# WHITEBOARD - Eine XML-basierte Architektur für die Analyse natürlichsprachlicher Texte

Günter Neumann
Ulrich Schäfer

Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI) GmbH,
Language Technology Lab, Saarbrücken, http://www.dfki.de/lt

Inhaltsverzeichnis

# WHITEBOARD - Eine XML-basierte Architektur für die Analyse natürlichsprachlicher Texte

Günter Neumann
Ulrich Schäfer

Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI) GmbH, Language Technology Lab, Saarbrücken, http://www.dfki.de/lt

## *1. Einleitung*

### 1.1. Das Projekt

Ziel des WHITEBOARD-Projektes[1] ist die Entwicklung, Implementierung und Evaluation einer neuartigen Systemarchitektur, welche die Kombination von Sprachtechnologien für praktische Anwendungen erlaubt. Sprachtechnologien bieten verschiedenartige Möglichkeiten für eine partielle Analyse von Texten, die für information retrieval, information extraction, language checking und viele weitere Anwendungen genutzt werden können. Die Verarbeitungsmethoden und -werkzeuge unterscheiden sich in vielerlei Dimensionen, zum Beispiel bezüglich der Ebenen linguistischer Beschreibung, der Tiefe der Analyse oder der Art, in der Wissen abgeleitet wird (linguistisch oder statistisch). Die Funktionalität der Methoden ist häufig überlappend, sie unterscheiden sich jedoch in ihren Stärken und Schwächen. Eine der schwierigsten Aufgaben der Sprachverarbeitung ist die Suche nach optimalen Kombinationen heterogener Techniken und Verarbeitungskomponenten – die Herausforderung für das WHITEBOARD-Projekt.

### 1.2. Die Architektur

Die neue Architektur, die entwickelt und untersucht wird, basiert auf dem Konzept eines (automatisch) annotierten Textes. Die verschiedenen Sprachtechnologie-Komponenten reichern einen mit XML annotierten Text mit Meta-Information, die ebenfalls in XML kodiert ist, an. Jede Komponente kann vorher zugewiesene Annotationen nutzen oder unbeachtet lassen. Die WHITEBOARD-Architektur besteht aus einer einzigen geteilten Datenstruktur, die gleichzeitig Eingabe, Zwischenrepräsentation und Ausgabe des Systems ist. Diese Architektur ermöglicht die pragmatische Kombination verschiedener Verarbeitungsansätze, wobei neue Wege der Kombination flacher und tiefer Verarbeitungsmethoden aufgezeigt werden.

### 1.3. Die Komponenten

WHITEBOARD baut auf existierenden Komponenten des DFKI Language Technolgy Lab auf: Das morphologische Verarbeitungssystem Morphix, die Tagger und Phrasen-Parser TnT

[1] Das Projekt WHITEBOARD wird vom Bundesministerium für Bildung, Wissenschaft, Forschung und Technologie (BMB+F) gefördert. Förderkennzeichen 01IW002.

[1], Chunkie [8] und SPPC [6], das Information-Extraction-System SMES [3], das effiziente HPSG-Parsing-System PET [2,7], HPSG-Grammatiken für Deutsch, Englisch und Japanisch. Die Komponenten spiegeln die gesamte Bandbreite moderner statistik-, regelbasierter und hybrider Verfahren, sowie flacher und tiefer Verarbeitungsansätze wider.

### 1.4. Die Anwendungen

Zwei Anwendungen werden zum Zweck der Evaluation und Demonstration der Ergebnisse realisiert. Eine dieser Anwendungen ist information extraction. Da das automatische Verstehen ganzer Texte für absehbare Zeit nicht erreichbar zu sein scheint, ist die Strategie eine graduelle Erweiterung unserer Information-Extraction-Technologie.
Die zweite Anwendung ist controlled language checking. Auch hier können wir von der heutigen Technologie keine vollständige und korrekte Analyse eines gesamten Texts erwarten. Wir könnten allerdings in der Lage sein, unsere tiefe Analyse in der Art zu spezialisieren, dass sie in bestimmten Umgebungen, die für die korrekte Diagnose und Korrektur von Fehlern relevant sind, mit ausreichender Präzision eingesetzt werden kann.

