RealPro, GOSSIP: Meaning-Text Theorie basierte Realisierer für die Generierung natürlicher Sprache

Reinhard Kuntz
Hauptseminar Natürlichsprachliche Benutzerschnittstellen

30.05.2006

Abstract

In dieser Arbeit werden nach einer kurzen Einführung in die Meaning-Text Theorie (MTT) einige auf dieser Theorie basierende Systeme vorgestellt. Es folgt eine tiefer gehende Betrachtung der Systeme GOSSIP und RealPro. Dabei wird das Augenmerk besonders auf verschiedene Aspekte der Umsetzung der MTT in die Praxis gelegt.

1 Einleitung

In [Kahane, S. 1] heißt es, dass „das Ziel der MTT ist, Systeme mit expliziten Regeln für den Zusammenhang zwischen Bedeutung und Text in verschiedenen Sprachen zu realisieren.“ Ob dieses Ziel der MTT (siehe Abschnitt 2) in der Praxis umgesetzt werden kann, soll in dieser Arbeit betrachtet werden. Dabei liegt der Schwerpunkt nicht etwa auf der MTT, sondern bei Ansätzen diese in praxistaugliche Systemen umzusetzen. Deshalb sollen nach einer kurzen Einführung in die MTT zunächst einige MTT-basierte Systeme vorgestellt werden. Darauf folgend wird auf die Systeme GOSSIP (siehe Abschnitt 4) und RealPro (siehe Abschnitt 5) genauer eingegangen, indem die Ansätze und Architekturen der beiden Systeme vorgestellt werden. Außerdem soll versucht werden herauszuarbeiten, welche Ansätze in den Systemen verfolgt werden, um die Beziehung zwischen Meaning und Text zu realisieren. Diese m-n-Beziehung (siehe Abschnitt 2) würde bei naiver Implementierung exponentiell viele Möglichkeiten und damit einen für die Praxis unbrauchbaren Aufwand ergeben. Außerdem soll die Umsetzung der „Meaning-Text Model“ (siehe Abschnitt 2) in den Systemen betrachtet werden. Die Trennung von Algorithmen bzw. Verfahren von den linguistischen Regeln soll dabei verstärkt in Betracht gezogen werden, da dies neben anderen Aspekten Voraussetzung für domänen- und sprachunabhängige MTM ist, welche für weitere Systementwicklungen sehr wünschenswert wären.

2 Meaning-Text Theorie

In diesem Abschnitt soll kurz beschrieben werden, was die Meaning-Text Theorie ist bzw. beinhaltet. Die MTT wurde in Moskau von Zolkovski und Mel’cuk im Rahmen von Forschungsarbeiten im Bereich Maschinelle Übersetzung entwickelt [Kahane 2001, S. 1]. Wie schon in Abschnitt 1 beschrieben, besteht das Ziel der MTT darin, „Systeme mit expliziten Regeln für den Zusammenhang zwischen Bedeutung und Text in verschiedensten Sprachen zu realisieren“ [Kahane, S. 1].

Für diesen Zweck stellt die MTT im wesentlichen drei Postulate auf, durch die eine natürliche Sprache laut der MTT definiert ist. Das erste Postulat besagt, dass eine natürliche Sprache eine viele-zu-viele Beziehung zwischen „meanings“ und „texts“ ist, woher auch der Name der Theorie herrührt. Unter „meanings“ sind dabei Inhalte bzw. Bedeutungen zu verstehen. Mit „texts“ sind Formulierungen jeglicher Länge – Phrasen, Sätze, Abschnitte – ge-

meint. Auch kann „texts“ als Gesprochenes angesehen werden [Kahane 2001, S. 1].

Das zweite Postulat beschreibt diese Beziehung zwischen Meaning und Text als eine formale Einheit, die die linguistischen Aktivitäten eines Muttersprachlers simuliert [Kahane 2001, S. 1]. Diese formale Einheit wird als „Meaning-Text Model“ (MTM) bezeichnet und beschreibt mit einer endlichen Menge von Regeln alle korrespondierenden Paare von Semantik und Phonetik. Im Endeffekt stellt eine MTM also nichts anderes als eine Grammatik dar [Kahane, S. 2]. Wie die Intuition eines Muttersprachlers simuliert werden soll, bleibt in der MTT offen.

In den meisten Sprachtheorien gibt es Repräsentationsebenen zwischen Semantik und Phonetik. Auch die MTT definiert solche im dritten Postulat. Im Unterschied zu anderen Theorien wird in der MTT jedoch differenzierter mit den Schichten zwischen Meaning und Text verfahren. Insgesamt gibt es sieben Repräsentationen, wie in Abbildung 1 dargestellt. Dabei wird in der MTT großer Wert darauf gelegt, dass Bereiche unterschiedlicher Dimensionen vorliegen. Bei den semantischen Repräsentationen handelt es sich um Graphen mit Prädikat-Argument Relationen, während bei den syntaktischen Repräsentationen Bäume („non ordered dependency trees“) und bei den morphologischen Repräsentationen Ketten bzw. „Strings“ vorliegen [Kahane 2001, S. 3]. Die Meaning-Text Beziehung ist als vollkommen modular anzusehen. So kann ein MTM ebenfalls in sechs Teile aufgeteilt werden, einer für jedes Modul (siehe Abb. 1 zwischen den Repräsentationen [Kahane, S. 3+4].

3 MTT-basierte Systeme

Nachdem im letzten Abschnitt eine kurze Einführung in die MTT gegeben wurde, sollen nun einige Systeme vorgestellt werden, die diese umsetzen. Wie in Abschnitt 2 bereits erläutert, ist die MTT im Bereich der maschinellen Übersetzung entstanden. So ist es nicht verwunderlich, dass die ersten Realisierungen der MTT mit den Systemen ETAP1 und ETAP2 in diesem Bereich zu finden sind. ETAP1 ist zur Übersetzung vom Französischen ins Englische und ETAP2 übersetzt vom Englischen ins Russische [Kahane, S. 31+32]. Bei dem Nachfolgersystem ETAP3 aus jüngerer Zeit gibt es die Möglichkeit zwischen mehreren Sprachpaaren zu wählen.

Im Bereich der Generierung natürlicher Sprache ist GOSSIP (Generation Of Operation System Summary In Prolog) [Kittredge, Iordanskaja, Polguère, S. 8] das erste System, welches die MTT umsetzt. GOSSIP wurde in Quintus Prolog Version 2.0 implementiert und läuft auf Sun 3 Workstations [Kittredge, Iordanskaja, Polguère, S. 9]. Das System erstellt (englische) natürlichsprachliche Berichte über die Nutzung eines Computersystems. Diese sollen Administratoren dabei unterstützen, sich schnell über die Systemnutzung und Auslastung einen Überblick zu verschaffen und eventuell sogar Systemmissbrauch aufzudecken und zu belegen [Paiva 1998, S. 14+15]. (Vertiefend wird in Abschnitt 4 auf GOSSIP eingegangen.)

