Eine Sprachverstehenskomponente in einem Konstruktionsszenario

Hans Brandt–Pook

Mai 1999

Eine Sprachverstehenskomponente in einem Konstruktionsszenario

Der Technischen Fakultät der
Universität Bielefeld

zur Erlangung des Grades eines

Doktor–Ingenieur

vorgelegt von

Hans Brandt–Pook

Bielefeld — Mai 1999

Danksagung

Diese Arbeit entstand in der Arbeitsgruppe „Angewandte Informatik“ an der Technischen Fakultät der Universität Bielefeld. An erster Stelle bedanke ich mich bei allen Mitgliedern dieser Arbeitsgruppe für die gute Zusammenarbeit, die vielen anregenden Gespräche und das überaus angenehme Arbeitsklima, das geprägt ist von kollegialer Solidarität und kritischer Diskussion.

Gerhard Sagerer als Betreuer hat das Werden der Arbeit in jedem Stadium interessiert begleitet, vorbehaltlos unterstützt und einige Male mit erfrischenden Ideen angekurbelt. Ihm und Dieter Metzing danke ich ferner für die Diskussionen über das Manuskript sowie die Übernahme der Gutachten.

Franz Kummert hatte stets ein offenes Ohr sowohl für grundsätzliche als auch für Detailfragen. Ich danke ihm außerdem für die vielen Anregungen und Hinweise, die er bei der Niederschrift der Arbeit gegeben hat. Durch wertvolle Ratschläge hat mir mein Bürokollege Gernot Fink häufig auf unkomplizierte Art sehr direkt weitergeholfen. Frank Lömker danke ich für die kostbare Zeit, die er mit mir bei der Suche nach Fehlern in meiner Programmierung zugebracht hat. Ich bedanke mich bei Sven Wachsmuth insbesondere für dessen umfangreiche Arbeit bei der Evaluierung des Systems sowie bei Martin Hoffenke, der die Visualisierung von Ergebnissen ermöglichte.

Mein größter Dank geht an Claudia, Jette, Lasse und Linus. Diese Arbeit wäre nicht entstanden ohne ihre Liebe und das wunderbare Leben mit ihnen, das mir immer wieder neue Sichtweisen und Impulse gibt und mir dadurch auch ermöglicht, das eigene Tun richtig einzuordnen.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Nicht das Auge sieht, nicht das Ohr hört, nicht die Hand handelt, nicht das Gehirn denkt und lernt, sondern der ganze Mensch ist es, der jeweils durch das entsprechende Organ sieht, hört, handelt, denkt, lernt.

Hugo Kükelhaus

Die Sprache ist ein natürliches Kommunikationsmittel, das die meisten Menschen von klein auf erlernen und benutzen können. Sie dient dem Austausch von vagen Ideen und spekulativen Ansichten genauso wie der Übermittlung von harten Fakten und belegbaren Informationen. Sprache wird sowohl in dem Gedicht eines die Liebe beschreibenden Dichters als auch in jeder Verordnung von Verwaltungsbehörden verwendet. Sie erreicht den Empfänger auf die unterschiedlichsten Weisen: in Schriftform auf bedrucktem Papier oder dem Bildschirm, per Telefon oder direkt von einem anderen Menschen ausgesprochen. In vielen Fällen wird das sprachlich Formulierte von anderen Kommunikationsmodalitäten begleitet, beispielsweise von Bildern oder Gesten. Die menschliche Sprache ist also ein unter vielen Aspekten unglaublich vielfältiges Phänomen, von dem in dieser Arbeit nur ein kleiner Teil betrachtet werden kann.

In der vorliegenden Arbeit geht es um die automatische Verarbeitung gesprochener Sprache, die sich in mehreren Punkten wesentlich von Sprache in Schriftform unterscheidet:

1. Der naheliegendste Unterschied zwischen geschriebener und gesprochener Sprache ist der, daß letztere ein akustisches Signal erzeugt, das von einem Empfänger — Mensch oder Maschine — „hörbar“ ist. Damit verbunden sind eine Reihe von Phänomenen, die es in einem geschriebenen Text in dieser Form nicht gibt:
   - Unter Umständen erfolgt eine Überlagerung mit anderen Sprachsignalen.

- Störungen bei der Übermittlung des Signals treten auf.
- Es gibt artikulatorische Verschleifungen.
- Dialektale und andere Aussprachevarianten kommen vor, so daß eine Spracheinheit in vielfältigster Weise realisiert werden kann.
- Es liegt ein kontinuierliches Signal ohne Trennungszeichen vor.

Wir Menschen lernen mit solchen Eigenheiten umzugehen. Wir sind in der Lage aus einem gestörten oder von anderen Stimmen überlagerten Sprachsignal den für das Verständnis wichtigen Teil zu isolieren [Han89]. Lücken im akustischen Signal schließen wir durch die Analyse des sprachlichen Kontextes und realisieren dabei oft gar nicht das Fehlen einer sprachlichen Einheit [Fuj86]. Mit artikulatorischen Verschleifungen wie in

„Kannste mir mal ne Mark leihen?“

können wir meist genau so sicher umgehen wie mit zahlreichen Dialekten in einer Sprache. Auch die Gewinnung von kleineren Spracheinheiten wie Silben oder Wörtern aus dem Sprachschall, der selbst keine sichtbaren Diskontinuitäten aufweist, bereitet uns keine Mühe. Für die automatische Sprachverarbeitung sind all diese Eigenschaften gesprochener Sprache große Probleme, deren Bearbeitung schon seit über vierzig Jahren Gegenstand der Forschung ist ([ST95] gibt ab Seite 11 einen kurzen Abriß ihrer Geschichte und Paradigmen). Dabei ergeben sich neben der Suche nach geeigneten Modellierungen und Verfahren, welche die Problemstellung direkt betreffen, auch große Komplexitätsprobleme, weil die anfallenden Datenmengen sehr groß und die kombinatorischen Möglichkeiten bei der Wort– und Satzbildung enorm sind.

2. Ein geschriebener Satz zeichnet sich in der Regel durch grammatikalische Vollständigkeit aus. Bei gesprochener Sprache sollte man darauf nicht hoffen — beispielsweise waren bei den von [Hit88] untersuchten Äußerungen 26 Prozent bezüglich der Grammatik für die geschriebene Sprache unvollständig. Eine Äußerung wie

   „Einmal Pommes rot weiß bitte.“

   enthält weder Subjekt noch Verb. Gleichwohl ist die Intention des Sprechenden für jeden Adressaten verständlich[1] und somit das Ziel der Kommunikation erreicht. Soll gesprochene Sprache robust automatisch verarbeitet werden können, muß also Vorsorge für den Fall getroffen werden, daß eine Äußerung zwar semantisch eindeutig aber syntaktisch unvollständig ist.

[1] Eine kritische Korrekturleserin hat mich darauf hingewiesen, daß diese Behauptung wohl nur für den westfälischen Sprachraum uneingeschränkt haltbar ist.

3. Eine weiteres Merkmal gesprochener Sprache ist die Existenz von Satzabbrüchen und Reparaturen [Lev83]. Oftmals sind einfache Häsitationen die Ankündigung dieser Konstruktionen:

   „Ich wohne jetzt in Nieder– äh Oberjöllenbeck."

   Mitunter bekommen aber auch Wörter die Funktion einer Reparaturmarkierung, die in anderen Kontexten eine völlig andere syntaktische und semantische Kategorie besitzen, wie folgende Äußerung[2] belegt:

   „Käse, ich mein' natürlich März."

   Für die automatische Sprachverarbeitung stellt das Vorkommen von Reparaturen in gesprochener Sprache insofern ein Problem da, als daß sie zunächst aufzuspüren sind und dann der richtige Bezug gefunden werden muß. Im System muß also modelliert sein, daß das Wort „Käse" eine Reparatur einleiten kann und während der Abarbeitung der Äußerung muß analysiert werden, welchen Teil des bisher Gesprochenen die folgenden Wörter korrigieren.
4. Prosodische Informationen, die vor allem durch die Tonhöhe, die Lautheit und zeitliche Strukturierung einer Äußerung ausgedrückt werden, spielen in der zwischenmenschlichen Kommunikation eine große Rolle [Nöt89]. Sie dienen sowohl der Übermittlung emotionaler Zustände wie Freude oder Ärger als auch der Hervorhebung des wesentlichen in einer Äußerung sowie der Gliederung des Gesprochenen. In der geschriebenen Sprache werden einige dieser Funktionen von der Interpunktion übernommen. Durch ihre Beachtung können Menschen beispielsweise den Unterschied zwischen

   „Am Dienstag geht es nicht um zehn Uhr. Am Mittwoch würde es mir passen."

   und

   „Am Dienstag geht es nicht. Um zehn Uhr am Mittwoch würde es mir passen."

   erkennen. Leider ist die automatische Gewinnung von prosodischen Merkmalen aus einer Äußerung und ihre richtige Interpretation ein bisher erst in Ansätzen gelöstes Problem. Daher wird bei der maschinellen Interpretation einer Äußerung häufig auf diese wichtige Informationsquelle verzichtet.

Die Triebfeder für die Forschung auf dem Gebiet der automatischen Sprachverarbeitung ist die Mensch–Maschine–Kommunikation. Es ist naheliegend, die Sprache als natürliches Kommunikationsmittel zwischen den Menschen auch zur Interaktion mit einem Rechner zu benutzen. Somit können auf einfache Art und Weise Rechner und ihre Programme bedient, Maschinen

[2] Äußerung n002k032 aus dem Verbmobil–Korpus (siehe Abschnitt 2.1)

gesteuert oder Einträge in Datenbanken abgefragt werden. Gesprochene Sprache als Form der Mensch–Maschine–Kommunikation ermöglicht es dem Menschen die Hände frei zu haben für andere Tätigkeiten. Sie erfordert außer einem Mikrofon keine weiteren Eingabegeräte[3] — mit modernen Telekommunikationsgeräten ist sogar eine Interaktion über sehr große räumliche Entfernungen möglich. Außerdem ist eine sprachliche Eingabe nicht wie die Interaktion per Tastatur und Bildschirm von einer Beleuchtung abhängig.

Besonders reizvoll für eine einfache Kommunikation mit einem Rechner ist die Nutzung mehrerer Eingabemodule je nach den gegebenen Möglichkeiten und Erfordernissen. Auf die Integration traditioneller Eingabegeräte wie Tastatur oder Maus, die eine möglichst intuitive Bildschirmoberfläche manipulieren können, sollte dabei ebensowenig verzichtet werden wie auf ein sprachverstehendes Modul. Auch Informationen, die in Videosequenzen oder Infrarotbildern verborgen liegen, können wertvolle Hilfen bei der Mensch–Maschine–Kommunikation liefern. Sie können sowohl Aspekte des Diskursbereichs analysieren und somit beispielsweise Ambiguitäten auflösen als auch direkt zur Eingabe genutzt werden, zum Beispiel durch eine Auswertung von menschlichen Gesten.

In der automatischen Sprachverarbeitung unterscheidet man zwischen der Sprach*erkennung* und dem Sprach*verstehen*. Die Erkennungsaufgabe besteht darin, aus einem akustischen Sprachsignal eine symbolische Repräsentation zu gewinnen, die beispielsweise als Folge von Wörtern ausgegeben wird. Die Sprachverstehenskomponente versucht, den Sinn des Gesprochenen zu bestimmen, also eine Interpretation vorzunehmen. Im folgenden werden diese Begriffe in diesem Sinne verwendet.

Betrachtet man die Aufgabenspezifikation von sprachverarbeitenden Systemen, so lassen sich verschiedene Typen feststellen:

Kommandosysteme: Kommandosysteme sind darauf ausgelegt einige isoliert artikulierte Wörter, zum Beispiel „ja“, „nein“ oder die Ziffern, sicher zu erkennen und daraufhin eine Aktion auszuführen. Erste Ergebnisse, die noch sprecherabhängig erzielt wurden, finden sich bereits in [Dav52].