Aktuell sehr im Kommen sind QA-Systeme (Frage-Antwort-Systeme). Sie stellen eine Kombination von Methoden des information retrieval und information extraction dar. Auf eine natürlichsprachliche Eingabe hin kann ein QA-System Dokumente oder Webseiten nach passenden Dokumenten durchsuchen und als Resultat mit Hilfe der gefundenen Dokumente eine natürlichsprachliche Antwort extrahieren. Im Rahmen eines WHITEBOARD-assoziierten Projektes wurde ein erster deutschsprachiger Demonstrator realisiert [3], s. Abschnitt 4.

Schließlich bildet die WHITEBOARD-Architektur auch eine Grundlage für den automatisierten Wissenserwerb (machine learning) durch Analyse großer Textmengen. Durch Kombination verschiedener Sprachverarbeitungskomponenten (mittels statistik- oder regelbasierter Algorithmen) aufbereitete linguistischen Ressourcen können selbst wieder als Datenquellen für neue oder verbesserte Sprachverarbeitungskomponenten dienen (Bootstrapping-Prozess, z.B. für template learning).

## *2. XML-basierte Architektur*

Bereits vor dem XML-Hype seit Mitte der 90er Jahre gab es in der Computerlinguistik zahlreiche Ansätze, Texte automatisch oder manuell mittels SGML-tags zu annotieren (vgl. TEI). In WHITEBOARD wird die automatische XML-Annotation auf allen linguistischen Beschreibungsebenen angewandt und ist Teil der Softwarearchitektur. Dabei übernimmt XML auch die Funktion des universellen Kommunikationsvehikels zwischen Modulen, die andernfalls nicht direkt miteinander kommunizieren können (XML-chart). Während viele vorhandene Module z.B. in C++ implementiert sind, werden Architekturkern und die darauf aufbauenden Anwendungen in Java implementiert. So müssen nur XML-Schnittstellen für vorhandene Module implementiert werden (sofern diese nicht ohnehin bereits existierten).

Neben der Natürlichkeit des Konzeptes, Dokumente durch "Annotation" mit Metainformation anzureichern, gibt es zahlreiche weitere Vorteile, XML zu verwenden, welches sich durch gegenüber SGML vor allem durch ein schlankeres und effizienter zu verarbeitendes Framework auszeichnet: XML kann als "lingua franca" für Textannotation dienen, es ist kompatibel mit vorhandener struktureller Dokumentinformation z.B. aus HTML-tags oder anderen Dokumentauszeichnungsstandards und Metasprachen des W3C.

Durch monotones Anreichern von Information wird ein Mindestmaß an Robustheit erzielt: Versagt eine anreichernde Komponenten, so kann in gewissem Umfang ein Fallback auf die vorhergehende Annotationsebene erfolgen. Auch eine nicht-monotone "Reparatur" falscher oder unzureichender Zwischenergebnisse von Komponenten ist möglich (vgl. language checking). XML und die zahlreichen dafür vorhandenen Werkzeuge für query, Transformation und Extraktion bilden eine gute Grundlage für rapid prototyping z.B. bei der Entwicklung neuer Verfahren. So ist auf XML-Ebene eine einfache Simulation noch nicht implementierter Komponenten möglich. Teilergebnisse der Annotation können einfach gespeichert oder transformiert und offline weiterverwendet werden oder z.B. in Datenbanken abgelegt und wieder entnommen werden.