Die Systeme FoG und LFS basieren auf GOSSIP. Sie bringen jedoch mit der Möglichkeit englische und französische Texte zu erzeugen, einen neuen Aspekt - die Multilingualität - mit ein. FoG ist Teil des Systems FPA (Forecast Production Assistant), welches als erstes System [Kahane, S. 31+32] die Generierung von Wettervorhersagetexten bereitstellt [Paiva 1998, S. 12].
LFS (Labour Forcast Statistic) erzeugt aus großen Datenmengen Berichte über Erwerbsstatistiken mit Aspekten wie beispielsweise der Anzahl von Arbeitslosen und deren Veränderung im Vergleich zu vorherigen Monaten [Paiva 1998, S. 16]. Implementiert wurde LFS in Quintus Prolog für Sun 4 Workstation [Iordanskaja uw. 1992, S. 1021]. Ein Beispiel für einen von LFS erzeugten Bericht in Englisch und Französisch ist in Abbildung 2 dargestellt.

JOYCE ist als Teil der Software Design Umgebung Ulysses entwickelt worden. Als integriertes Tool dient JOYCE zur Erstellung verschiedener Textarten. Zum einen können Berichte über sogenannte „flow analyses“ generiert werden. Zum anderen ist die Erzeugung von System-Design-Beschreibungen unterschiedlicher Länge

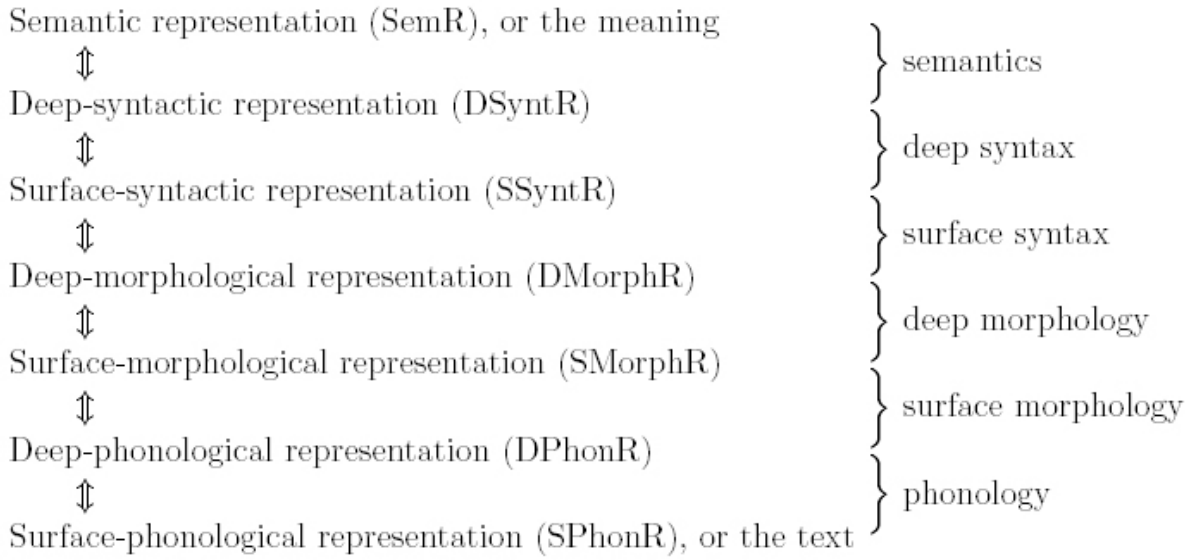

Abbildung 1: Schichten der Repräsentationen und Module der MTT [Kahane, S. 3]

möglich. Diese sollen bei der Systementwicklung als Grundlage zur Erstellung von Dokumentationen im Anschluss an den Design-Prozess dienen [Rambow, Korelsky, S. 40].

Das von LexiQuest entwickelte System LexiGen, früher AlethGen, generiert Briefe zum Beantworten von Reklamationen [Kahane, S31+32].

MultiMeteo ist ein weiteres System zum Generieren von Wetterprognosetexten bzw. Wetterberichten. Es wird im größten europäischen Wettervorhersagezentrum verwendet. Wie bei FoG ist es möglich, multilinguale Wetterberichte zu erstellen. Zusätzlich bietet das System die Möglichkeit, Berichte unterschiedlicher Struktur zu generieren, so dass verschiedene thematische Schwerpunkte gesetzt und die Berichtslänge gewählt werden kann [Kahane, S. 31+32].

Im Gegensatz zu den bisher vorgestellten Systemen ist RealPro ein generisches Realisierer-System. RealPro basiert auf den oben schon beschriebenen Systemen FoG und JOYCE und wurde von CoGenTex entwickelt [Bohnet 2005, S. 15]. Die Erstimplementierung wurde in C++ vorgenommen und brachte eine API in C++ und in Java mit [Lavoie, Rambow, S. 265]. RealPro wurde mehrfach überarbeitet und in Java reimplementiert. Als Realisierer „aus der Konservendose“ ist RealPro sowohl im Bereich der maschinellen Übersetzung als auch im Bereich der Generierung natürlicher Sprache einsetzbar. Die reimplementierte Variante kommt unter anderem in der Übersetzung von Koreanisch nach Englisch, bei der Generierung von Wetterberichten und zum Beschreiben von Objektwelten zum Einsatz [Bohnet 2005, S. 15]. (In Abschnitt 5 findet sich eine tiefer gehende Betrachtung zu RealoPro.)

4 GOSSIP

Im vorhergehenden Abschnitt wurde ein Überblick über verschiedene auf MTT basierende Systeme und deren Anwendung gegeben. Auch GOSSIP wurde schon kurz vorgestellt. In diesem Abschnitt soll GOSSIP nun tiefergehend betrachtet werden. Dazu wird zunächst ein Beispiel eines von GOSSIP generierten Textes vorgestellt und darauf folgend die Architektur beschrieben. Die Funktionsweise von GOSSIP wird daraufhin Schritt für Schritt erläutert. Am Ende dieses Abschnittes wird noch kurz auf das System FoG eingegangen, das auf GOSSIP aufbaut und seinerseits Grundlage für das System RealPro ist.

COMMENTARY

Overview

Estimates for November 1989 from Statistics Canada's Labour Force Survey show that the seasonally adjusted level of employment rose by 32000 and that the level of unemployment increased by 30000. The unemployment rate increased by 0.2 to 7.6.

Employment

For the week ended November 3, 1989, the seasonally adjusted level of employment was estimated at 12568000, up 32000 from October. The increase was concentrated among women aged 25 and over. The employment / population ratio remained virtually unchanged (62.1).

Employment among women aged 25 and over rose by 44000 and their employment / population ratio increased by 0.5 to 52.3.

