on Information and Knowledge Engineering*, Las Vegas, NV.

Oepen, S., Callahan, E., Flickinger, D., Manning, C. D., and Toutanova, K. (2002). Lingo redwoods: A rich and dynamic treebank for HPSG. In *Beyond PARSEVAL workshop at the Third International Conference on Language Resources and Evaluation, LREC-2002*, Las Palmas, Spain.

Popescu, A.-M., Etzioni, O., and Kautz, H. (2004). Towards a theory of natural language interfaces to databases. In *Proceedings of the 8th international conference on Intelligent User Interfaces*, pages 149–157.

Reynolds, D. (2004). *Jena 2 Inference support*. Online manual at http://jena.sourceforge.net/inference/index.html.

Rosetta, M. T. (1994). *Compositional translation*. Kluwer Academic Publishers, Dordrecht, The Netherlands.

Schäfer, U. (2004). Using XSLT for the integration of deep and shallow natural language processing components. In *Proceedings of the ESSLLI 2004 workshop on Combining Shallow and Deep Processing for NLP*, pages 31–40.

Subirats, C. and Sato, H. (2004). Spanish FrameNet and FrameSQL. In *Proceedings of the LREC Workshop on Building Lexical Resources from Semantically Annotated Corpora*, Lisbon, Portugal.

Uszkoreit, H., Jörg, B., and Erbach, G. (2003). An ontology-based knowledge portal for language technology. In *Proceedings of ENABLER/ELSNET Workshop* International Roadmap for Language Resources.

Woods, W. A., Kaplan, R. M., and Nash-Webber, B. L. (1972). The Lunar sci-

ences natural language information system: Final report. Technical Report BBN Report 2378, Bolt Beranek and Newman Inc., Cambridge, MA.

Yatabe, S. (2001). The syntax and semantics of left-node raising in Japanese. In Flickinger, D. and Kathol, A., editors, *The Proceedings of the 7th International Conference on Head-Driven Phrase Structure Grammar*, pages 325–344, Stanford. CSLI Publications.

Zheng, Z. (2002). AnswerBus Question Answering System. In *Proceedings of the Human Language Technology Conference (HLT 2002)*, San Diego, CA.