Diktiersysteme: Die Systeme, mit deren Hilfe ein gesprochener Text von einem Rechner erfaßt wird, heißen Diktiersysteme. Das bestechende an diesen Systemen ist die Größe des zugrunde liegenden Wortschatzes (über 50.000 Wörter), der meist noch während der Arbeit mit dem System erweitert werden kann [Kuh99]. Allerdings wird dem Benutzer ein Training des Systems abverlangt. Erst seit etwa zwei Jahren ist es möglich, auf kurze Pausen zwischen den einzelnen Wörtern zu verzichten [Mal98].

[3] In neueren Forschungsarbeiten geht man von einer Sprachaufnahme mit Raum– oder Funkmikrofonen aus, welche die Bewegungsmöglichkeit eines Sprechers in keiner Weise mehr einschränken [Wah97].

Auskunftssysteme: Auskunftssysteme sind sprecherunabhängig und verarbeiten meist kontinuierlich gesprochene Spontansprache, die oftmals auch nur in Telefonqualität vorliegt. Auf eine Anfrage hin erhält ein Benutzer eine Auskunft, die in der Regel vom System durch eine Datenbankabfrage gewonnen wird. Der Wortschatz von Auskunftssystemen liegt bei einigen tausend Wörtern. Neuere Anwendungen sind zum Beispiel in [Sei97, Bar97, Wan98] beschrieben. In Kapitel 2 wird auch ein im Rahmen dieser Arbeit fertiggestelltes Auskunftssystem vorgestellt.

Integrierte Systeme: Sprachverarbeitungssysteme, die in ein komplexeres Gesamtsystem eingebunden sind, nenne ich *integrierte Systeme*. Im Unterschied zu Auskunftssystemen kann eine adäquate Leistung von integrierten Systemen nur erbracht werden, wenn sie in hohem Maße mit anderen Modulen des Gesamtsystems kooperieren. Insbesondere ist eine Planungskomponente erforderlich, weil die Aufgabe des Gesamtsystems so komplex ist, daß eine reine Datenbankabfrage zur Lösung des Problems in der Regel nicht ausreicht. Integrierte System müssen in der Lage sein, mit anderen Eingabegeräten und Modulen zur Interpretation anderer Sensordaten zu interagieren. Stehen einem Benutzer nämlich mehrere Kommunikationskanäle zur Verfügung, werden nicht alle Intentionen sprachlich formuliert. Vielmehr wird in der sprachlichen Beschreibung einer Aufgabe auch Bezug genommen auf die anderen Kommunikationskanäle („Du siehst doch den roten Würfel da.“) oder die von ihnen bereitgestellte Information wird implizit vorausgesetzt („Jetzt leg’ den mal auf den anderen.“). Ein weitere Anforderung an integrierte Systeme besteht darin, daß sie sich der Verarbeitungsgeschwindigkeit des Gesamtsystems anpassen müssen. Ein Beispiel für ein solches System wird in dem in Kapitel 2 beschriebenen TRAINS-Projekt entwickelt.

Während Kommando– und Diktiersysteme dem Benutzer außer der Ausführung keine Rückmeldung darüber geben, was sie verstanden haben, treten Auskunfts– und integrierte Systeme in einen Dialog mit dem Benutzer.

Die vorliegende Arbeit hat folgendes Ziel: Es wird die Entwicklung der sprachverstehenden Komponente für ein komplexes System beschrieben, in welchem ein Roboter Konstruktionsaufgaben nach den Anweisungen eines menschlichen Instrukteurs ausführen kann. Dazu wird vorhandenes Sprachmaterial aus einer simulierten Mensch–Maschine–Kommunikation zur Modellbildung analysiert. In der Arbeit wird ein integriertes System vorgestellt, das sprecherunabhängige kontinuierliche Spontansprache verarbeitet und die sprachlichen Anweisungen im Kontext des Dialogs und anderer Informationen über die Szenerie interpretiert. Um die Leistungsfähigkeit des gesamten Konstruktionssystems nicht zu schmälern, spielt bei der Entwicklung der Sprachverstehenskomponente eine effiziente Verarbeitung der Äußerungen eine große Rolle.

Die Arbeit ist wie folgt aufgebaut. Im folgenden Kapitel 2 wird ein Überblick über aktuelle Dialogsysteme gegeben. Stellvertretend für verschiedene Entwicklungsansätze werden drei sprachverstehende Systeme vorgestellt. Kapitel 3 beinhaltet Untersuchungen für die Modellbildung. Es wird die Domäne, in der die Sprachverstehenskomponente arbeitet, vorgestellt. Anhand eines Korpus werden Spezifikationen für die Modelle, welche die Grundlage für die Realisierung bilden, entwickelt. Eine wichtige Entscheidung bei der weiteren Entwicklung von automatischen Sprachverstehenssystemen ist sodann die Wahl des Formalismus. In Kapitel 4 stelle ich darum die von mir gewählte Wissensrepräsentationssprache ERNEST^{++} vor. Damit sind alle Grundlagen gelegt, um im anschließenden Kapitel 5 das implementierte System ausführlich darzulegen. Dazu werden seine Stellung im gesamten Konstruktionssystem veranschaulicht sowie seine einzelnen Module betrachtet und die Verarbeitungsstrategie vorgestellt. In Kapitel 6 evaluiere ich die Verstehenskomponente unter verschiedenen Gesichtspunkten, bevor Kapitel 7 die Arbeit zusammenfaßt.

Kapitel 2

Dialogsysteme

Eine Maschine kann so wenig Informationen produzieren wie ein Elektrizitätswerk Energie erzeugt.

Joseph Weizenbaum

Die Aufgabe von Dialogsystemen besteht darin, die Kommunikation mit einem Rechner für die Menschen zu vereinfachen. Ein Dialogsystem ist ein System, das alle Aspekte der sprachlichen Interaktion mit dem Rechner bearbeitet. Dazu gehören die Spracherkennung und das Sprachverstehen der einzelnen Äußerungen sowie die Möglichkeit, sich in einer Äußerung auf zuvor Gesprochenes zu beziehen. Da der Rechner in immer mehr Bereichen des Lebens der Menschen eingesetzt wird, liegt in der adäquaten Gestaltung dieser Form der Mensch–Maschine–Interaktion eine große Herausforderung. Während in den Anfängen des automatischen Sprachverstehens vor allem Kommandosysteme entwickelt wurden, liegt der Schwerpunkt der Forschung seit einigen Jahren auf Dialogsystemen, die kontinuierlich gesprochene Spontansprache verarbeiten können. Dabei haben die meisten Forscher auf diesem Gebiet nicht den Anspruch, auf dem Rechner solche Algorithmen zu realisieren, die das menschliche Sprachverstehen nachbilden. Vielmehr steht im Vordergrund, geeignete Verfahren zu finden, welche die verschiedenen Rollen der an der Mensch–Maschine–Kommunikation Beteiligten berücksichtigen. In der Einleitung von [Fer96] ist dieser Aspekt sehr prägnant formuliert:

> „The guiding principle of our approach is that human–computer interaction should be treated as a *dialogue* between the participants, each of whom brings different skills and objectives to the conversation. … The result of this approach is truly *mixed-initiative* interaction.“

Es gibt inzwischen unzählig viele Publikationen zu Dialogsystemen mit ganz unterschiedlichen Anwendungen und Entwicklungsgrundsätzen, deren Beschreibung den Rahmen dieser Arbeit

sprengte. Maier beobachtet drei mögliche Herangehensweisen bei der Entwicklung von Dialogsystemen [Mai97]:

1. Orientierung an den Anforderungen der Systembenutzer und an der Aufgabenstellung
2. Ausnutzung von Modellen zur Mensch–Mensch–Kommunikation
3. Gebrauch von Modellen, die aus einer simulierten Mensch–Maschine–Kommunikation gewonnen werden.

Sicherlich kann man nicht immer eine harte Trennlinie zwischen den drei Möglichkeiten ziehen. Dennoch bieten sie einen Ansatz zur Differenzierung und Vergleichbarkeit. Ich werde in diesem Kapitel beispielhaft für diese verschiedenen Grundlagen der Entwicklung drei Dialogsysteme vorstellen. In *TRAINS-96* wurde die erste Herangehensweise gewählt. *Verbmobil* und dem System zur Terminabsprache lagen Modelle zur Mensch–Mensch–Kommunikation zugrunde.

2.1 Verbmobil

Das Verbmobil–Projekt ist ein sehr großes Projekt unter Beteiligung von Universitäten und Industrieunternehmen, das viele Bereiche der automatischen Sprachverarbeitung bearbeitet. Ziel des langfristigen Projektes ist die Entwicklung eines Systems zur automatischen Übersetzung einer Äußerung in eine Fremdsprache. Dazu muß eine spontansprachliche Eingabe erkannt, interpretiert, übersetzt und schließlich in der Zielsprache synthetisiert werden. Als Szenario für die hier beschriebene erste Phase des Projektes wird angenommen, daß ein Deutscher und ein Japaner in englischer Sprache kommunizieren, die sie beide mindestens passiv beherrschen. Ziel des Dialogs ist die Vereinbarung eines gemeinsamen geschäftlichen Termins. Sollte die Aushandlung aufgrund von mangelnden aktiven Sprachkenntnissen ins Stocken geraten, kann sich jeder Gesprächspartner in seiner Muttersprache an Verbmobil wenden, das dann die Äußerung ins Englische übersetzt und so den Fortgang des Dialogs sichert.

Das Verbmobil–Projekt hat auf dem Gebiet der deutschen automatischen Sprachverarbeitungs–Forschung eine Leitfunktion. Eine umfassende Darstellung aller Aspekte von Verbmobil ist kaum möglich — eine sehr anschauliche Gesamtdarstellung des Projektes findet sich in [Wah97]. Nach einem Architekturüberblick werde ich mich auf die Beschreibung des automatischen Sprachverstehens einer deutschen Äußerung beschränken, weil das auch der Gegenstand meiner Arbeit ist.

Architektur

Die funktionale Architektur von Verbmobil ist eine nicht–hierarchische Multiagentenarchitektur. Verbmobil besitzt kein zentrales Kontrollmodul. Vielmehr bearbeiten autonome Module in sich

geschlossene Teilaufgaben (beispielsweise Spracherkennung, syntaktisch–semantische Analyse, Transfer oder Synthese) und kommunizieren bei Bedarf mit anderen Modulen mittels der zu diesem Zweck entwickelten Kommunikationsumgebung *ICE (Intarc Communication Environment)* [Amt96]. Abbildung 2.1 zeigt die wesentlichen Module und Kommunikationswege, die im Verbmobil–Forschungsprototypen 1.0 verwirklicht sind.