Employment among men aged 25 and over fell by 12000 and their employment / population ratio decreased by 0.3 to 72.5.

Part-time employment increased by 25000. The increase was evenly distributed between men and women.

Full-time employment remained virtually unchanged. An increase among women was offset by a decrease among men.

The level of employment fell by 10000 in agriculture, by 12000 in transportation, communication and other utilities and by 12000 in primary industries other than agriculture. The level of employment rose by 68000 in services and by 20000 in trade. The level of employment remained virtually unchanged in the other sectors.

The level of employment rose by 11000 in Quebec, by 8000 in Alberta, by 6000 in British Columbia and by 5000 in Ontario. The level of employment remained virtually unchanged in the other sectors.

Unemployment and Participation Rate

The seasonally adjusted level of unemployment was estimated at 1032000 for November 1989, up 30000 from October. The unemployment rate rose by 0.2 to 7.6 and the participation rate increased by 0.3 to 67.2.

The increase in unemployment was concentrated among men aged 25 and over.

Unemployment among men aged 25 and over increased by 24000 while unemployment remained virtually unchanged among women aged 25 and over.

The unemployment rate among men aged 15 to 24 increased by 0.7 to 12.9.

The participation rate among men aged 15 to 24 increased by 0.5 to 73.4 and the participation rate remained virtually unchanged among women aged 15 to 24.

The seasonally adjusted level of unemployment remained virtually unchanged in most provinces. The level of unemployment increased only in Ontario (+ 24000).

COMMENTAIRE

Aperçu

Les estimations tirées de l'enquête de Statistique Canada sur la population active pour novembre 1989 indiquent que le niveau désaisonnalisé de l'emploi a augmenté de 32000 et que le niveau du chômage a augmenté de 30000. Le taux de chômage a augmenté de 0.2 à 7.6.

Emploi

Pour la semaine se terminant le 3 novembre 1989, le niveau désaisonnalisé d'emploi est estimé à 12568000, en hausse de 32000 par rapport à octobre. La hausse a principalement touché les femmes de 25 ans et plus. Le rapport emploi / population n'a pratiquement pas varié (62.1).

L'emploi chez les femmes de 25 ans et plus a augmenté de 44000 et le rapport emploi / population chez celles de 25 ans et plus a augmenté de 0.5 à 52.3.

L'emploi chez les hommes de 25 ans et plus a diminué de 12000 et le rapport emploi / population chez ceux de 25 ans et plus a baissé de 0.3 à 72.5.

L'emploi à temps partiel a augmenté de 25000. La hausse s'était également répartie entre les hommes et les femmes.

L'emploi à temps plein n'a pratiquement pas varié. Une hausse chez les femmes a été compensée par une baisse chez les hommes.

Le niveau d'emploi a diminué de 10000 dans le secteur de l'agriculture, de 12000 dans celui des transports, communications et autres services publics et de 12000 dans les industries primaires autres que l'agriculture. Le niveau d'emploi a augmenté de 68000 dans les industries de services et de 20000 dans le secteur du commerce. Le niveau d'emploi n'a pratiquement pas varié dans les autres secteurs.

Le niveau d'emploi a augmenté de 11000 au Quebec, de 8000 en Alberta, de 6000 en Colombie-Britannique et de 5000 en Ontario. Le niveau d'emploi n'a pratiquement pas varié dans les autres provinces.

Chômage et taux d'activité

Le niveau désaisonnalisé de chômage est estimé à 1032000 pour novembre 1989, en hausse de 30000 par rapport à octobre. Le taux de chômage a augmenté de 0.2 à 7.6 et le taux d'activité a augmenté de 0.3 à 67.2.

La hausse du chômage a principalement touché les hommes de 25 ans et plus.

Le chômage chez les hommes de 25 ans et plus a augmenté de 24000 alors que le chômage n'a pratiquement pas varié chez les femmes de 25 ans et plus.

Le taux de chômage chez les hommes de 15 à 24 ans a augmenté de 0.7 à 12.9.

Le taux d'activité chez les hommes de 15 à 24 ans a augmenté de 0.5 à 73.4 et le taux d'activité n'a pratiquement pas varié chez les femmes de 15 à 24 ans.

Le niveau désaisonnalisé de chômage n'a pratiquement pas varié dans la plupart des provinces. Le niveau de chômage a augmenté seulement en Ontario (+ 24000).

Abbildung 2: Beispiel eines von LFS erzeugten Berichtes in Englisch und Französisch [Iordanskaja uw. 1992, S. 1022]

user id	device id	user activity	files used	starting time	finishing time	elapsed time	cpu time
martin	ttyp0	login	[]	8:20:03	–	–	0
martin	ttyp0	editor	[f1]	8:30:00	9:10:32	0:40:32	240
martin	ttyp1	editor	[f2]	8:42:21	9:13:14	0:30:53	183
martin	ttyp0	logout	[]	9:21:05	–	1:01:02	0
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
jessie	ttyd0	login	[]	11:03:46	–	–	0
jessie	ttyd0	editor	[f5]	11:12:45	12:48:22	1:35:37	573
jessie	ttyd1	compiler	[f4]	11:23:32	11:31:01	0:07:29	300
jessie	ttyd1	editor	[f3]	11:32:25	11:45:56	0:13:31	70
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•

Abbildung 3: Eingabedaten für GOSSIP [Paiva 1998, S. 15]

The system was used for 7 hours 32 minutes 12 seconds. The users of the system ran compilers and editors during this time. The compilers were run six times, for 47% of the cpu-time. The editors were run twelve times, for 53% of the cpu-time. Two users, Jessie and Martin, logged on to the system. Jessie used the system for 63% of the time in use. Martin used the system for 40% of the time in use.

Abbildung 4: Beispiel eines von GOSSIP generierten Textes [Paiva 1998, S. 15]

Wie in Abschnitt 3 schon beschrieben soll GOSSIP Administratoren dabei unterstützen, sich schnell über die Systemnutzung einen Überblick zu verschaffen und eventuell sogar Systemmissbrauch aufzudecken und zu belegen. Um GOSSIP verwenden zu können, ist es notwendig, die Aktivitäten von Benutzern in einer Datenbank aufzuzeichnen. Diese Daten können daraufhin GOSSIP zur Verarbeitung übergeben werden. Wie entsprechende Eingabedaten für GOSSIP aussehen können, ist in Abbildung 3 auszugsweise dargestellt. Ein zu dieser abgebildeten Eingabedatenmenge passender von GOSSIP genertierter Bericht führt Abbildung 4 auf.