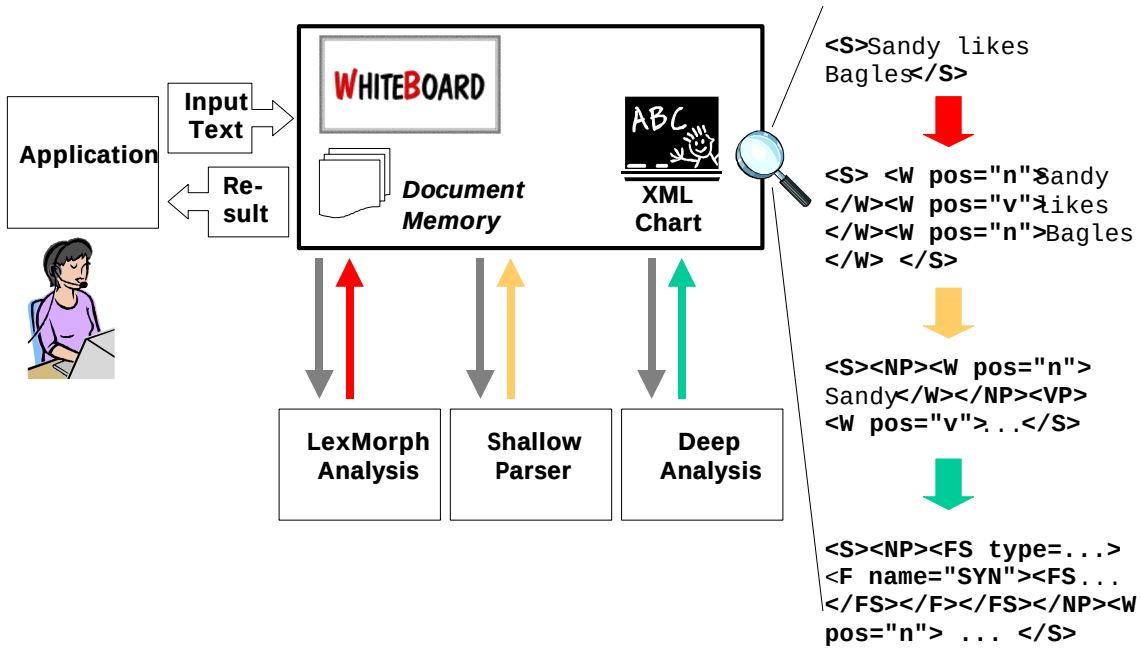


**Abbildung 1: Die Whiteboad-XML-chart (stark vereinfachtes Beispiel)**

## 3. XML-Annotation: Ein Beispiel

Wie sieht nun die XML-Annotation konkret aus? Im Folgenden soll ein Beispiel für die schrittweise Anreicherung der XML-chart durch die Sprachverarbeitungskomponenten gegeben werden (XML chart layer). Die Aufzählungsreihenfolge der Komponenten entspricht der Verarbeitungsabfolge, d.h. in der Regel baut eine Komponente auf den Ergebnissen der vorangegangenen Komponente(n) auf, wobei die Layer entweder durch Einbettung oder durch (Hyper)links verbunden werden:

1. Tokenisierung: Die Worte und Symbole, aus denen der Text besteht, werden anhand einfacher Regeln klassifiziert und mit dem W-Tag (W für word) umschlossen. Beispielannotation für "30. April":

```
<W tokenclass="number_followed_by_period">30.</W>
<W tokenclass="first_capital_word">April</W>
```

2. Lexikalisch-morphologische Analyse: Anhand eines Lexikons und unter Anwendung morphologischer Regeln werden Hypothesen über die Wortklasse (pos: part of speech) und Kasus, Flexion usw. hinzugefügt. Zur lexikalisch-morphologischen Analyse gehört auch die Erkennung von Komposita, also zusammengesetzten Wörtern, deren Bestandteile zwar Lexikoneinträge besitzen, nicht jedoch all ihre Kombinationen (vgl. "Donaudampfschifffahrtskapitän").

```
<W tokenclass="first_capital_word" lexentry="yes">
   April <READING stem="april" pos="N">
           <INFL gender="M" case="NOM" number="SG" />
           <INFL gender="M" case="DAT" number="SG" />
           <INFL gender="M" case="AKK" number="SG" />
         </READING>
</W>
```

3. Wortklassen-Filterung: regelbasiert oder statistisch werden bestimmte Wortklassen-Hypothesen ausgeschlossen, auch ein Ranking der wahrscheinlichsten Hypothesen ist möglich.

```
<W tokenclass="first_capital_word" lexentry="yes" preferred_reading="N">
      April <READING stem="april" pos="N">… </READING></W>
```

4. Erkennung von Eigennamen, Orts- und Zeitangaben (named entities) mittels regel- und/oder statistikbasierter Verfahren: z.B. "Richard Hirschmann GmbH". Dazu muss nicht jedes Wort im Lexikon gefunden werden; zur Erkennung des Firmennamens genügt hier ein Pattern X + "GmbH". "Hirschmann" wird hier lediglich aufgrund der Kompositazerlegung in "hirsch" und "mann" (vgl. 2.) als Lexikoneintrag erkannt.

```
<NAMED_ENTITY index="2" type="ORGANIZATION"
    subtype="ORGANIZATION_COMPANY">
  <W tokenclass="first_capital_word" lexentry="no">Richard</W>
  <W tokenclass="first_capital_word" comp="hirsch mann" pre-
      ferred_reading="N" lexentry="yes">Hirschmann</W>
  <W tokenclass="mixed_word_first_capital" stem="gmbh" pre-
      ferred_reading="N" lexentry="yes">GmbH</W>
</NAMED_ENTITY>
```