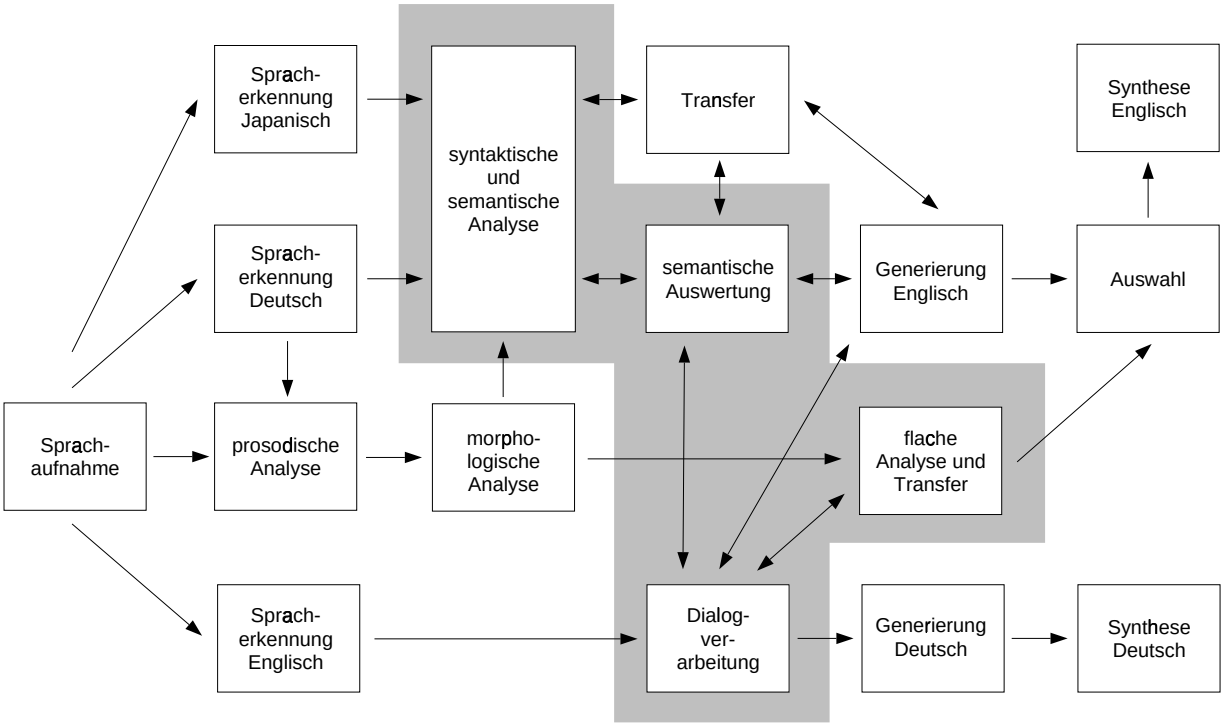

Abbildung 2.1: Funktionale Architektur von Verbmobil (nach [Wah97] und [Bub96])

Die Aufgabe des Aufnahme–Moduls besteht darin, die Sprachsignale aufzunehmen und zu digitalisieren. Auf seine Ergebnisse greifen die Spracherkennungsmodule und das Prosodiemodul zu.

Verbmobil arbeitet in zwei Modi. Solange die Verständigung der beiden Beteiligten klappt, befindet sich Verbmobil im sogenannten *listening mode*. In diesem Modus ist der Spracherkenner für englische Eingaben aktiv und sucht nach Schlüsselwörtern, die vom dialogverarbeitenden Modul zur Bestimmung von Dialogakten benutzt werden. Auf diese Weise ist das System stets auf dem laufenden über den Inhalt des Gesprächs. Dies ist die Grundlage dafür, im *interpreting mode* richtig agieren zu können. In diesem Modus, der von einem Gesprächsteilnehmer per Knopfdruck angefordert wird, leistet Verbmobil die konkrete Hilfe im Dialog.

Im interpreting mode sind die Spracherkenner für deutsche beziehungsweise japanische Eingaben zugeschaltet, deren Ergebnis bewertete Worthypothesengraphen sind. Die Resultate des deutschen Spracherkenners — er deckt im Forschungsprototypen einen Wortschatz von 2461 Wörtern ab — werden von dem Prosodie–Modul mit prosodischer Information, wie zum Bei-

spiel Satzgrenzen, annotiert. Durch die anschließende morphologischen Analyse kann der Worthypothesengraph möglicherweise ausgedünnt werden. Die Module zur Syntax– und Semantikanalyse, zur semantischen Auswertung und zur flachen Analyse erledigen zusammen mit der Dialogverarbeitung das Sprachverstehen. Sie sind in Abbildung 2.1 grau unterlegt und werden im nächsten Abschnitt erläutert. Das Transfer–Modul bildet die ermittelte Bedeutung des Gesprochenen auf semantische Strukturen in der Zielsprache ab, welche die Grundlage für die Generierung eines korrekten englischen Satzes bilden, der schließlich synthetisiert wird.

Verbmobil kann auch deutsche Anfragen zum Stand des Dialogs auf deutsch beantworten. Zur Ausführung solcher Klärungsdialoge stehen die Module zur Generierung und Synthese deutscher Sprache zur Verfügung.

Sprachverstehen in Verbmobil

Verbmobil ist hochgradig nebenläufig und verarbeitet jede Äußerung gleichzeitig nach unterschiedlichen Paradigmen. Bei der linguistisch fundierten, sehr aufwendigen sogenannten *tiefen Analyse* werden Syntax und Semantik nicht sequentiell sondern verschränkt untersucht. Dazu werden vom Modul zur syntaktisch–semantischen Analyse aus dem Worthypothesengraphen grammatisch korrekte Wortfolgen extrahiert und auf ihre syntaktischen und semantischen Lesarten hin untersucht. Die auf Unifikationsgrammatiken und kompositioneller Semantikrepräsentation basierende Analyse liefert einen *Verbmobil Interface Term (VIT)* [Bos96]. Ein VIT ist ein zehnelementiger Term, in dem alle während der Verarbeitung einer Äußerung gewonnenen Informationen gespeichert werden — er stellt somit die zentrale Datenstruktur für die sprachverstehenden Module dar. Dem Modul zur semantischen Auswertung kommt die Aufgabe zu, die mehr oder weniger spezifizierten Ausdrücke der semantischen Repräsentation zusätzlich so weit zu disambiguieren, daß die Regeln des Transfermoduls anwendbar sind. Dazu liefert der Dialogkontext, der im Modul zur Dialogverarbeitung gespeichert ist, wichtige Informationen.

Die sogenannte *flache Verarbeitungsstrategie* liegt vor allem darin begründet, daß bei der Formulierung und Erkennung von Spontansprache Fehler gemacht werden. Infolgedessen gibt es häufig keinen syntaktisch korrekten Pfad im Worthypothesengraphen, und die oben beschriebenen Module zur tiefen Analyse können keine Interpretation liefern. In dem Modul zur flachen Analyse und zum Transfer werden darum parallel zwei einfache approximative Übersetzungen erzeugt, die im Dialogkontext oftmals ausreichen. Diese beruhen auf schematischen beziehungsweise auf dialogakt–basierten Übersetzungsverfahren.

Das Auswahl–Modul entscheidet jeweils, welches Ergebnis schließlich synthetisiert wird. In diese Entscheidung fließt nicht nur die Qualität der Übersetzung, sondern beispielsweise auch das Laufzeitverhalten des Systems ein. Es ist auch möglich, daß Lücken in der tiefen Analyse mit Ergebnissen der flachen Übersetzung geschlossen werden. Mit diesem hybriden Übersetzungsansatz erreichte der Verbmobil–Forschungsprototyp 1.0 eine „approximativ korrekte“ [Wah97,

Seite 55] Übersetzungsrate von 74,2 Prozent, das heißt, daß in fast drei von vier Übersetzungen der vom Sprecher intendierte Inhalt der Äußerung erkannt und in der Zielsprache verständlich erzeugt wurde. Das Ergebnis wurde auf einer Testmenge von über 20.000 Äußerungen erzielt, die keine dem Spracherkenner unbekannten Wörter enthielten. Leider finden sich keine aussagekräftigen Angaben über das Laufzeitverhalten von Verbmobil während der Evaluation.

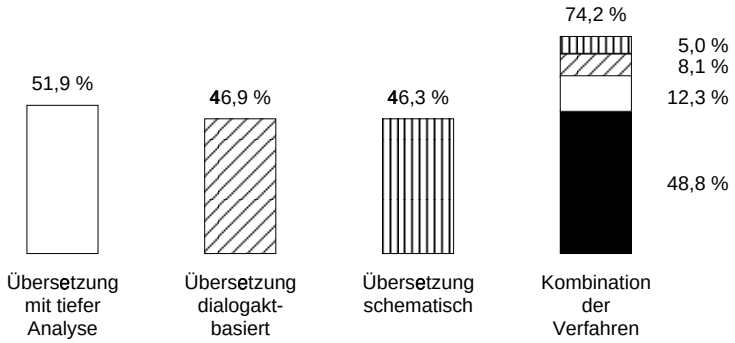

Abbildung 2.2: Ergebnisse der Verbmobil End–zu–End Evaluation (nach [Bub97])

Abbildung 2.2 veranschaulicht, wie sich die Kombination der Verarbeitungsstrategien auf die Testergebnisse überaus positiv auswirkte. 51,9 Prozent der Übersetzungen waren mit Hilfe der tiefen Verarbeitungsstrategie erfolgreich. Die beiden Übersetzungsstrategien, die auf einer flachen Analyse basierten, erreichten Erfolgsquoten von 46,9 beziehungsweise 46,3 Prozent. In 48,8 Prozent der Testfälle lieferten mindestens zwei Verfahren eine richtige Übersetzung; in 12,3 Prozent war es nur die tiefe Verarbeitungsstrategie, die befriedigend arbeitete. Folglich lieferte in 13,1 Prozent der Äußerungen nur die flache Analyse eine im genannten Sinne korrekte Übersetzung. Die Ergebnisse unterstreichen die postulierte Leistungsfähigkeit von Verbmobil und bestätigen die Bedeutung des Projektes für die Entwicklung der Forschungsvorhaben zur automatischen Sprachverarbeitung.

2.2 TRAINS

Die Forschungsarbeiten im TRAINS–Projekt [All95b, All96, Fer96, Ste97] umfassen nicht nur Aspekte der automatischen Sprachverarbeitung, sondern auch Arbeiten zum unsicheren Schließen und zur Planung, sowie zur Integration von multimodalen Benutzereingaben. In dem langfristig angelegten Projekt an der University of Rochester wird in mehreren Stufen ein System entwickelt, in dem ein besonderes Gewicht auf die bidirektionale Gestaltung der Mensch–Maschine–Interaktion gelegt wird. Das Ziel ist die Realisierung eines *mixed–initiative Planning Assistent*. Die weitere Beschreibung des Projektes in diesem Abschnitt bezieht sich stets auf das TRAINS-96 System, also den Entwicklungsstand am Ende des Jahres 1996.

TRAINS-96 hilft einem Benutzer, der als *human manager* bezeichnet wird, bei der Planung einer Route in einer Transportdomäne. Dazu wird dem Manager eine Karte präsentiert, auf der einige Städte markiert sind. Die Aufgabe besteht darin, den Transport einiger Maschinen zwischen abgebildeten Städten zu erarbeiten. Um einen angeregten Dialog zu erzwingen, sind in das Szenario Transportschwierigkeiten (zum Beispiel aufkommende Unwetter oder Verkehrsprobleme) eingebaut, die unter Umständen eine Veränderung der ursprünglich geplanten Route erfordern. TRAINS-96 akzeptiert eine gesprochene, getippte oder graphische Eingabe und benutzt als Ausgabegeräte ein Sprachsynthesegerät und ein graphisches Display.

Die Abbildung 2.3 zeigt die wesentlichen Systemteile. Das Spracherkennungsmodul basiert auf dem *SPHINX–II* Spracherkenner [Hua93], der an die verwendete Kommunikationssprache *KQLM (Knowledge Query and Manipulation Language)* [Fin94] angepaßt wurde.