GOSSIP ist, wie in Abbildung 5 dargestellt, mittels einer Pipeline-Architektur realisiert. Diese lässt sich in zwei Einheiten unterteilen. Die ersten vier Komponenten bilden den Planungsteil des Systems. In diesem wird bestimmt, welcher Inhalt im Text enthalten sein soll. Die Komponenten ab der semantischen Repräsentation (SemR) bilden den zweiten Teil. Dieser stellt den eigentlichen Realisierer dar und setzt die in Abschnitt 2 vorgestellten Schichten und Module der MTT nahezu eins zu eins um. Im Folgenden sollen nun die Funktionsweise und verschiedene weitere Aspekte von GOSSIP entlang der Module der Pipeline erläutert werden.

Ausgehend von Loggingdaten, wie in Abbildung 3 dargestellt, erzeugt GOSSIP mittels der „Database Component“ einen schematischen „topic-tree“. Dies geschieht, indem verschiedenen The-

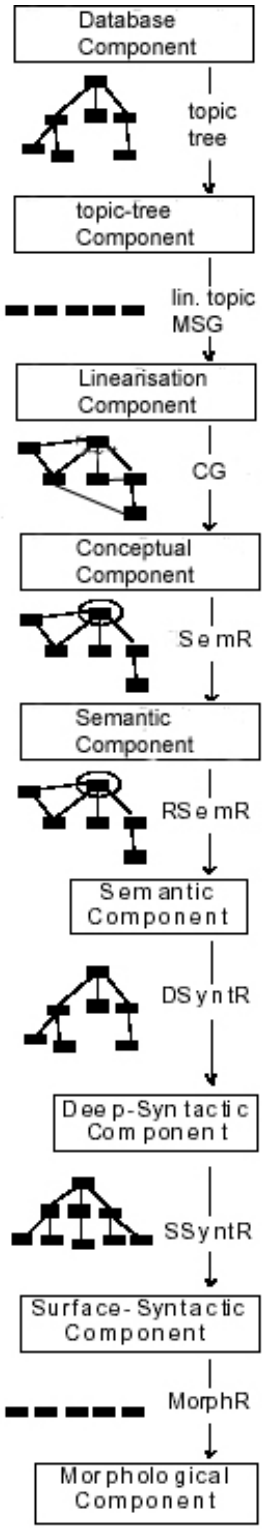

Abbildung 5: Architektur von GOSSIP (in Anlehnung an [Paiva 1998, S. 18])

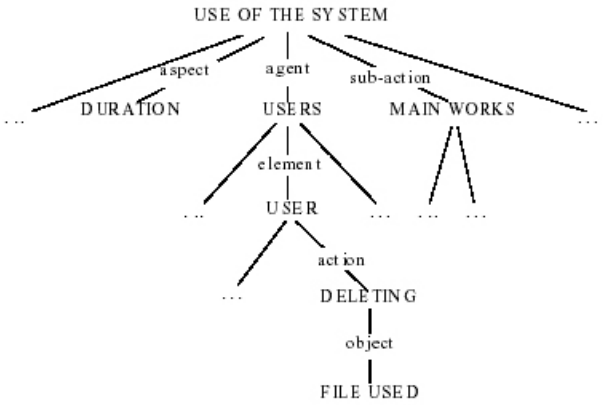

Abbildung 6: Stark verkürzter topic-tree [Paiva 1998, S. 16]

men, wie Benutzer, Tätigkeit, Ausführungsdauer und Ähnlichem, die Fakten aus der Datenbank zugeordnet und dementsprechend in einen Baum einsortiert werden [Paiva 1998, S. 14+15]. Dabei kann die Aufnahme von Fakten auch mehrfach in verschiedene Themen bzw. „topics“ stattfinden. Ein stark verkürzter „topic-tree“ ist in Abbildung 6 dargestellt.

In der „topic-tree Component“ wird der erzeugte Baum bearbeitet. Anhand gleicher Unterbäume werden Gruppierungen vorgenommen [Paiva 1998, S. 14+15]. Außerdem sucht die Komponente Fakten im Baum heraus, die verglichen werden können oder die im Gegensatz zueinander stehen. Auch findet das Heraussuchen von quantifizierbaren und zusammenfassbaren Fakten statt. Nach Auffinden solcher Fakten wird der Baum entsprechend bearbeitet [Paiva 1998, S. 16]. Neben diesen Aktionen gibt es in GOSSIP die Möglichkeit, für jedes Thema bzw. „topic“ spezifische Aktionen zu hinterlegen. Beim Traversieren des Baumes kommen diese beim Erreichen des entsprechenden Themenknotens zur Ausführung. Auf diese Weise können spezielle Aktionen getriggert werden. Beispielsweise kann eine Überprüfung stattfinden, ob die von einem Benutzer gedruckte Datei in dessen Aufgabenbereich fällt [Paiva 1998, S. 15]. Abschließend erzeugt die „topic-tree Component“ aus jeder „topic“ des bearbeiteten Baumes eine Nachricht. Die folgende Komponente bekommt diese schließlich als

sortierte Kette der Nachrichten übergeben.

Die „Linearization Component“ erzeugt vermutlich für jede Nachricht einen „Conceptual Graph“ (CG). (Ob für jede Nachricht oder für die gesamte Nachrichtenkette ein CG erstellt wird, geht aus [Paiva 1998, S. 14+15] nicht eindeutig hervor. Es ist jedoch zu vermuten, dass die CG-Erzeugung pro Nachricht stattfindet, zumal in [Iordanskaja uw. 1992, S. 1021] auf eine entsprechende daraus resultierende satzweise Verarbeitung im auf GOSSIP basierenden System LFS Bezug genommen wird.)
„Conceptual Graphs“ stellen eine Methode zur Wissensrepräsentation aus der KI dar. Diese geht auf Sowa zurück und hält die Möglichkeit bereit, Schlussfolgerungen über den repräsentierten Informationen und Zusammenhänge zu ziehen. In der „Linearization Component“ nutzt man diese Möglichkeit der CG um weitere Zusammenfassungen vorzunehmen. Zusätzlich bringt die „Linearization Component“ Thema-Rhema-Markierungen (siehe „Conceptual Component“) in den CGs an.