5. Phrasen-Erkennung (chunking) fasst mehrere W-tags (oder named entities ) nach linguistischen Kriterien zusammen: "aus dem Unternehmen"

```
<CHUNK type="PP"><W …>aus</W> <W …>dem</W> <W
    …>Unternehmen</W></CHUNK>
```

6. Nebensatz- und Satzgrenzenerkennung (aus Gründen der Übersichtlichkeit ohne die eingebetteten XML-tags):

```
<S><SC>Er werde in Kürze in Personalunion Hirschmanns Amt
  übernehmen,</SC>teilte Rheinmetall mit.</S>
```

7. Die tiefe Analyse (z.B. zur genauen Semantikkonstruktion) kann auf Satzebene weitere komplexe Repräsentationsebenen einführen. Einer der Schwerpunkte des WHITEBOARD-Projektes ist die Kombination der flachen Verfahren (Ebenen 1-6) mit der tiefen Analyse auf Basis unifikationsbasierter getypter Merkmalsstrukuren, welche direkt oder mittels Abstraktion in XML repräsentiert werden können. Aus

Platzgründen kann auf diesen Aspekt von WHITEBOARD hier nicht näher eingegangen werden.

8. Die Ontologie-Suche dient der Abbildung von lexikalischer auf die semantische Ebene mittels Ontologie-Datenbank als Vorstufe für weitere Verfahren des Textverstehens und der Textextraktion. Beispiel: GermaNet-Hypernym-Hierarchie zu "GmbH" (in der Regel wird allerdings nur ein kleiner Ausschnitt der Ontologie-Hierarchie in XML annotiert):

```
<SEM-LEXICON-ANNOTATION>
  <ENTRY WORD_form="gmbh" pos="n">
    <WORD>Gesellschaft_mit_beschränkter_Haftung</WORD>
    <WORD>GmbH</WORD>
    <HYPERNYM>
      <WORD>Gesellschaft</WORD>
      <HYPERNYM>
        <WORD>Vereinigung</WORD>
        <HYPERNYM>
          <WORD>Organisation</WORD>
          <HYPERNYM>
            <WORD>Zusammenschluss</WORD>
            <HYPERNYM>
              <WORD>Gruppe</WORD>
              <HYPERNYM>
                <WORD>Teilmenge</WORD>
                <HYPERNYM>
                  <WORD>Teil</WORD>
                </HYPERNYM>
              </HYPERNYM>
            </HYPERNYM>
          </HYPERNYM>
        </HYPERNYM>
      </HYPERNYM>
    </HYPERNYM>
  </ENTRY>
</SEM-LEXICON-ANNOTATION>
```

9. Auf der Anwendungsebene kann schließlich das Resultat ebenfalls der XML-chart hinzugefügt werden. Dadurch kann nachvollzogen werden, aufgrund welcher Beiträge der einzelnen Komponenten das Ergebnis zustande kam. Bei der Anwendung Informationsextraktion wird die erwartete Analyse mehrerer Sätze in domänenabhängigen "templates" zusammengefasst. Dem template filling liegen komplexe Operationen zugrunde, welche die Ergebnisse mehrerer darunterliegender XML-layer verwenden, z.B. lexikalisch-morphologischer Analyse, named entity-Erkennung und tiefer Analyse. Am Beispiel für ein gefülltes template aus dem Bereich der Stellenwechsel-Domäne in Wirtschaftsmeldungen wird ersichtlich, dass diese in einem Format vorliegen, welches direkt in einer Datenbank gespeichert bzw. weiterverarbeitet werden kann:

```
<TEMPLATE type="management_succession_event">
        <COMPANY>Richard Hirschmann GmbH</COMPANY>
        <PERSON_OUT>Gerhard Jaskulke</PERSON_OUT>
        <PERSON_IN>Klaus Eberhardt</PERSON_IN>
        <DATE>30. April</DATE>
</TEMPLATE>
```

Aus den oben genannten Beispielen ergeben sich bereits einige komplexe Fragestellungen, die im Forschungsprojekt zumindest in Ansätzen gelöst werden sollen: Auf welche Weise sollen sich überlappende linguistische Repräsentationen dargestellt werden? Wie können nichtlokale Abhängigkeiten (z.B. "Wann **finden** die Filmfestspiele in Berlin in diesem Jahr **statt**?") repräsentiert werden? Wie löst man das Problem der Repräsentation multipler Lesarten und sich daraus ergebender Kombinationen? Auf welche Weise erfolgt die Auswahl und Kombination von mit numerischen Bewertungen belegten Lesarten, z.B. bei Ergebnissen statistisch basierter Verfahren?