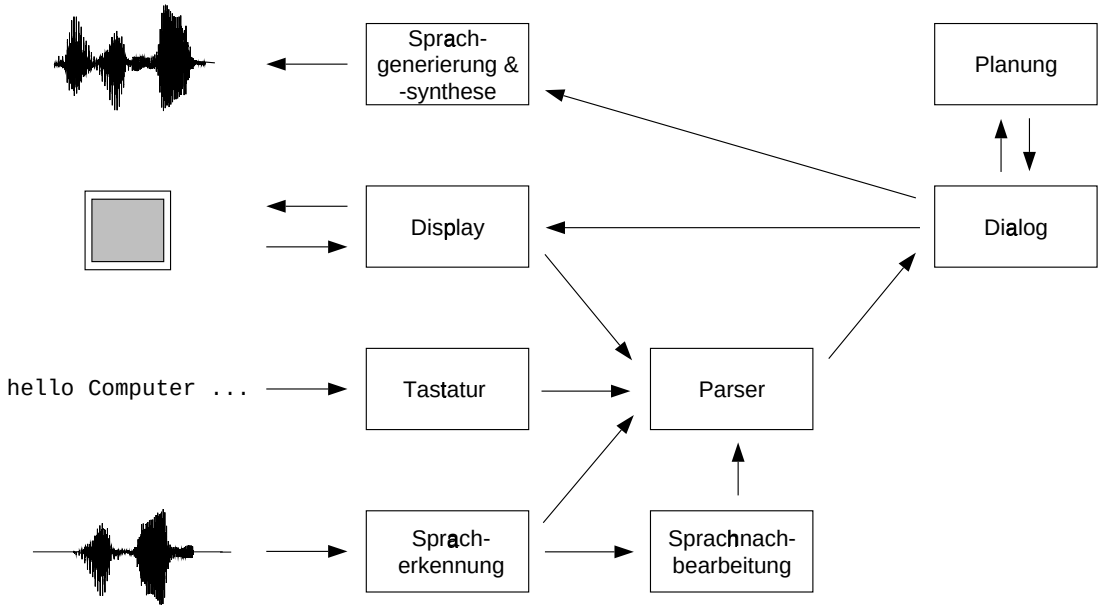

Abbildung 2.3: Wesentliche Module von TRAINS-96

Aufgabe der Sprachnachbearbeitung [Rin96] ist die Korrektur von Spracherkennungsfehlern. Dazu wird ein Kanalmodell angenommen, das Artikulationsvarianten in dem Gesprochenen, aufnahmebedingte akustische Verzerrungen und Fehlleistungen des Spracherkenners modelliert. Das Kanalmodell wird automatisch trainiert, indem die Ausgaben des Erkenners mit den transkribierten Referenzäußerungen verglichen und die Vertauschungen, Einfügungen und Löschungen des Erkenners protokolliert werden. Auf dieser Grundlage werden während der Analyse die Worthypothesenketten des Spracherkenners überarbeitet.

Das TRAINS–Projekt betrachtet alle Aktionen und Äußerungen des Benutzers als sprachliche Aktionen (*linguistic actions*). Aus diesem Grund werden alle Benutzereingaben (gesprochene Sprache, getippte und graphische Eingabe) vom Parser interpretiert. Liegt dem Parser ein Ergebnis des Sprachnachbearbeitung–Moduls vor, so wird die Ausgabe des Spracherkenners ignoriert.

Eine getippte Eingabe, die das Tastatur–Modul zur Weiterverarbeitung aufbereitet, behandelt der Parser genauso wie gesprochene Sprache. Die Verarbeitung von mausbasierten graphischen Benutzereingaben ist in der Realisierung von TRAINS-96 noch rudimentär und Gegenstand aktueller Arbeiten. Der Parser generiert aus diesen multimodalen Eingaben eine logische Repräsentation als erste Stufe des Interpretationsprozesses. Das Ergebnis des Parsings ist daher nicht eine syntaktische Analyse der Eingabe, sondern eine Sequenz von Sprechakten, welche die Aussage charakterisieren. In [Fer96] wird folgendes Beispiel angeführt. Von der Äußerung

„okay now lets take the last train and go from Albany to Milwaukee“

hat der Spracherkenner

„okay now I take the last train in go from Albany to is“

erkannt. Der Parser generiert daraus die Sprechakte CONFIRM („okay“), TELL („now I take the last train“) und REQUEST („go from Albany“), wobei der Sprechakt TELL eine allgemeine Äußerung beschreibt, deren propositionaler Gehalt nicht bestimmt werden konnte. Der Parser hat also nicht eine komplette Interpretation liefern können. Gleichwohl konnte er soviel extrahieren und in Sprechakten repräsentieren, daß der Dialog sinnvoll fortgesetzt werden konnte.

Das Dialog–Modul ist verantwortlich für die Interpretation einer Äußerung und die daraus resultierende Systemreaktion. Dazu analysiert es die in der Ausgabe des Parsers enthaltene Information und ergänzt sie mit Wissen, das sich aus dem bisherigen Dialogverlauf ergeben hat. Auch der aktuelle Dialogstatus wird berücksichtigt, um die richtige Interpretation und Systemreaktion zu bestimmen. Das Dialog–Modul organisiert außerdem die Systemausgaben des Display– und des Sprachgenerierungsmoduls. Abbildung 2.4 zeigt eine dargebotene Karte während eines Dialogs. Nach jeder Interpretation ändert sich die Karte durch zusätzlich eingezeichnete Routen. Die im Diskurs bereits besprochenen Routen sind hell dargestellt. Somit hat ein Benutzer eine sehr anschauliche Repräsentation des im Dialog Erarbeiteten.

Falls problembezogene Äußerungen wie zum Beispiel ein Routenvorschlag vorliegen, muß das Planungs–Modul angesprochen und dessen Ergebnis in die Interpretation des Dialog–Moduls integriert werden. Das Planungs–Modul besitzt sowohl eine Komponente zur Planung von Routen, als auch eine Komponente, welche sich die bisherige Interaktion und die bisher benannten Städte und Routen merkt. So ist es möglich, daß der Benutzer verschiedene Szenarien durchspielen, vergleichen beziehungsweise verändern und Bezug auf vorhergehende Szenarien nehmen kann.

Sechzehn Personen hatten die Aufgabe, jeweils fünf komplexe Transportaufgaben mit TRAINS-96 zu planen. Sie wurden durch ein kurzes Tutorium und die Präsentation einiger Übungssätze in die Interaktionsmöglichkeiten mit dem System eingewiesen und waren angehalten, zügig zu arbeiten. Die Interaktion mit der Tastatur oder der Maus sollte möglichst nur dann gewählt werden, wenn die sprachliche Kommunikation als mißglückt empfunden wurde.

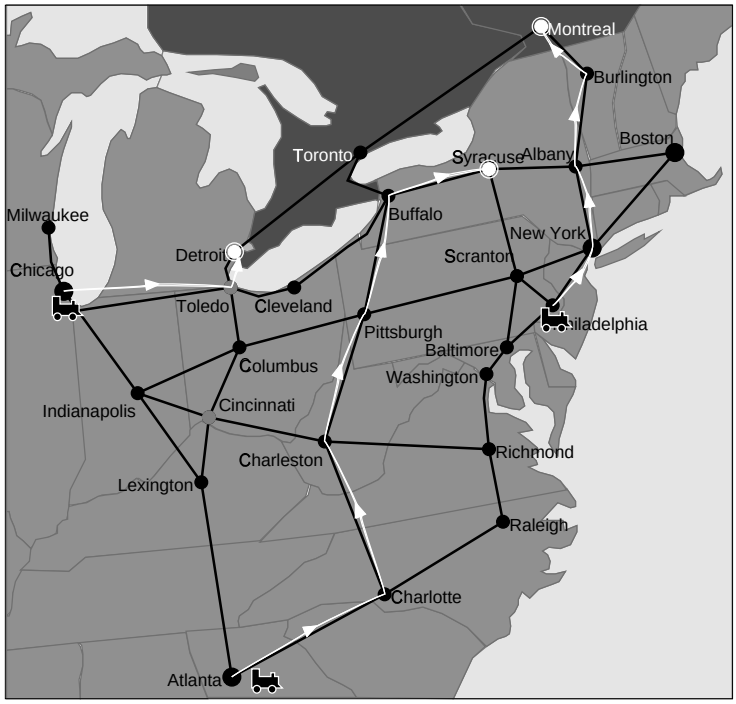

Abbildung 2.4: In TRAINS-96 verwendete Karten (aus [Fer96])

Der erste Dialog mit dem System ging nicht in die Ergebnisse ein — er wurde als Trainingsdialog betrachtet. Tabelle 2.1 zeigt die erzielten Ergebnisse. „Erfolgreich“ war ein Dialog, wenn die gestellte Aufgabe erfüllt wurde. In fünf Dialogen stürzte das System während der Analyse ab. Es zeigte sich, daß es besonders schwer war, einen Dialog dann erfolgreich zu beenden, wenn eine bereits eingeschlagene Route wegen aufkommender Hindernisse verändert werden sollte. Die Testpersonen füllten nach jeder Aufgabe einen Fragebogen zur Qualität von TRAINS-96 aus. Dabei erhielten die sprachverarbeitenden Systemleistungen eine deutlich schlechtere Bewertung als die Planungsfähigkeit des Systems. Daher schlägt [Ste97] vor, die flexiblere Routengestaltung als einen Schwerpunkt der zukünftigen Arbeit an dem System zu sehen und die Daten der Dialoge in das Training des Sprachnachbearbeitung–Moduls einfließen zu lassen.

Dialoge	erfolgreich	erfolglos	abgebrochen
64	42	17	5
100 %	65,6 %	26,6 %	7,8 %

Tabelle 2.1: Evaluationsergebnisse von TRAINS-96

2.3 Terminabsprachesystem

In diesem Abschnitt stelle ich ein Dialogsystem vor, das die semantische Netzwerksprache ERNEST (siehe Kapitel 4) benutzt. Da Teile des sprachverstehenden Moduls dieses Systems unabhängig von einem Anwendungsbereich modelliert sind, finden sie auch Verwendung in anderen ERNEST–Dialogsystemen [Mas94, Kum98b]. Insofern beschreibt die folgende Darstellung nicht nur ein spezielles Dialogsystem, sondern die bisherige Art und Weise der Modellierung der auf ERNEST basierenden Sprachverstehenssysteme.

Die Aufgabe des Dialogsystems besteht darin, mit einem Benutzer einen Termin zu vereinbaren. Dazu wird dem System ein Terminkalender bereitgestellt, in dem bestimmte Zeiten als belegt und andere als mögliche Terminzeiten gekennzeichnet sind. Ein Benutzer kann dem System nun spontansprachlich einen Terminvorschlag machen. Das System bittet gegebenenfalls um eine Konkretisierung (falls der Vorschlag nicht präzise genug war) oder lehnt den Vorschlag ab (falls in dem internen Terminkalender der Zeitbereich als belegt markiert ist). Dieses Wechselspiel wiederholt sich so lange, bis schließlich ein gemeinsamer Termin gefunden wurde. Abbildung 2.5 zeigt einen typischen Dialog.

System 1: „Guten Tag! Sie können mit mir einen Termin vereinbaren. Bitte machen Sie einen Vorschlag!“

Benutzer 1: „Wie wäre es denn Mittwoch nachmittags um vier?“

System 2: „Nein, am Mittwoch um sechzehn Uhr geht es leider nicht. Ich kann am Mittwoch zwischen acht Uhr und zwölf Uhr dreißig. Wann paßt es Ihnen während dieser Zeit?“

Benutzer 2: „Nein, dann lieber am Freitag.“

System 3: „Ja, Freitag sieht es gut aus. Wann paßt es Ihnen denn an diesem Tag? Bei mir ginge es von acht Uhr bis dreizehn Uhr und von siebzehn Uhr dreißig bis zwanzig Uhr“

Benutzer 3: „Es ginge vormittags um zehn.“

System 4: „Ihr Vorschlag paßt mir sehr gut. Bitte bestätigen Sie noch einmal den Termin. Ist es richtig, daß wir uns Freitag um zehn Uhr treffen?“

Benutzer 4: „Ja. Das ist okay.“

System 5: „In Ordnung, also dann bis Freitag um zehn.“

Abbildung 2.5: Beispiel für einen Dialog mit dem Terminabsprachesystem

Im Rahmen der Promotion wurde dieses System, das in einer ersten Vorabversion vorlag, zur Lauffähigkeit gebracht. Erste Ergebnisse wurden bereits publiziert [BP96, Fis98]. Hauptsächlich

habe ich an der Wissensbasis gearbeitet — daher liegt der Schwerpunkt der folgenden Darstellung auch auf diesem Teil des Systems.

2.3.1 Systembeschreibung

Die Sprachverstehenskomponente gewinnt aus dem Ergebnis der Spracherkennung eine Interpretation und steuert den Dialog mit einem Benutzer des Systems. Ihre wesentlichen Bestandteile sind eine Wissensbasis, in der das deklarative und prozedurale Wissen über den Problemkreis abgelegt ist und eine eng mit der Spracherkennung interagierende Analysestrategie.