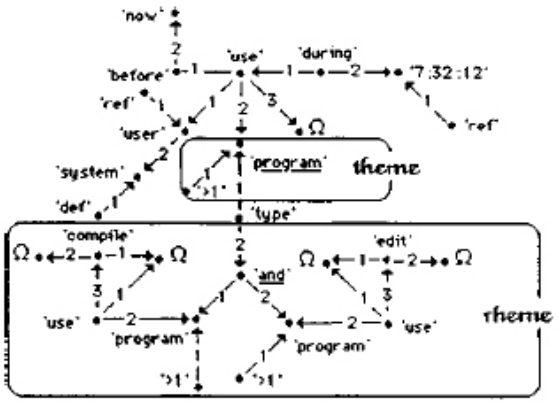

Abbildung 7: Semantische Repräsentation mit Thema-Rhema-Markierung [Kittredge, Iordanskaja, Polguère, S. 10]

Auch in der „Conceptual Component“ wird weiter zusammengefasst und gruppiert. Außerdem wird die Generierung der semantischen Repräsentation (SemR) vorgenommen, wobei eine Übernahme der Thema-Rhema-Markierungen aus den CGs stattfindet. Diese Markierungen beschreiben die Kommunikationsstruktur und sind sehr wichtig, da sie in GOSSIP den einzigen Ansatz bilden, die Auswahlmöglichkeiten in der m-n-Beziehung zwischen Meaning und Text einzuschränken. In Abbildung 7 ist eine semantische Repräsentation des Satzes „The users of the system ran compilers and editors during this time.“ dargestellt. Die Thema-Markierung beinhaltet die schon bekannten Informationen bzw. die Thematik, zu der weitere Informationen – das Rhema – kommuniziert werden sollen. Im Beispiel in Abbildung 7 betrachtet man als bekanntes Thema, dass die Benutzer Programme ausgeführt haben. Das es sich bei den Programmen um Compiler und Editoren handelt stellt die neu zu kommunizierende Informationen im Beispiel dar. Bei normal betonten Sätzen steht im Englischen und Deutschen das Thema am Anfang eines Satzes. Das Rhema steht am Ende eines Satzes. Die Knoten in den semantischen sowie den syntaktischen Repräsentationen werden durch Lexeme gebildet. Mit einem Lexem bezeichnet man eine Einheit des Wortschatzes, welche begriffliche Bedeutung trägt [Bohnet 2005, S17]. Dabei ist zu beachten, dass ein Lexem unabhängig von Kriterien wie Zeit und Anzahl ist. Es stellt also gewissermaßen die Grundform einer begrifflichen Bedeutung dar.

In der ersten „Semantic Component“ werden kommunikationsdominante Knoten markiert. Dies geschieht unter anderem um die Bestimmung des Wurzel-Lexems in der zweiter „Semantic Component“ durch Einschränken der Auswahlmöglichkeiten zu ermöglichen [Kittredge, Iordanskaja, Polguère, S. 9]. Unter Beibehaltung des Informationsgehalts nimmt die erste „Semantic Component“ außerdem Reduzierungen bzw. Verkleinerungen der SemR vor. Durch diesen Vorgang entsteht die RSemR (Reduzierte Semantische Repräsentation). Diese wird in der zweiten „Semantic Component“ nach der Bestimmung des Wurzel-Lexems von der Prädikat-Argument-Struktur (Graph) zu einer Tiefen-Syntaktischen-Relation (einem Baum) überführt [Paiva 1998, S. 16]. In beiden Semantischen Komponenten werden zur Bewerkstelligung der geschilderten Aufgaben unter anderem „lexikalische Funktionen“ verwendet. „Lexikalische Funktionen“ sind abstrakte Abbildungsregeln von einem Lexem auf ein anderes

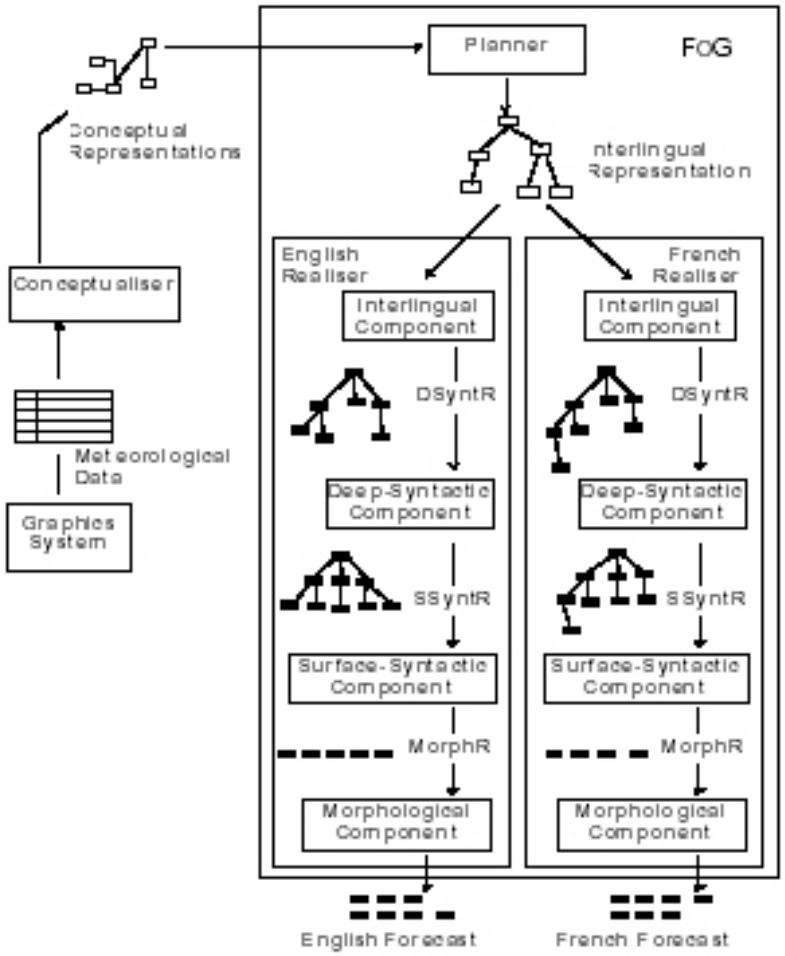

Abbildung 8: Architektur von FoG [Paiva 1998, S. 14]

[Iordanskaja uw. 1992, S. 1021]. Die „lexikalische Funktion“ $A_0(X)$ gibt zum Beispiel ein Adjektiv mit der Bedeutung des Lexems X zurück. So ergibt $A_0(city) = urban$.
$Cont(X)$ liefert den Term mit gegenteiliger Bedeutung von X, beispielsweise $Cont(top) = bottom$ [Paiva 1998, S.17].

Die „Deep-Syntactic Component“ führt anhand der schon beschriebenen Thema-Rhema-Markierungen erste Topicalisierungen für die SSyntR („Sureface-Syntactic Representation“) durch [Kittredge, Iordanskaja, Polguère, S. 9]. Außerdem werden syntaktische Unterbäume, Hilfsverben, Artikel und Präpositionen bestimmt und in den Syntaxbaum eingefügt [Paiva 1998, S. 14+15].

Das Linearisieren der Baumstruktur führt die „Surface-Syntactic Component“ durch. Dieser Prozess bestimmt endgültig die Wortreihenfolge. Wie in den vorigen Komponenten wird auch hierbei wieder die Thema-Rhema-Markierung zur Eingrenzung der Wahlmöglichkeiten herangezogen [Kittredge, Iordanskaja, Polguère, S. 9]. Beim Linearisieren propagiert die „Surface-Syntactic Component“ grammatikalische Informationen von Kopfknoten an Kinder bzw. Dependents. Das ist notwendig, damit in der „Morphological Component“ anhand dieser Informationen die endgültige Wortformbestimmung durch Flektieren vorgenommen werden kann.