## *4. Anwendung: Web-basierte Informationsextraktion*

Abschließend soll eine Anwendung der WHITEBOARD-Architektur im Bereich der Web-basierten Informationsextraktion beschrieben werden.
Ausgehend von einer domänenspezifischen Ontologie (Tourismus) ist das Ziel die Identifikation und Extraktion von relevanten Textstellen und die Bestimmung ihrer domänenspezifischen Beziehungen (abstract). Nach dem Parsen einer natürlichsprachlichen Benutzeranfrage sollen die relevanten, d.h. zur Frage passenden Textstellen ausgegeben werden. Im folgenden werden die linguistisch relevanten Komponenten eines solches Systems beschrieben.

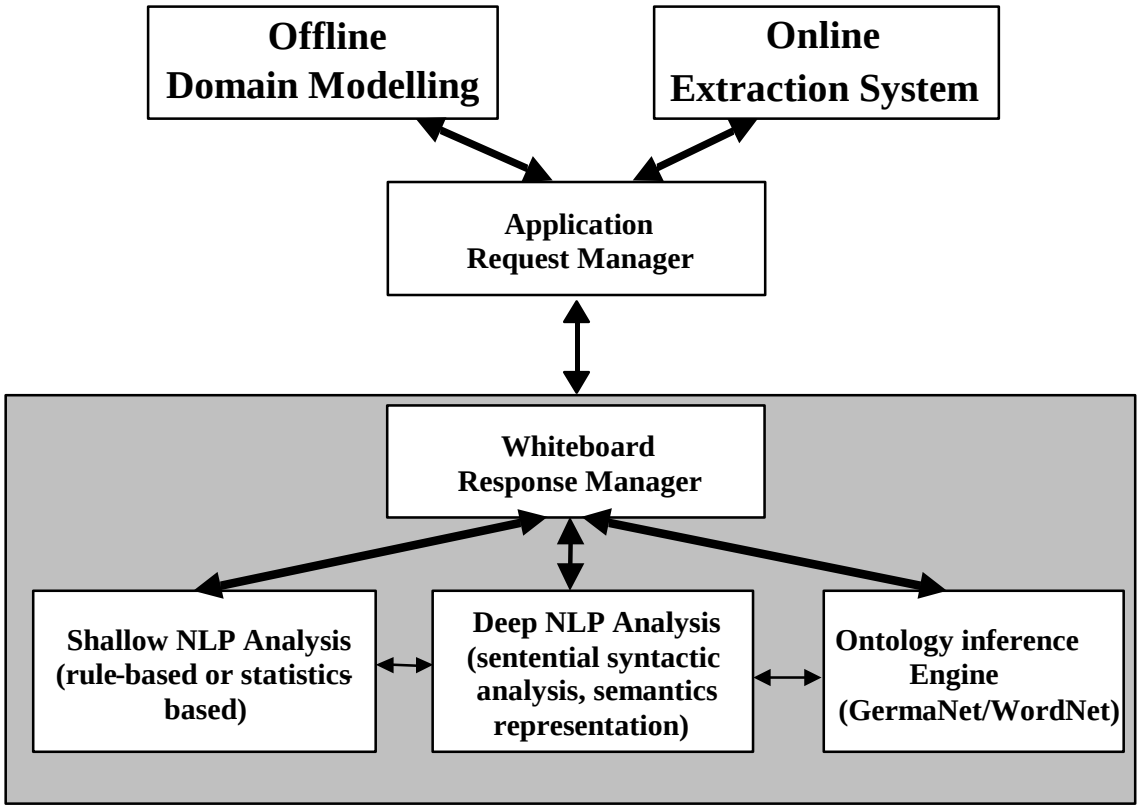


**Abbildung 2: Beispielarchitektur für Information Extraction**

## 4.1. Abstractgenerierung

Ein Abstract stellt eine kondensierte, semantische Repräsentation der relevanten Teile eines Textes dar. Bei der Berechnung von Abstracts verfolgen wir eine optimistische bottom-up Strategie. Zuerst werden einfache Beziehung zwischen relevanten Wörtern und Konzepten mittels eines Domänenlexikons bestimmt. Das Domänenlexikon definiert eine unmittelbare Beziehung zu Wörtern (genauer zu Lemmata) und Konzepten aus der Ontologie.
Zunächst werden hierbei nur Wörter bzw. Stämme von Nomen betrachtet. Die Domänenlexika sind manuell erstellt, wobei die Auswahl der relevanten Nomen korpusbasiert durch den Einsatz von statistischen Verfahren (z.B. n-grams) automatisch unterstützt wird. Nach der syntaktischen Analyse des Textes können nun alle nominalen Terme, die einen Bezug zum Domänenlexikon aufweisen, extrahiert werden. Dabei handelt es sich um einzelne Wörter, Eigennamen, aber auch einfache und komplexe Nominal- und Präpositionalphrasen. Während der Termextraktion werden Instanznamen vergeben, mit Hilfe derer textlinguistische Bezüge modelliert werden. Während die anaphorische Resolution von normalen Nominalphrasen über rein linguistische Verfahren hinausweist, wurde darauf geachtet, Eigennamen als rigiden Designatoren einheitliche Instanznamen zuzuweisen.