Wissensbasis der Sprachverstehenskomponente

Die Wissensbasis des Terminabsprachesystems enthält sowohl linguistisches als auch domänenspezifisches Wissen, also Informationen darüber wie üblicherweise ein Termin ausgehandelt wird. Abbildung 2.6 gibt einen Überblick über den Aufbau des Wissensbasis.

Die *Hypothesenebene* stellt in den auf ERNEST beruhenden Netzen die Schnittstelle zur Spracherkennung dar. Die vom vorgelagerten spracherkennenden Prozeß erzeugten Worthypothesen werden entgegengenommen und an den im sprachverstehenden Modul benutzten Formalismus angepaßt. Wie später gezeigt wird, ist im Terminabsprachesystem allerdings ein Verfahren eingesetzt, das Worthypothesen auch an andere Stellen des semantischen Netzes einbinden kann. Somit verliert diese Ebene während der Analyse eines Sprachsignals ihren klassischen Schnittstellencharakter.

Auf der *Syntaxebene* werden die Wortarten der Hypothesen bestimmt und daraus syntaktische Konstituenten gewonnen. Darüber hinaus ermöglicht der in [Hil95] beschriebene und im System verwandte Ansatz eine besondere Behandlung der Zeitkonstituenten. Das Wissen darüber wie Zeitangaben formuliert werden, läßt nämlich eine sehr spezielle Modellierung dieser sprachlichen Äußerungen zu. Es ist exakt modelliert, welche Ausprägungen die verschiedenen Zeitangaben haben können. Folglich wird die Hypothesenfolge „der erste Sonntag im Mai“ als Tagesangabe und nicht als allgemeine Nominalgruppe weiterverarbeitet.

Die Modellierung der *semantischen Ebene* fußt auf der Tiefenkasus–Theorie von Fillmore [Fil68]. Sie besagt, daß ein Verb in einer bestimmten Bedeutung Leerstellen, die *Tiefenkasus*, eröffnet. Dies möchte ich am Beispiel des Verbs „ausmachen“ verdeutlichen, indem ich zwei Bedeutungen des Verbs betrachte. „Ausmachen“ kann bedeuten etwas zu beenden:

„In einer halben Stunde machst du aber den Fernseher aus!“ (2.1)

Dieses Verb kann aber auch im Sinne von „verabreden“ benutzt werden:

„Ich hatte mit dir doch ein Rendezvous für Montag ausgemacht.“ (2.2)

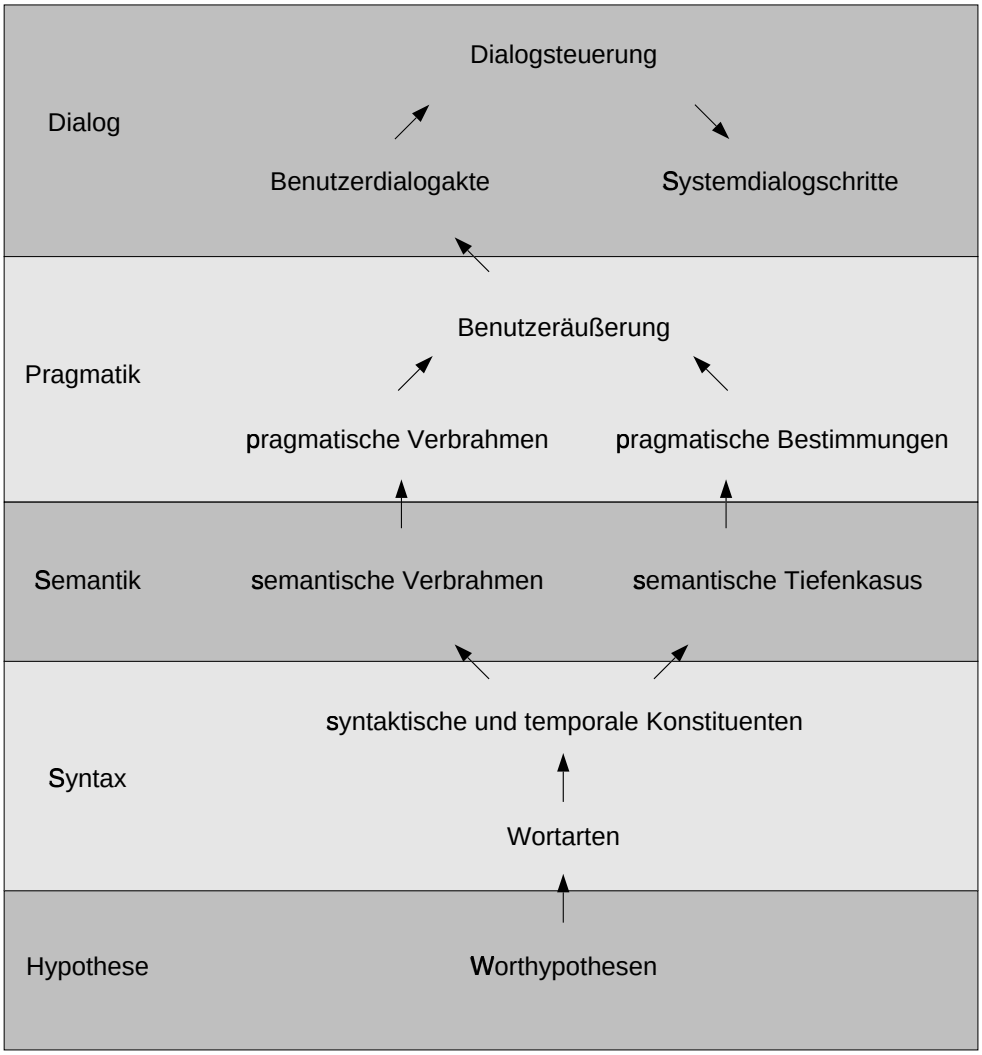

Abbildung 2.6: Überblick über die Wissensbasis des Terminabsprachesystems

Das Verb „ausmachen“ eröffnet im Terminabsprachesystem daher die folgenden vier Tiefenkasus:

Tiefenkasus	Bedeutung	Vorkommen in 2.1 und 2.2
S_AGENT	wer ist der Akteur?	„du“, „ich“
S_EXPERIENCER	wem geschieht etwas?	„dir“
S_OBJECT	was wird ausgemacht?	„den Fernseher“, „ein Rendezvous“
S_TIME	wann geschieht etwas?	„in einer halben Stunde“, „für Montag“

Tabelle 2.2: Semantischer Verbrahmen für „ausmachen“

Nicht alle dieser Tiefenkasus sind obligatorisch, wie die Äußerung „Ich mache den Fernseher aus.“ zeigt. Sie ist durchaus vollständig und sinnvoll, obwohl sie neben dem Verb nur S_AGENT („ich“) und S_OBJECT („den Fernseher“) enthält. Insgesamt sind im Terminabsprachesystem

23 solche *semantische Verbrahmen* für Verben aus der zugrunde liegenden Domäne und 13 Tiefenkasus realisiert.

Die semantischen Verbrahmen und Tiefenkasus finden ihre Entsprechung in den *pragmatischen Verbrahmen* und *pragmatischen Bestimmungen* auf der nächst höheren Ebene der Wissensbasis. Diese Modelle schränken die allgemeine Modellierung der Semantikebene auf das Anwendungsgebiet des Dialogsystems ein. Während auf der Semantikebene beispielsweise der Tiefenkasus S_OBJECT die Frage „*Was* wird ausgemacht“ beantwortet, bezieht sich die entsprechende pragmatische Bestimmung P_OBJECT im Terminabsprachesystem auf die Frage „*Was für eine Art von Treffen* wird ausgemacht?“.

Pragmatische Verbrahmen und Bestimmungen repräsentieren daher den propositionalen Gehalt einer Äußerung in der Domäne. Sie werden zusammengefaßt zu einem Modell für eine komplette Äußerung eines Systembenutzers während eines Dialogs. Somit sind auf dieser Ebene die Modelle für die Interpretation einer Einzeläußerung gegeben.

Schon an dem folgenden kurzen Dialog

> „Wann wollen wir denn ‘Titanic’ sehen?“
>
> „Laß uns doch den Dienstag nehmen.“
>
> „Und welche Vorstellung?“
>
> „Die um acht wäre in Ordnung.“

wird klar, daß in einem Dialog Kontextinformation häufig unabdingbar ist, um eine Einzeläußerung richtig zu verstehen. Die Antwort „Die um acht wäre in Ordnung.“ ist nur deshalb befriedigend, weil vorher schon geklärt war, an welchem Tag der Kinobesuch stattfinden soll. Diese Antwort erhält also den Charakter einer Präzisierung eines zuvor unterbreiteten Vorschlages. Auf der *Dialogebene* des Terminabsprachesystems finden sich daher Modelle für verschiedene Dialogakte des Benutzers, wie zum Beispiel Modelle für einen Terminvorschlag, eine Präzisierung eines Vorschlages oder auch die Ablehnung beziehungsweise Bestätigung eines Termins. Die Dialogsteuerung beinhaltet ein Dialoggedächtnis, mit dessen Hilfe auch Informationen, die über mehrere Äußerungen verteilt vorliegen, zu einem konsistenten Terminwunsch des Benutzers zusammengefaßt und mögliche Ambiguitäten aufgelöst werden. Das obige Beispiel zeigt, daß auch allgemeines Wissen erforderlich ist, um aus Gesprochenem Gemeintes zu erschließen: üblicherweise finden morgens keine Kinovorstellungen statt und daher drückt „um acht“ wahrscheinlich aus, daß der Sprecher gern um zwanzig Uhr ins Kino möchte. In der Domäne Terminabsprache gilt ähnliches: die Zeitkonstituente in dem Vorschlag „um drei Uhr wäre es gut“ wird daher zu fünfzehn Uhr aufgelöst, falls keine weitere Spezifikation vorliegt.

Die Dialogsteuerung entscheidet auch darüber, welches der jeweils angemessene Systemdialogschritt ist. Dabei liegt das in Abbildung 2.7 dargestellte Dialogmodell zugrunde.

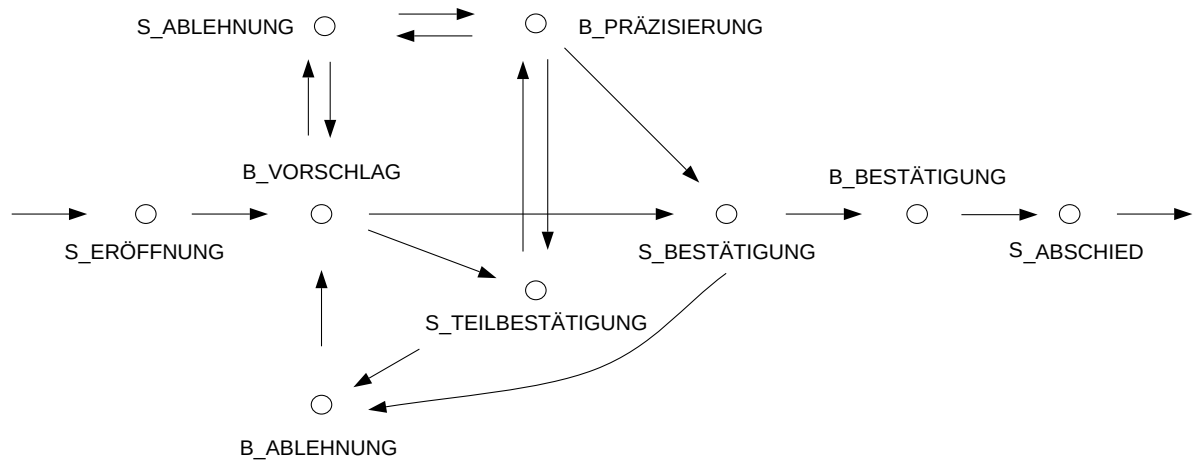

Abbildung 2.7: Dialogmodell des Terminabsprachesystems

Der Dialog beginnt mit einer Eröffnung des Systems (S_ERÖFFNUNG), die einen Terminvorschlag des Benutzers (B_VORSCHLAG) erbittet. Dieser initiale Vorschlag kann vom System abgelehnt, bestätigt oder teilbestätigt werden, wobei der Systemdialogschritt S_TEILBESTÄTIGUNG (dritte Systemausgabe in Abbildung 2.5) dann gewählt wird, wenn ein Vorschlag zwar aufgrund des internen Terminkalenders des Systems prinzipiell möglich, aber noch nicht konkret genug ist. In diesem Fall wird vom Benutzer eine Präzisierung seines Vorschlages (B_PRÄZISIERUNG) erwartet. Wenn der Termin schließlich fixiert ist, bestätigt das System ihn noch einmal (S_BESTÄTIGUNG) und erfragt eine abschließende Bestätigung des Benutzers (B_BESTÄTIGUNG). Somit ist dem Benutzer vor dem Abschluß des Dialogs noch einmal die Gelegenheit gegeben, eventuelle Fehlleistungen des Systems durch eine Ablehnung zu korrigieren (B_ABLEHNUNG) und erneut einen Vorschlag zu machen. Dieses Verfahren erhöht die Zuverlässigkeit des Systems enorm.