Nachdem die Architektur und die Arbeitsweise von GOSSIP erläutert wurden, soll zum Abschluss die-

ses Abschnittes noch kurz auf einige Aspekte von FoG eingegangen werden, da FoG neben JOYCE Grundlage für das im folgenden Abschnitt näher beschriebene System RealPro ist. FoG basiert genauso wie LFS auf GOSSIP (siehe Abschnitt 3). Wie in Abbildung 8 zu erkennen ist, gleicht die Architektur und die Arbeitsweise von FoG der von GOSSIP sehr. Auch FoG weist einen Planungsteil auf. Am Ende von diesem liegt jedoch anstelle einer semantischen Repräsentation eine interlinguale Repräsentation vor. Diese Änderung ermöglicht das Erzeugen von englischen sowie französischen Texten. Die Funktionsweise und Architektur des eigentlichen Realisierers ist von GOSSIP übernommen. Auffälligerweise geschieht dies für jede Sprache. Das ist notwendig, da die Mechanismen bzw. Algorithmen zur Realisierung von den linguistischen Aspekten (den MTM) nicht strikt getrennt sind [Paiva 1998, S. 12-14]. Als Schnittstelle zum Realisierer wird die DSyntR bzw. die „Interlingual Component“ verwendet. Diese setzt die sprachunabhängige Semantik in die Syntax um, welche somit die erste sprachabhängige Repräsentationsebene darstellt.

5 RealPro

Am Ende des letzten Abschnitts wurde kurz auf FoG eingegangen, welches neben JOYCE die Grundlage für das System RealPro darstellt. In diesem Abschnitt soll nun RealPro tiefergehend beschrieben werden. Im Unterschied zu den bisher beschriebenen Systemen ist RealPro ein generischer Realisierer (siehe Abschnitt 3). Deshalb soll zunächst darauf eingegangen werden, wie diese Eigenschaft in der Implementierung erreicht wurde. Daraufhin wird auf den von den Entwicklern von RealPro gewählten Ansatz zur Einschränkung der Auswahlmöglichkeiten in der m-n-Beziehung eingegangen. Am Ende dieses Abschnittes findet außerdem eine Betrachtung einiger Aspekte der Umsetzung von MTM in RealPro statt, wobei mehrere Beispiele von Lexikoneinträgen und Grammatikregeln erläutert werden.

Wie in Abbildung 9 ersichtlich wurden im Unterschied zu FoG (siehe Abschnitt 4) und JOYCE

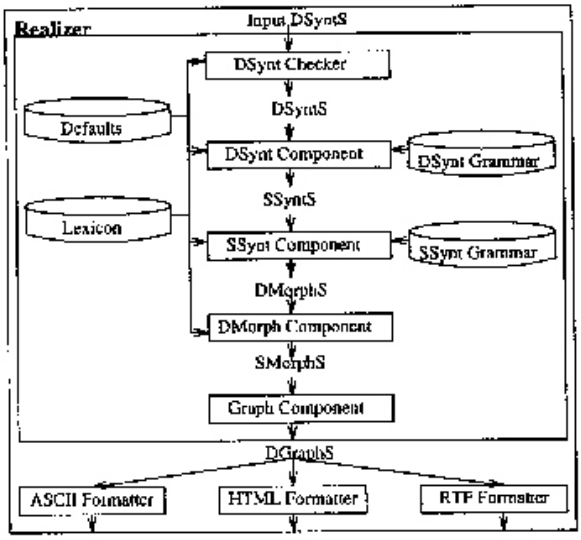

Abbildung 9: Architektur von RealPro [Lavoie, Rambow, S. 267]

bei der Gestaltung der Architektur von RealPro die linguistischen Aspekte aus den Realisiererkomponenten herausgezogen. Durch dieses explizite bilden der MTM mittels der Einheiten „Default“ und „Lexicon“ (siehe Abb. 9 links) sowie den Grammatik-Einheiten (siehe Abb. 9 rechts) wird in RealPro ein Teil der Sprach- und Domänenunabhängigkeit erreicht. Außerdem trägt die Wahl der Schnittstelle mit der DSyntR, wie schon bei FoG, dazu bei [Lavoie uw., S. 60]. Um diese Schnittstelle allgemein anwendbar zu machen, definierte CoGenTex eine auf dem ASCII-Textformat basierende Spezifikationssprache für „Syntactic Dependency Trees“, welche für Englisch in [CoGenTex, Inc. 2000] ausführlich beschrieben und anhand zahlreicher Beispiele illustriert ist.

```
SEE [ question:+ ]
( I boy [ number:pl ]
     ( ATTR  THIS1  )
  II Mary [ class:proper_noun ] )
```

Abbildung 10: Beispiel für DSyntR in RealPro spezifischer ASCII-Form [Lavoie, Rambow, S. 267]

Für den Satz „Do these boys see Mary?“ [Lavoie, Rambow, S. 266] zeigt Abbildung 10 beispielhaft die Beschreibung in der DSyntR-

Spezifikationssprache von RealPro. Argumente werden hinter Lexemen in eckigen Klammern spezifiziert, wie das in Abb. 10 beim Lexem „SEE“ mit dem Argument „question“ zu sehen ist. Dieses Argument bewirkt, dass der Satz zu einer Frage wird. In runden Klammern werden ineinander verschachtelt die Kanten und Knoten der Unterbäume beschrieben. Dabei werden zunächst die Kantenbeschriftungen – in Abb. 10 „I“, „II“, und „ATTR“ – aufgeführt. Darauf folgen dann wieder die jeweiligen Lexeme und deren Attribute.

Im Gegensatz zu FoG und JOYCE wird zur Abbildung der verschiedenen Repräsentationen in RealPro lediglich ein Mechanismus (basierend auf sogenannten Dependenzstrukturen) verwendet. RealPro arbeitet dabei mit einem deklarativen Ansatz. Das bedeutet, dass durch die Regeln des MTM, gebildet mittels des Default, der Grammatiken und dem Lexikon, die Abbildung von der DSyntR auf den Text eindeutig definiert ist. Das ist unter anderem erst durch die Wahl der Schnittstelle auf der Ebene der syntaktischen Bäume möglich. Bedingt durch diese Schnittstellenwahl wird die Problematik, einen Wurzelknoten beim Umwandeln eines Graphen zu einem Baum auswählen zu müssen, ausgeklammert und dem Anwender von RealPro überlassen [Bohnet 2005, S. 18].