Zentraler Aspekt bei der Abstractgenerierung ist die Erstellung relationaler Beziehungen zwischen diesen Termen, die prinzipiell durch Inferenz über der Ontologie möglich ist. Daher werden in einem nächsten Schritt alle möglichen Paare über der Menge der nominalen Terme gebildet (Kandidatenmenge) und der Inferenzmaschine der Ontologie übergeben. Sie überprüft nun, für welche Paare welche Relationen definiert sind und führt damit eine Filterung durch. Alle dadurch relationierten Termpaare (relationale Tupel) definieren den Abstract für ein Textdokument. Für die Erstellung von Abstracts aus den Ergebnissen der domänenspezifischen Termextraktion werden Heuristiken eingesetzt, um die Größe der Kandidatenmenge sinnvoll zu beschränken. Dabei werden linguistische Heuristiken eingesetzt (z.B. minimal attachment: kombiniert aufeinanderfolgende Nominal- und Präpositionalphrasen, für die keine vollständige Syntaxanalyse bestimmt wurde; eine Modifier-Heuristik verbindet das Substantiv einer Phrase mit allen vorkommenden Modifikatoren) und Heuristiken, die sich auf Spezialausdrücke beziehen (Adressen-Heuristik: erzeugt eine Adresseninstanz und verbindet sie mit allen identifizierten Bestandteilen der Adresse) bzw. Heuristiken, welche die HTML-Struktur berücksichtigen (z.B. Titelheuristik: kombiniert die zwischen den Titeltags stehenden Terme mit allen Termen des Fließtextes). Hier ein Beispiel für einen Abstract für den Satz *Wir bieten ein Hotel mit ruhigen Zimmern.*

```
<LIST>
  <TUPLE name="output">
    <FRAGMENT>
      <TYPE>Zimmer</TYPE>
      <INST>Zimmer_193</INST>
      <NAME>zimmer</NAME>
    </FRAGMENT>
    <FRAGMENT>
      <TYPE>Zimmer</TYPE>
      <INST>Zimmer_193</INST>
      <NAME>ruhig</NAME>
    </FRAGMENT>
    <CONSTRAINT type="HEURISTIC">MODIFIER</CONSTRAINT>
  </TUPLE>
  <TUPLE name="output" rel="hat_Zimmer">
    <FRAGMENT>
      <TYPE>Hotel</TYPE>
      <INST>Hotel_192</INST>
```

```
        <NAME>hotel</NAME>
      </FRAGMENT>
      <FRAGMENT>
        <TYPE>Zimmer</TYPE>
        <INST>Zimmer_193</INST>
        <NAME>zimmer</NAME>
      </FRAGMENT>
      <CONSTRAINT type="HEURISTIC">SENTENCE-HEURISTIC NP-PP-
        HEURISTIC</CONSTRAINT>
    </TUPLE>
  </LIST>
```

Die Verbindung der Konzepte *Zimmer* und *Hotel* wurden gleich von zwei Heuristiken vorgeschlagen und von der Ontologie mit der Relation *hat_Zimmer* versehen.