Analyseverfahren

Das Terminabsprachesystem benutzt ein integriertes Verfahren zur Erkennung und Interpretation der spontansprachlichen Eingabe. Eine ausführliche Darstellung dieses Verfahrens findet sich in [Fin95a]. Die grundlegende Idee besteht darin, die Strukturen der Wissensbasis auf eine Hierarchie von *Hidden–Markov–Modellen (HMMs)* [Hua90] abzubilden, so daß das verwendete Spracherkennungssystem ISADORA [ST91] in der Lage ist, *strukturierte* Erkennungshypothesen an die Verstehenskomponente zu liefern.

Dazu werden in einem automatischen Verfahren die modellierten linguistischen Konstituenten soweit dekomponiert, bis schließlich nur noch elementare Modelle vorliegen. Die in den linguistischen Konstituenten formulierten Restriktionen werden dabei in die entstehenden Substrukturen propagiert. Der so entstandene Baum von Modellen wird auf einen komplexen Baum von HMMs transformiert. Die auf diese Weise entstandenen strukturierten HMMs, die sogenann-

ten *semantischen Hidden–Markov Netzwerke (SHMNs)*, repräsentieren sowohl linguistisches als auch akustisch–phonetisches Wissen. Die strukturelle Äquivalenz des HMM-Baumes und der linguistischen Wissensbasis macht einen Transfer der Erkennungsergebnisse zu der Verstehenskomponente sehr einfach. Es ist damit möglich, komplette wohlstrukturierte Instanzen von linguistischen Konstituenten *direkt* in den Verstehensprozeß zu integrieren — ein Großteil der aufwendigen linguistischen Analyse des Gesagten wird also in die Sprach*erkennung* verlagert.

Die Verwendung von SHMNs läßt eine sehr flexible Steuerung der Interpretation zu. Zu Beginn einer Äußerung werden alle SHMNs aktiviert, weil ja völlig unbekannt ist, was der Benutzer sagen wird. Der Spracherkenner liefert darauf strukturierte Ergebnisse, deren Bedeutung sodann extrahiert wird. Im Verlauf der Äußerung können immer weitreichendere restriktive Annahmen über den Fortgang des Gesprochenen gemacht werden. Daher werden dann nur die SHMNs von der Kontrolle der Verstehenskomponente aktiviert, die zu der Interpretation des bisher Gesprochenen passen. Enthält die Äußerung beispielsweise schon eine Instanz vom Tiefenkasus S_AGENT, so wird das dazu gehörende SHMN nicht mehr aktiviert, da in aller Regel eine Äußerung in diesem Szenario nur genau einen Agenten enthält. Es entwickelt sich also ein Wechselspiel zwischen der Prädiktion von SHMNs, der Spracherkennung, der Transformation ihrer Ergebnisse in die Verstehenskomponente und der Fortsetzung der Interpretation.

2.3.2 Ergebnisse

Das System wurde von sieben Sprechern und drei Sprecherinnen getestet, die mit dem System nicht vertraut waren. Sie erhielten lediglich die Information, daß mit dem System ein Termin vereinbart werden kann. Anhand von vorgegebenen Terminkalendern sollte jeder Testteilnehmer drei Termine vereinbaren. Die Systemausgabe wurde den Testpersonen ebenso visuell dargeboten wie das Ergebnis des Erkennungsprozesses, welches die Grundlage der linguistischen Interpretation gewesen war. Bei der Bewertung des Systems spielten interne Daten (zum Beispiel die korrekte interne Repräsentation einer Zeitangabe) keine Rolle. Vielmehr wurde das System als „Black Box“ betrachtet und die Systemausgabe als Maßstab für die Interpretationsleistung genommen. Somit wird die Leistung des Systems insgesamt auf seine Benutzeradäquatheit und Korrektheit analysiert. Begegnet das System der Äußerung

> „Ich möchte einen Termin für Dienstag nachmittag.“

zum Beispiel mit der Ausgabe

> „Nein, am Mittwoch habe ich keine Zeit. Bitte machen Sie einen neuen Vorschlag“

liegt offensichtlich eine Fehlleistung vor, selbst wenn das System intern die Konstituente „einen Termin“ richtig analysiert hätte.

Äußerungen	vollständig interpret.	partiell interpret.	falsch interpret.	nicht interpret
145	70	21	18	36
100 %	48,3 %	14,5 %	12,4 %	24,8 %

Tabelle 2.3: Systemleistung bezogen auf die Interpretation einzelner Äußerungen

Zunächst wurde bewertet, ob eine einzelne Äußerung des Benutzers korrekt interpretiert wurde (siehe Tabelle 2.3). Kontextwissen, welches das System aus dem bisherigen Verlauf des Dialogs hätte schließen müssen, blieb bei dieser Auswertung unberücksichtigt. Eine Interpretation war „vollständig“, falls der gesamte propositionale Gehalt des Gesagten vom System bestimmt und angemessen reagiert wurde. Falls nicht alle Informationen, die in der Äußerung enthalten waren, analysiert werden konnten, wurde das Ergebnis mit „partiell interpretiert“ bezeichnet. „falsch“ war eine Systemleistung dann, wenn die Ausgabe eine in keiner Weise plausible Reaktion auf die Äußerung darstellte. Forderte das System den Benutzer auf, das zuletzt Gesprochene noch einmal zu wiederholen, wurde diese Leistung mit „nicht interpretiert“ bewertet. Häufig war eine extrem schlechte Sprach*erkennung* die Ursache für einen solchen Fehlschlag. Die Tabelle zeigt, daß 62,8 Prozent der Äußerungen ganz oder teilweise korrekt interpretiert wurden. Das war eine ausreichende Grundlage dafür, viele Dialoge erfolgreich zu gestalten (siehe Tabelle 2.4).

Dialoge	erfolgreich	erfolglos
30	28	2
100 %	93,3 %	6,6 %

Tabelle 2.4: Systemleistung bezogen auf den gesamten Dialogverlauf

Die Systemleistung, bezogen auf den gesamten Dialogverlauf, wurde danach bewertet, ob schließlich ein Termin vereinbart werden konnte. Solche Dialoge wurden als „erfolgreich“ eingestuft. Nur zwei der 30 Dialoge erreichten dieses Ziel nicht. Die Testpersonen beurteilten die Dialogführung als angemessen, was sich auch in der geringen Anzahl der Dialogschritte widerspiegelte: durchschnittlich benötigten die Benutzer nur fünf Dialogschritte, um ihr Ziel zu erreichen. Das System wurde in keiner Weise bezüglich der Laufzeit optimiert — daher bemängelten einige der Testpersonen zu Recht die auftretenden Wartezeiten auf eine Systemreaktion.

In einer weiteren Testreihe wurde untersucht, inwieweit Arbeiten über Diskurspartikeln [Sch87, Sch95a] in das System integriert werden können [Fis98]. Diskurspartikeln bergen oftmals wichtige Informationen, wie beispielhaft an der Partikel „ja“ gezeigt werden soll. In der Äußerung

„Ja, guten Tag. Ich hätte gern einen Termin.“

signalisiert „ja“ dem Gesprächspartner die Übernahme der Initiative durch den Sprecher — „ja“ hat eine *Redeübernahme*–Funktion. Die Partikel kann auch die gegenteilige Funktion erfüllen. In

„Wir sehen uns doch am Sonntag, ja?“

hat „ja“ eine *Redeübergabe*–Funktion und fordert den Gegenüber zur Fortsetzung des Gespräches auf. In

„Wir könnten ja auch Montag nehmen.“

realisiert „ja“ eine Abtönung des Vorschlages, hat also eine *modale* Funktion. Im letzten Beispiel schließlich wird die *Antwort*–Funktion von „ja“ sichtbar:

„Ja, am Dienstag um zehn Uhr paßt es mir.“

Im Terminabsprachesystem werden auf der semantischen Ebene von der Funktion *berPartikel-Funktion* die Diskursfunktionen der Partikeln anhand ihrer Stellung in der Äußerung und des Dialogkontextes ermittelt — die Regel für „ja“ ist in Algorithmus 2.1 zu sehen. Insgesamt

◇ Funktion: berPartikelFunktion
▷ Parameter: ein Wort *W*
▷ Parameter: Typ der letzten Systemausgabe *l_sys_ausg_typ*
▷ Parameter: Stellung *pos* von *W* in der Äußerung
◁ Rückgabe: Diskursfunktion von *W*

```
...
if (W == „ja“) then
    if (pos == initial) then
        if (l_sys_ausg_typ == S_BESTÄTIGUNG ∨
            l_sys_ausg_typ == S_TEILBESTÄTIGUNG) then
                diskurs_funktion := Antwort;
        else
                diskurs_funktion := Redeübernahme;
    elseif (pos == medial) then
        diskurs_funktion := Modal;
    else    // position == final
        diskurs_funktion := Redeübergabe;
...
return diskurs_funktion;
```

Algorithmus 2.1: Regel zur Bestimmung der Diskursfunktion von „ja“

wurden Regeln für 87 verschiedene Diskurspartikeln integriert. Die erzielten ersten Ergebnisse belegen, daß dieses Verfahren häufig ausreicht, um Diskurspartikeln korrekt klassifizieren zu können. Die Teststichprobe bestand aus 40 per Tastatur eingegebenen Äußerungen, die insgesamt 75 Diskurspartikeln enthielten. 83 Prozent dieser Partikeln wurden korrekt verarbeitet.

2.4 Resümee

Um einen Überblick über den derzeitigen Stand der Entwicklung von Dialogsystemen zu geben, habe ich in diesem Kapitel stellvertretend für verschiedene Entwicklungsgrundsätze drei Dialogsysteme vorgestellt.

Im Verbmobil–Projekt geht es um die Entwicklung eines Systems zur Hilfe bei Kommunikationsproblemen, die auf lückenhaften englischen Sprachkenntnissen der Dialogpartner beruhen. Kommt der Dialog ins Stocken, kann man sich in seiner Muttersprache an das System wenden, welches dann die Äußerung ins Englische übersetzt. Verbmobil befindet sich, falls es gerade keine Übersetzungsarbeit leistet, in einem zuhörenden Modus. Auf diese Weise verfolgt es den Dialog und kann den Dialogkontext zur Übersetzungsarbeit nutzen. Die Module zum Sprachverstehen ermöglichen sowohl eine effiziente flache Analyse der Eingabe als auch eine aufwendige tiefe linguistische Analyse. Durch die Kombination der Verfahren werden die besten Ergebnisse erreicht. Es scheint also so zu sein, daß eine Kombination eines vor allem an der Linguistik orientierten Ansatzes (tiefe Analyse) und eines ingenieurwissenschaftlich geprägten Ansatzes (flache Analyse) erfolgversprechend ist.