Um den deklarativen Ansatz mittels eines Mechanismus umzusetzen, wird in RealPro der jeweils der Ebene entsprechende zu bearbeitende Baum top-down traversiert [Bohnet 2005, S. 17]. Zum Beispiel ein „Surface-Syntactic Tree“, wie in Abbildung 11 gezeigt, welcher den Satz „Mary winning this competition means she can study in Paris and can live with her aunt, whom she adores.“ repräsentiert. Beim Traversieren des Baumes wird für jeden Knoten und jede Kante überprüft, ob eine „passende“ Regel vorhanden ist. Eine Regel ist „passend“, wenn alle Bedingungen der Regel zutreffen. Zur Anwendung der in Abbildung 12 gezeigten DSynt-Regel müsste beispielsweise zutreffen, dass es sich bei dem Lexem X um ein Noun handelt und dieses einen bestimmten Artikel erhalten soll. Ist dies der Fall, wird dem Lexem X der Knoten „THE“ im Baum angehängt. In Abbildung 11 kam eine ähnliche Regel im linken Teilbaum zur Ausführung, welche den Knoten „THIS“ hinzugefügt hat.

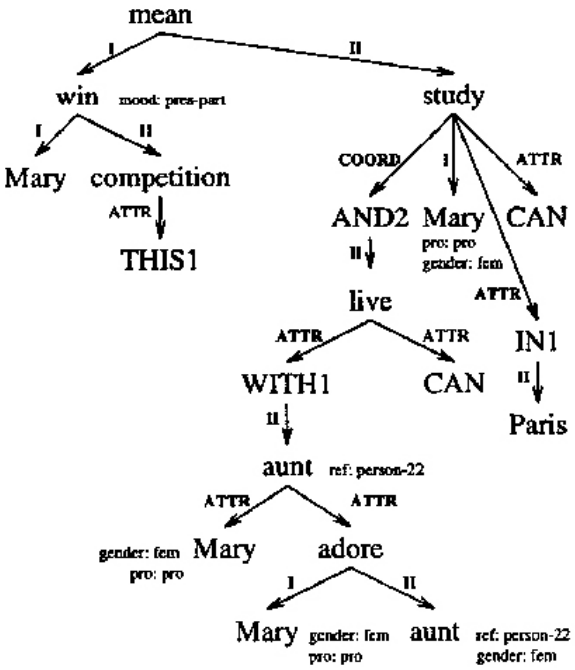

Abbildung 11: Graphische Darstellung eines „Surface Syntactic Tree“ [Lavoie, Rambow, S. 266]

```
DSYNT-RULE:

$X [ class:noun article:def ]
     <-->
$X ( determinative THE )
```

Abbildung 12: Beispiel einer DSynt-Rule [Lavoie uw., S. 64]

Gibt es mehrere passende Regeln, wird die speziellste gewählt. Das heißt, dass die Wahl auf die passende Regel mit der größten Anzahl zutreffender Bedingungen fällt. Weisen mehrere Regeln gleich viele zutreffende Bedingungen auf liegt ein Sonderfall vor. Für solche gibt es in RealPro die Möglichkeit, Regeln eine Reihenfolge zuzuordnen, nach der sie angewendet werden sollen [Bohnet 2005, S. 17]. Durch diesen deklarativen Ansatz wird erreicht, dass RealPro mit einem linearen Aufwand (in Abhängigkeit von der Anzahl der Knoten im Eingabebaum) auskommt [Lavoie, Rambow, S. 267+268]. Die Möglichkeit, Regeln eine Anwendungsreihenfolge zu geben, führt allerdings zu einer Vermischung von

linguistischen und regelhaften Informationen im MTM, insbesondere im Lexikon. Dies erschwert das Übernehmen der MTM in andere Domänen erheblich, da nach dem Abändern und Ergänzen von Einträgen meist die Regeln komplett neu sortiert werden müssen [Bohnet 2005, S. 18].

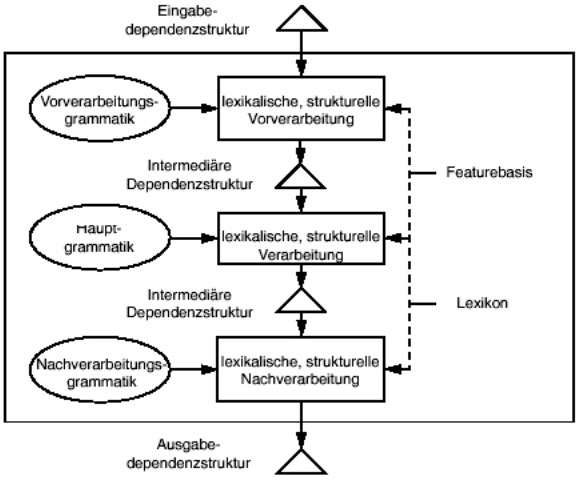

Abbildung 13: Kern-Architektur des Algorithmus von RealPro [Bohnet 2005, S. 15])

```
DEFAULT:  verb [ tense:past mood:ind ]
```

Abbildung 14: Beispiel einer Default-Regel [Lavoie, Rambow, S. 267]

Beim Erläutern der Funktionsweise von RealPro wurde bisher verschwiegen, dass der Algorithmus dreistufig aufgebaut ist, wie in Abbildung 13 gezeigt. Dementsprechend gibt es im Lexikon und in den Grammatiken für jeden Teile – Vor-, Haupt- und Nachverarbeitung – entsprechende Regeln.
In der Vorverarbeitung (siehe Abb. 13 oben) wird die Eingabe der Stufe zur Hauptverarbeitung vorbereitet. So kommen beispielsweise Default-Regeln zum Einsatz, um fehlende Attribute zu ergänzen. Eine entsprechende Default-Regel für RealPro ist in Abbildung 14 abgebildet. Sie bewirkt, wenn nichts Anderes in der Eingabe spezifiziert ist, dass der generierte Text in Vergangenheitsform gebildet wird. Die eigentliche Arbeit leistet die Hauptverarbeitung (Abb. 13 Mitte). Hier kommen Hauptgrammatiken zum Einsatz, wie die in Abbildung 12 gezeigte Grammatikregel oder wie die in Abbildung 15 enthaltene Lexikon-Regel. Die Nachverarbeitung (Abb. 13 unten) passt schließlich die Struktur, wie für die Ausgaberepräsentation benötigt, an [Bohnet 2005, S. 15+16].

Jede Regel, egal ob Grammatik-Regel oder Lexikon-Regel, weist eine Bedingungsseite auf. Diese ist mit einem Doppelpfeil von der Auswirkungsseite getrennt, die die Ergänzungen oder Ersetzungen für die Anwendungsstelle enthält, wie in Abbildung 12 schon sichtbar wurde. Einen Unterschied zwischen Lexikon- und Grammatik-Regel gibt es lediglich beim Gültigkeitsbereich. Grammatik-Regeln haben allgemeine Gültigkeit und können bei jeglichen Lexemen zur Anwendung kommen. Lexikonregeln jedoch gelten nur für ein bestimmtes Lexem. In Abbildung 15 ist beispielhaft ein Lexikoneintrag für das Lexem „SELL“ aufgeführt. Er enthält zwei DSynt-Hauptregeln. Auch sind einige morphologische Regeln enthalten. Ein Eintrag für das Lexem „SEE“ ist in Abbildung 16 als weiteres Beispiel aufgeführt.