## 4.2. Parsing von Benutzeranfragen

Die einheitliche Architektur für die Verarbeitung von domänenrelevanten Dokumenten und Anfragen sieht die Verwendung derselben Sprachverarbeitungskomponenten vor. Der Dialogkomponente werden allerdings nicht wie in der Abstractgenerierung relationierte Tupel angeboten, sondern alles analysierte Material, da bei einfachen Anfragen möglicherweise gar keine Relationen gefunden werden. Die XML-Schnittstelle muss dabei nicht nur gefundene chunks repräsentieren, sondern auch Wörter, die nicht zu Phrasen zusammengefasst wurden, um eine robuste Anfrageanalyse zu gewährleisten. Folgendes Beispiel zeigt das Parsingergebnis für die Anfrage *Gibt es hier ein Hotel?* (nicht alle Information ist dargestellt):

```
<QUERY>
  <S>
    <CHUNK type="VG">
      <W tokenclass="first_capital_word" lexentry="yes" preferred_reading="V">
        Gibt<READING stem="geb" pos="V" />
      </W>
    </CHUNK>
    <CHUNK type="NP">
      <W tokenclass="lowercase_word" lexentry="yes"
          preferred_reading="PERSPRON">
        es<READING stem="es" pos="PERSPRON" />
      /W>
    </CHUNK>
    <W tokenclass="lowercase_word" lexentry="yes" preferred_reading="ADV">
      hier<READING stem="hier" pos="ADV" />
    </W>
    <CHUNK type="NP">
      <W tokenclass="lowercase_word" lexentry="yes" preferred_reading="INDEF">
        ein<READING stem="ein" pos="V" />
            <READING stem="ein" pos="INDEF" />
            <READING stem="ein" pos="VPREF" />
      </W>
      <W tokenclass="first_capital_word" lexentry="yes" preferred_reading="N">
        Hotel <READING stem="hotel" pos="N" />
      </W>
      <NPSEM>
        <DET>ein</DET>
        <HEAD>Hotel</HEAD>
        <DOMAINTYPE>Hotel</DOMAINTYPE>
```

```
        </NPSEM>
      </CHUNK>
      <W tokenclass="separator_symbol" lexentry="yes" preferred_reading="INTP">
        ? <READING stem="QUESTSIGN" pos="INTP" />
      </W>
    </S>
  </QUERY>
```

Das Token "hier" wurde keiner Phrase zugeordnet, deshalb wird das Ergebnis der morphologischen Anfrageanalyse geliefert. "ein Hotel" wurde als NP mit dem Domänentypen HOTEL erkannt und das Fragezeichen als Satzendemarker mit Fragesemantik.

## 4.3. Generierung von Suchergebnissen

Wesentliche Aufgabe der Textgenerierung ist die Erstellung von natürlichsprachlichen Textzusammenfassungen auf Basis des Ergebnisses der Auswertung einer Datenbankanfrage. In der Regel handelt es sich hierbei um relevante Stellen aus den aktivierten Dokumenten und ihre entsprechenden Abstracts, die im Rahmen der Textkondensation bestimmt wurden. Multilinguale Anwendungen erfordern die Generierung von Zusammenfassungen in der Sprache, in der auch die Anfrage formuliert wurde. Dies bedeutet insbesondere, dass auch die Textgenerierung multilinguale Funktionalität zur Verfügung stellen muss.

Das Generierungsverfahren basiert auf parametrisierbaren Textschablonen (auch Generierungs-Templates genannt). Diese bestehen aus linguistischen Mustern (konkrete Wörter, Stämme, Phrasen) und Typen aus der Ontologie. Die Typen stellen Platzhalter dar für die durch den Datenbankzugriff bestimmten Instanzen der Abstracts. Beispiel: "[Stadt] hat viele [Hotel]", "Das Hotel [Hotel] befindet sich in [Stadt]." oder "[Hotel] hat von [Start-Time] bis [End-Time] geöffnet."

Anstatt die Generierungs-Templates durch aufwendige Inspektion der Textmenge manuell zu erstellen, folgen wir einem reversiblen und lernbasierten Ansatz. Reversibilität bedeutet hierbei die Verwendung derselben Wissensquellen in Parsing- und Generierungsrichtung. Im aktuellen Kontext bedeutet Reversibilität die Verwendung von durch die Textkondensation berechneten Abstracts und ihrer linguistischen Zwischenergebnisse (auf Basis von XML) zur Erstellung der Textschablonen.