Gegenstand in TRAINS-96 ist die Simulation einer Transportaufgabe. In dem Dialog zwischen Mensch und Maschine erhalten beide Kommunikationspartner die Möglichkeit zur Initiative, um die gestellte Aufgabe optimal zu lösen. In TRAINS-96 kann die gestellte Aufgabe nicht von einem sprachverstehenden System allein gelöst werden. Vielmehr stehen die sprachverarbeitenden Module in enger Kommunikation mit einem Planungsmodul und anderen Eingabekanälen. Im TRAINS–Projekt ist also ein Anspruch umgesetzt worden, der auch für das im Rahmen dieser Arbeit entwickelten Systems formuliert ist, nämlich die Realisierung eines in einem komplexen Gesamtsystem integrierten sprachverstehenden Systems.

Das Terminabsprachesystem ermöglicht die Vereinbarung eines Termins mit Hilfe eines Rechners. Die Spracherkennung und das Sprachverstehen interagieren in diesem System sehr eng und basieren teilweise auf einer gemeinsamen Wissensbasis. Die erzielten Ergebnisse belegen, daß ein Mensch–Maschine–Dialog auch dann erfolgreich gestaltet werden kann, wenn nicht jede Konstituente korrekt erkannt oder interpretiert wird. Schließlich zeigt das Terminabsprachesystem, daß das semantisches Netzwerksystem ERNEST ein geeigneter Formalismus zur Realisierung sprachverstehender Systeme ist.

Zu zeigen bleibt im weiteren, inwieweit die geschilderten jeweiligen Grundzüge der Systeme als Anregungen für die eigene Arbeit genutzt werden können. Im folgenden Kapitel geht es allerdings zunächst um die Vorstellung der Domäne, in der die Sprachverstehenskomponente eingesetzt werden soll.

Kapitel 3

Untersuchungen zur Modellbildung

Nehmen Sie nun den blauen Würfel und drehen Sie damit von unten her die rote Schraube fest, so daß aber nach vorne eine Windung besteht — versteht er nicht.

Aus dem Korpus [Bri95b]

Es gibt gegenwärtig keine automatischen sprachverarbeitenden Systeme, die jede beliebige Äußerung korrekt interpretieren können. Die große Vielfalt der Sprache macht die Realisierung einer solchen Maschine auch auf absehbare Zeit unmöglich. Vielmehr muß im vorhinein genau festgelegt werden, wofür und unter welchen Randbedingungen das System benutzt werden soll. Der erste Schritt in dieser Spezifikation ist die Festlegung des Szenarios, für welches das System zu entwickeln ist. Ausgehend von der Aufgabenstellung des Systems müssen anhand der in dem Szenario verwendeten Sprache, dem *Sprachfragment*, Modelle entwickelt werden, welche die Grundlage für die Implementation des Systems bilden. Die Untersuchung von Datenmaterial zum Zwecke der Modellbildung für die zu realisierende Sprachverstehenskomponente in einem Konstruktionsszenario ist der Gegenstand dieses Kapitels. Das Ziel der Untersuchungen besteht darin, klar formulierte und abgesicherte Anforderungen an die benötigten Modelle zu haben.

Die im folgenden vorgestellte Untersuchung zur Modellbildung basiert teilweise auf Ergebnissen von Arbeiten im Kontext des Sonderforschungsbereiches 360 „Situierte Künstliche Kommunikatoren“ (SFB 360). Ein Schwerpunkt in diesem Sonderforschungsbereich liegt auf der Untersuchung des Sprachgebrauchs in Konstruktionsdialogen — in [Ric96] wird das Forschungsprojekt als Ganzes vorgestellt. Es wäre zweifellos das beste Vorgehen gewesen, zur Modellbildung direkt auf linguistisch sehr fundierte und abgesicherte Forschungsarbeiten in diesem SFB zurückzugreifen. Bisher gibt es allerdings keine Arbeit, aus der ich unmittelbar Modelle für die automatische Sprachverarbeitung hätte ableiten können. Deshalb werden in diesem Kapitel die Grundlagen für die Modelle gelegt. Es ist wie folgt aufgebaut. Zunächst befasse ich

mich mit dem methodischen Herangehen bei der Modellbildung In Abschnitt 3.2 stelle ich die Domäne, in der die Sprachverstehenskomponente arbeitet und in Abschnitt 3.3 ein Korpus aus dieser Domäne vor. Anschließend lege ich Modellspezifikationen für Objektbenennungen und Handlungsmodelle vor und erläutere sie ausführlich mit Hilfe des Korpus. Danach werden die Struktur der Äußerungen und der Dialogverlauf im Korpus diskutiert. Eine Zusammenfassung beschließt das Kapitel.

3.1 Methodik der Modellbildung

Ausgehend von der Zielstellung des Kapitels ergeben sich zwei wichtige Fragen zur Herangehensweise: Zum einen muß geklärt werden, auf welcher Datengrundlage die Modelle gewonnen werden sollen. Zum zweiten muß das methodische Vorgehen herausgearbeitet werden. Der Beantwortung dieser Fragen dient dieser Abschnitt.

Die für die Untersuchung des Sprachverhaltens in einem bestimmten Szenario benötigten Daten können auf zwei Arten gewonnen werden. Entweder entstammen sie der Durchführung psycholinguistischer Experimente oder sie kommen aus der Analyse eines möglichst repräsentativen Korpus.

Psycholinguistische Experimente: Ein Gegenstand der Kognitionswissenschaften ist die Erforschung der menschlichen Sprachproduktion unter dem Blickwinkel der dabei stattfindenden mentalen Prozesse [Lev89]. Bei den zu diesem Zweck durchgeführten psycholinguistischen Experimenten sollen die Variablen, welche die Sprachverarbeitung beeinflussen können, kontrolliert und somit zuverlässige Forschungsergebnisse zu ausgewählten Aspekten erzielt werden. Selbst wenn mit diesem Ansatz Fehlerquellen (insbesondere das Übersehen von Einflußfaktoren) verbunden sind, gestattet nach [Ric93, Seite 120] „nur diese Methode die Analyse kausaler Beziehungen und die Analyse mentaler Prozesse in nachprüfbarer Weise“.

Analyse eines Korpus: Ein Korpus im linguistischen Sinne besteht aus einer Menge mündlicher oder schriftlicher Äußerungen, welche als Untersuchungsmaterial die Grundlage der Sprachbeschreibung bildet. Ein Qualitätsmerkmal für ein Korpus ist folglich seine Größe: liegt ein umfangreiches oder gar repräsentatives Korpus vor, kann eine recht zuverlässige Untersuchung des Sprachfragmentes vorgenommen werden. Allerdings ist die Erstellung eines großen Korpus auch mit einem erheblichen Aufwand verbunden. Sollen insbesondere mündliche Äußerungen gewonnen werden, müssen diese zunächst in einer geeigneten Umgebung aufgenommen und dann transkribiert werden, was mit einem hohen Aufwand an manueller Arbeit verbunden

ist (vergleiche zum Beispiel [Koh94]). Im Unterschied zu den psycholinguistischen Experimenten ist die Korpusanalyse vor allem auf die Beantwortung der Frage gerichtet, *wie* die Menschen in dem gewählten Szenario sprechen und nicht *warum* sie sich für eine bestimmte Formulierung entscheiden und welche mentalen Prozesse bei der Formulierungsfindung ablaufen.

Die beiden Herangehensweisen an die Untersuchung des Sprachverhaltens in einem Szenario stehen sich nicht unversöhnlich gegenüber, sondern können sich in sinnvoller Weise ergänzen. Wird nämlich ein Korpus innerhalb einer situierten Umgebung erhoben, können die Ergebnisse der sprachpsychologischen Forschung beispielsweise bei der Gestaltung der Aufnahmebedingungen für dieses Korpus hilfreich sein. Da die Sprachproduktion, insbesondere die dabei ablaufenden mentalen Prozesse, nicht im Zentrum dieser Arbeit steht, werde ich die Modelle mit Hilfe eines Korpus entwickeln und überprüfen.

Die zweite wichtige Frage bei der Modellbildung ist die Frage nach der Methodik. Damit die Modelle überprüfbar und reproduzierbar sind, muß ein klares Verfahren zu ihrer Gewinnung angegeben sein. Dadurch ist auch dokumentiert, wie entsprechende Modelle für andere Domänen gewonnen werden können.

Lobin wählt in [Lob98] den Weg, eine für die semantische Repräsentation bewährte Theorie, nämlich die der *Konzeptuellen Semantik* von Jackendorff [Jac83, Jac89a, Jac89b], zu erweitern und zu verfeinern. Damit wird zweierlei geleistet. Zum einen können Handlungsanweisungen in adäquater Weise modelliert und verarbeitet werden. Zum anderen wird der ursprüngliche Formalismus weiterentwickelt. Der Anspruch von [Lob98] geht also weit über das Ziel dieses Kapitels hinaus.

Eine andere Methodik besteht darin, mit Hilfe von automatischen Verfahren Modelle aus einem Korpus abzuleiten. Dazu muß dieser annotiert und automatisch unter statistischen und anderen Gesichtspunkten analysiert werden. Beispielsweise wurden im Verbmobil–Projekt auf diese Weise die Dialogakt–Modelle überarbeitet [Sch94]. Lager, der in [Lag95] eine Umgebung zur Korpusanalyse vorstellt, macht allerdings darauf aufmerksam, daß ohne eine Wissensbasis, die oftmals von Hand erstellt werden muß und auf Expertenwissen beruht, keine automatische Annotierung eines Korpus möglich ist. Man darf also nicht hoffen, ohne jegliches Vorwissen fertige Modelle oder Modellgrundlagen allein mit automatischen Verfahren zu bekommen.

In der vorliegenden Arbeit werden die Modelle wie folgt gewonnen. Zunächst wird anhand der Aufgabenstellung des zu realisierenden Systems festgelegt, welcher Modelle es bedarf. Sodann wird die Realisierung der Modelle im Korpus studiert.[1] Daraus ergibt sich eine klare Anforderung an die Beschaffenheit und Kompetenz der Modelle, die von Experten anhand einer Stichprobe exemplarisch überprüft werden (siehe Abschnitt 3.4). Diese von einem Formalismus

[1] Wie in Kapitel 5 zu sehen sein wird, muß die Sprachverstehenskomponente auch Aufgaben bewältigen, die nicht am Korpus studiert werden können. Insofern kann in diesem Kapitel nicht für alle notwendigen Modelle eine Grundlage geschaffen werden.

unabhängigen Spezifikationen führen schließlich zur Realisierung der Modelle in der Wissensrepräsentationssprache ERNEST^{++} (siehe Kapitel 4 und Kapitel 5). Der erste Schritt ist also die Festlegung, welche Modelle es überhaupt geben soll. Dazu lege ich nun dar, wie die vorliegenden Konstruktionsdomäne genau beschaffen ist.