6 Ausblick

In dieser Arbeit wurde zunächst die Meaning-Text Theorie anhand ihrer drei Postulate eingeführt. Überblicksartig sind daraufhin einige MTT-basierte Systeme vorgestellt worden. Hierbei wurde deutlich, dass zunächst Systeme im Bereich der maschinellen Übersetzung und später im Bereich der Generierung natürlicher Sprache realisiert worden sind. Mit GOSSIP ist beispielhaft ein System aus dem Bereich der Generierung natürlicher Sprache ausführlich vorgestellt worden.
Aus Systemen des Bereichs Generierung natürlicher Sprache entwickelten sich multilinguale Ansätze und Systeme. Schließlich entstanden generische Systeme, die in beiden Bereichen einsetzbar sind. Stellvertretend für solche wurde RealPro vorgestellt.

Sicherlich kann festgehalten werden, dass die MTT in praxistaugliche Systeme umgesetzt werden kann. Dies bestätigt alleine schon die große Zahl, zum Teil auch kommerziell genutzter Sy-

```
LEXEME:         SELL
CATEGORY:       verb
GOV-PATTERN:    [
                   DSYNT-RULE:
                        SELL ( III $X3 )
                      <-->
                        SELL ( completive2 TO
                                 (prepositional $X3 ) )

                   DSYNT-RULE:
                        SELL ( IV $X4 )
                      <-->
                        SELL ( completive3 FOR
                                 (prepositional $X4 ) )
                ]
MORPHOLOGY:     [
                   ( [ tense:past ]        sold    [ inv ] )
                   ( [ mod:past-part ]     sold    [ inv ] )
                   ( []                    sell    [ reg ] )
                ]
```

Abbildung 15: Lexikoneintrag für das Lexem SELL [Bohnet 2005, S. 17]

```
LEXEME:     SEE
CATEGORY:   verb
MORPHOLOGY: [([mood:past-part] seen [inv] )
             ([tense:past]     saw  [inv] )]
```

Abbildung 16: Lexikoneintrag für das Lexem SEE [Lavoie, Rambow, S. 267]

steme dieser Art. Für die Umsetzung der in der MTT nur vage definierten m-n-Beziehung und der damit verbundenen Auswahlproblematik wurden mit einem dynamischen auf Thema-Rhema-Markierungen basierenden Ansatz bei GOSSIP und einem deklarativen Ansatz bei RealPro zwei unterschiedliche Umsetzungen gezeigt. Dabei ist für den Ansatz von GOSSIP (sowie FoG und LFS) zu beachten, dass keine Trennung der Mechanismen und der MTM vorhanden ist. Deshalb ist ein Übernehmen in eine andere Domäne äußerst schwierig. Bei RealPro wurde ein Teil der Problematik ausgeklammert. Ansonsten liegt sicher ein sehr guter Ansatz vor, bei dem die das MTM bildenden Teile aber ebenfalls nur mit großem Aufwand in andere Domänen übernommen werden können. Obwohl RealPro vielseitig eingesetzt werden kann, ist die Problematik der Domänenunabhängigkeit selbst bei diesem generischen Ansatz noch nicht vollständig gelöst. Auch die Sprachunabhängigkeit ist durch das Ausklammern der Graph-basierten Repräsentationsebenen und dem Transformieren dieser im generischen Ansatz noch zum Teil vorhanden.

Eine Lösung für diese Problematik zu suchen, wäre sehr empfehlenswert. Es würde zu einer starken Vereinfachung der Entwicklung weiterer Systeme führen. Auch wären dann Anwendungen wie sprachunabhängiges Speichern weicher Daten und „just in time“-Generierung der benötigten Sprachvariante vorstellbar.

Literatur

[Bohnet 2005] Bernd W. Bohnet. *Textgenerierung durch Transduktion linguistischer Strukturen*. Fakultät Informatik, Elektrotechnik

und Informationstechnik; Universität Stuttgart, 2005

[CoGenTex, Inc. 2000] *RealPro* General English Grammer - User Manual. CoGenTex Inc, USA 2000

[Iordanskaja uw. 1992] L. Iordanskaja, M. Kim, R. Kittredge, B. Lavoie and A. Polguère. *Generation of Extended Bilingual Statistcal Reports*. CoGenTex Inc. Quebec, 1992 `http://www.cs.mu.oz.au/acl/C/C92/C92-3158.pdf` (Mai 2006)

[Kahane] Sylvain Kahane. *The Meaning-Text Theory*. An International Handbook of Contemporary Research, Berlin: De Gruyter `http://panini.u-paris10.fr/kahane/?u_act=download&dfile=MTT-Handbook2003.pdf` (Mai 2006)

[Kahane 2001] Sylvain Kahane. *What is a Natural Language and How to Describe It?* Meaning-Text Approaches in contrast with Generative Approaches. Université Paris - Invited talk, Computational Linguistics, A. Gelbukh (ed.), 2001, Springer, 1-17.

[Kittredge, Iordanskaja, Polguère] Richard Kittredge, Lidija Iordanskaja, Alain Polguère. *Multi-Lingual Text Generation and Meaning-Text Theory*. Odyssee Reserch Appliquees, Inc. Montreal `http://scholar.google.de/url?sa=U&q=http://webspace.isi.edu/mt-archive/TMI-1988-Kittredge.pdf` (Mai 2006)

[Lavoie, Rambow] Benoit Lavoie, Owen Rambow. *A Fast and Portable Realizer for Text Generation Systems*. CoGenTex Inc, Ithana USA `http://acl.ldc.upenn.edu/A/A97/A97-1039.pdf` (Mai 2006)

[Lavoie uw.] Benoit Lavoie, Richard Kittredge, Tanya Korelsky und Owen Rambow. *A Framework for MT and Multilingual NLG Systems Based on Uniform Lexico-Structural Processing*. CoGenTex, Inc. `http://www.cs.mu.oz.au/acl/A/A00/A00-1009.pdf` (Mai 2006)

[Paiva 1998] Daniel S. Paiva. *A Survey of Applied Natural Language Generation Systems*. University of Brington, 1998 `http://scholar.google.de/url?sa=U&q=http://www.itri.brighton.ac.uk/~Daniel.Paiva/ITRI-98-03.ps.gz` (Mai 2006)

[Rambow, Korelsky] Owen Rambow, Tanya Korelsky. *Applied Text Generation.* `http://www.cs.mu.oz.au/acl/A/A92/A92-1006.pdf` (Mai 2005)