Zur Bestimmung von Generierungs-Templates haben wir eine automatische Methode auf Basis des explanation-based learnings (EBL) entwickelt und in Java implementiert [4]. Das Verfahren besteht aus 2 Schritten: a) Erstellung der Textschablonen, b) Erstellung eines semantischen Index auf die Patterns. Der erste Schritt basiert auf einer Bestimmung der relevanten analysierten Textpassagen und ihrer approximativen Generalisierung. Im zweiten Schritt wird ein Entscheidungsbaum erstellt mit den domänenspezifischen Typen aus den korrespondierenden Abstracts. Diese dienen als Traversierungshilfen zu Generierungspatterns für entsprechende semantische Eingaben, wie z.B.

```
<ABSTR>
  <CONCEPT name="Unterkunft">
    <NAME>Hotel Schoenborn</NAME>
    <SLOT name="in_Gebiet">
      <CONCEPT name="Stadt">
        <NAME>Rostock</NAME>
      </CONCEPT>
    </SLOT>
```

```
  </CONCEPT>
</ABSTR>
```

Dies führt zur Auswahl folgender Patterns (sortiert nach Frequenz):

[NP $\text{Head}_{\text{Unterkunft}}$] [V liegt] [PP in [NP $\text{Head}_{\text{Stadt}}$]]
[NP $\text{Head}_{\$C1}$] [V liegt] [PP in [NP $\text{Head}_{\$C2}$]]
[NP $\text{Head}_{\text{Hotel}}$] [V befindet sich in] [NP $\text{Head}_{\text{Hafenstadt}}$]]
[NP $\text{Head}_{\$C1}$] [V befindet sich in] [NP $\text{Head}_{\$C2}$]]
[NP $\text{Head}_{\text{Hotel}}$] [PP in [NP $\text{Head}_{\text{Stadt}}$]]

Was letztlich zur Instanzierung der folgenden alternativen Patterns führt (sortiert gemäß Ranking):

[NP Hotel Schoenborn] [V liegt] [PP in [NP Rostock]]
[NP Hotel Schoenborn] [V befindet sich in] [NP Rostock]

Dieses Verfahren zeichnet sich durch eine so hohe Robustheit aus, dass es das manuelle Erstellen von Textschablonen unnötig macht. Multilingualität wird durch einfachen Austausch der Patterngrammatik erreicht. Zur Erzeugung nicht-trivialer Sätze werden Methoden zur statistisch gesteuerten Kombination von Pattern eingesetzt.

## *5. Literatur*


[1] T. Brants: *TnT - A Statistical Part-of-Speech Tagger*. In Proceedings of the Sixth Applied Natural Language Processing Conference ANLP-2000, Seattle, WA, 2000.

[2] U. Callmeier: *PET. A platform for experimentation with efficient HPSG processing techniques.* Journal of Natural Language Engineering, (Special Issue on Efficient processing with HPSG: Methods, systems, evaluation). Editors: Flickinger, D., Oepen, S., Uszkoreit, H., Tsujii, J. Cambridge, UK: Cambridge University Press, 2000.

[3] M. Klettke, M. Bietz, I. Bruder, A. Heuer, D. Priebe, G. Neumann, M. Becker, J. Bedersdorfer, H. Uszkoreit, A. Maedche, S. Staab und R. Studer: *GETESS - Ontologien, objektrelationale Datenbanken und Textanalyse als Bausteine einer semantischen Suchmaschine (in German).* Datenbank-Spektrum, S. 14-24, 1/2001.

[4] G. Neumann: *Applying Explanation-based Learning to Control and Speeding-up Natural Language Generation.* In Proceedings of ACL/EACL-97, Madrid, 1997.

[5] G. Neumann: *Informationsextraktion.* In Klabunde et al (eds): Computerlinguistik und Sprachtechnologie - Eine Einführung. Spektrum Akademischer Verlag, Heidelberg, 2001.

[6] J. Piskorski and G. Neumann: *An Intelligent Text Extraction and Navigation System.* In proceedings of 6th International Conference on Computer-Assisted Information Retrieval (RIAO-2000), Paris, 2000.

[7] Carl J. Pollard and Ivan A. Sag. *Head-Driven Phrase Structure Grammar*. Chicago: University of Chicago Press, 1994.

[8] W. Skut and T. Brants: *Chunk tagger – statistical recognition of noun phrases*. In Proceedings of the ESSLLI Workshop on Automated Acquisition of Syntax and Parsing, Saarbrücken, Germany, 1998.

[9] C.M. Sperberg-McQueen, L. Burnard: *TEI Guidelines for Electronic Text Encoding and Interchange (P3)*. Chicago and Oxford, 1994.