3.2 Die Domäne

Die Konstruktionsgegenstände entstammen einem Spielzeugkasten für Kinder. Abbildung 3.1 zeigt die Art der Gegenstände, mit denen hantiert wird, und ein mögliches Ziel der Montage. Aufgrund der Bauteile wird dieses Szenario im folgenden *Baufix–Szenario*[2] genannt. Die Bau-

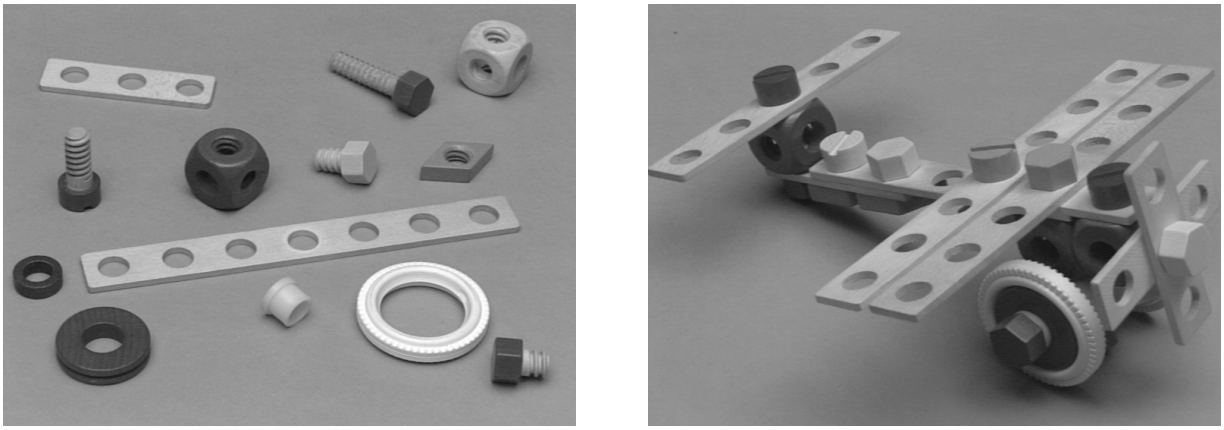

Abbildung 3.1: Beispiele für die verwendeten Konstruktionselemente und ein mögliches Ziel des Montageprozesses

teile sind zum großen Teil aus Holz und lassen sich einfach handhaben. Die zur Verfügung stehenden Bauteile möchte ich in ihrer Gesamtheit *Basiselemente* nennen. Sie können zunächst nach ihrem Typ (wie Schraubwürfel oder Leiste) unterschieden werden. Allerdings reicht diese Angabe nicht immer für eine eindeutige Unterscheidung aus. So gibt es Schrauben unterschiedlicher Länge, wobei die jeweilige Länge durch einen farbigen Schraubkopf eindeutig markiert ist. Außerdem muß bei Schrauben die Form ihres Kopfes beachtet werden: in allen Längen beziehungsweise Farben gibt es Rundkopf– und Sechskantschrauben. Eine bestimmte Schraube kann also unter Umständen nur durch die Angabe von Typ, Farbe, Form und Größe eindeutig identifiziert werden. Weiterhin gibt es Leisten unterschiedlicher Länge und Scheiben unterschiedlicher Dicke. Für den Konstruktionsprozeß ist es also sehr wichtig, Schrauben, Leisten und Scheiben genau zu spezifizieren. Die Schraubwürfel dagegen gibt es zwar in unterschiedlichen Farben, aber ihre Funktionalität ist farbunabhängig identisch.

[2] Baufix ist ein eingetragenes Warenzeichen der Lorenz GmbH, 82538 Geretsried.

3.3 Das Korpus

Steht für die Modellgewinnung nur Datenmaterial aus Dialogen zwischen Menschen zur Verfügung, trifft man auf ein Problem:

> „Die These ist: Menschen verhalten sich bei der natürlichsprachlichen MCI anders als in der zwischenmenschlichen Kommunikation.“ [Kra92, Seite 6][3]

Somit ergibt sich eine vertrackte Ausgangssituation bei der Entwicklung eines sprachverstehenden Systems: es wird Information über die Mensch–Maschine–Kommunikation in dem entsprechenden Szenario benötigt; diese liegt aber noch nicht vor, weil das System ja noch nicht existiert! Einen Ausweg aus dieser Situation bieten die *Wizard–of–Oz–Studien*, in welchen für Probanden die Interaktion mit einem Rechner simuliert wird. Dabei werden die vermeintlichen Systemreaktionen tatsächlich von einem Versuchsmitarbeiter erzeugt.[4] Idealerweise kann somit ein Entwicklungszyklus für Systeme zur Mensch–Maschine–Kommunikation etabliert werden, wie er nach [Bri95a] in Abbildung 3.2 zu sehen ist. Ausgangspunkt für die Systementwicklung ist nach diesem Schema die Analyse von Mensch–Mensch–Kommunikation, woraus sich eine erste Spezifikation für den konkreten Aufbau des Simulationssystems entwickeln läßt. Diese Spezifikation bildet die Grundlage für die Implementierung des Wizard–of–Oz–Systems, mit dem Sprachaufnahmen gewonnen werden. Die Beschreibung und Analyse der dabei erzielten Daten fließen in eine neue Spezifikation des Systems ein. Dieses Verfahren wird so lange iteriert bis schließlich eine Spezifikation und Realisierung des intendierten Systems durchgeführt wird. Leider sind es bedingt durch den bereits erwähnten erheblichen Aufwand oftmals finanziell–materielle und nicht wissenschaftlich–inhaltliche Gründe, die eine Beendigung der Iteration verlangen.

Im folgenden erläutere ich den Entwicklungszyklus der Wizard–of–Oz–Studie für ein Baufix–Szenario, die im Rahmen des SFB 360 durchgeführt wurde [Bri95a, Bri95b].

Voruntersuchungen

In die Erstellung der ersten Spezifikation des Wizard–of–Oz–Systems floß die Untersuchung von neun Mensch–Mensch–Dialogen ein, die im *SFB–Korpus* [SFB94] enthalten sind. Die Versuchspersonen in diesen Dialogen hatten die Aufgabe, das Flugzeug, wie es in Abbildung 3.1 gezeigt ist, zu bauen. Dabei fungierte eine Versuchsperson als *Instrukteur*. Sie instruierte die andere beteiligte Versuchsperson, den *Konstrukteur*. Jede beteiligte Person hatte einen identischen

[3] MCI steht in diesem Zitat für Mensch–Computer–Interaktion.

[4] Der eigentümliche Name für dieses Verfahren rührt von einem Märchen [Bau00]. Der Zauberer von Oz ist in Wirklichkeit eine Maschine, die aus dem Verborgenen heraus von einem Menschen bedient wird.

Abbildung 3.2: Entwicklungszyklus für Systeme zur Mensch–Maschine–Kommunikation mit Hilfe von Wizard–of–Oz–Studien (nach [Bri95a])

Satz von Bauteilen zur Verfügung. Dem Konstrukteur standen allein die sprachlichen Anweisungen des Instrukteurs als Montageanleitung zur Verfügung, denn Zeigegesten oder Hilfestellungen durch Blickkontakt waren dadurch unterbunden, daß die Probanden durch eine Sichtblende voneinander getrennt waren. Die Sichtblende hatte zusätzlich die Auswirkung, daß der Instrukteur zu keinem Zeitpunkt einen visuellen Eindruck vom Konstruktionsprozeß hatte.

Versuchsaufbau

Abbildung 3.3 zeigt eine Skizze des Versuchsaufbaus, der für die Wizard–of–Oz–Studie verwendet wurde. Die jeweilige Versuchsperson und die Projektmitarbeiter, welche das Systemverhalten simulierten („der Wizard“), hielten sich in getrennten Räumen auf. Die Versuchspersonen nahmen die Rolle des Instrukteurs, der Wizard die des Konstrukteurs ein. Die Anweisungen des

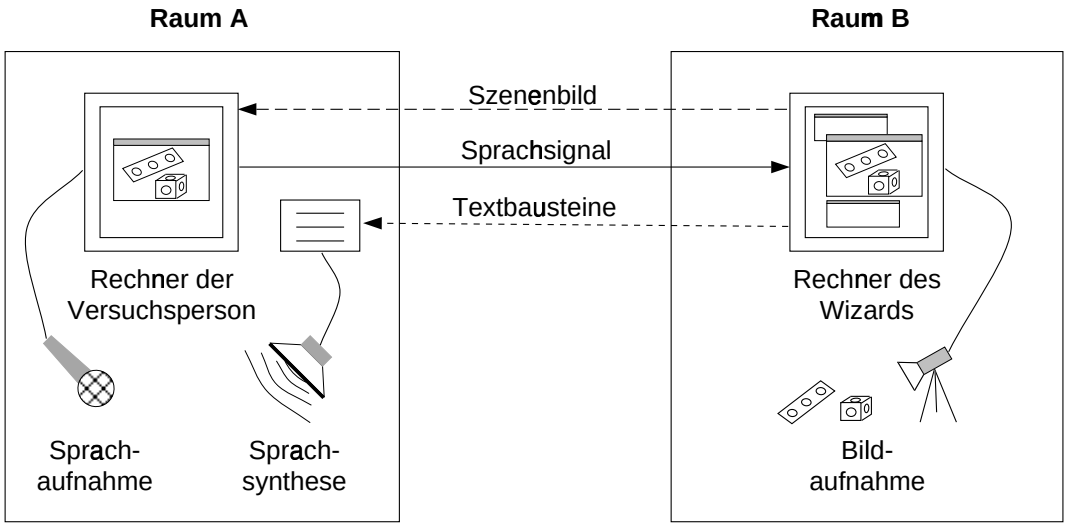

Abbildung 3.3: Versuchsaufbau in der Wizard–of–Oz–Studie für das Baufix–Szenario

Instrukteurs wurden jeweils in den Raum des Wizards (Raum B) übertragen. Dort wurden sie manuell ausgeführt, mit einer Kamera aufgenommen und das Ergebnis auf dem Bildschirm der Versuchsperson dargestellt. Der Wizard konnte akustische Systemreaktionen durch vorab festgelegte Textbausteine simulieren, die zu einem Sprachsynthesegerät gesendet wurden, das sich in Raum A befand. Die Projektmitarbeiter waren vom Versuchsleiter so instruiert worden, daß auch Fehlleistungen, wie sie bei realen Systemen vorkommen, simuliert wurden. Beispielsweise wurde ein Fehlschlagen der Spracherkennung dadurch suggeriert, daß die Systemreaktion „Ich konnte Ihre Äußerung leider nicht verstehen!“ in jedem Dialog mehrmals ausgegeben wurde.

Versuchsdurchführung

Die Durchführung der Studie vollzog sich in drei Iterationen, nämlich in einer *Trainingsphase* und in zwei modifizierten Szenarien. Ausgehend von den Untersuchungen der Mensch–Mensch-Dialoge wurde für die Trainingsphase als erster Iteration in der Studie ein initiales Wizard–of–Oz–System spezifiziert und realisiert. Es diente zur Gewinnung erster Erfahrungswerte mit dem System. Den Versuchsteilnehmern wurde mitgeteilt, daß ein Roboter Montageaufgaben erfüllen könne und nun eine natürlichsprachliche Ansteuerung dieses Roboters zu testen sei. Die vier Probanden erhielten die Aufgabe, „ein sprachverstehendes Programm“ [Bri95a, Anhang B, Seite V] anzuweisen, das Baufixflugzeug zusammenzubauen.

Gegenüber der Trainingsphase wurde die Aufgabenstellung im *Szenario I* unverändert belassen. Einige technische Unzulänglichkeiten, insbesondere eine unbefriedigende Reaktionszeit des Simulationssystems, wurden dagegen behoben. Außerdem wurden die Textbausteine, welche die Systemausgaben simulierten, gründlich überarbeitet und um dialogeinleitende und –abschließende Floskeln ergänzt. Es wurden 40 Dialoge aufgezeichnet.

In der dritten Phase, dem *Szenario II*, blieben die Rahmenbedingungen der Versuche bezüglich

des Versuchsaufbaus und der Systemreaktionen unverändert. Die Aufgabenstellung für die zehn Probanden wurde aber modifiziert: sie sollten nunmehr einen menschlichen Partner instruieren, das Flugzeug zu bauen. Er sei über das Mikrofon anzusprechen. Zum Hintergrund des Versuchs wurde den Teilnehmern mitgeteilt, daß er dem Testen neuer Kommunikationskanäle diene. Für die Versuchspersonen stellte sich die Situation also genau so dar wie in Szenario I — mit Ausnahme des angenommenen Kommunikationspartners. Während dort eine Maschine angewiesen werden sollte, war es nunmehr ein Mensch. Somit fungierte diese Gruppe als Kontrollgruppe bezüglich der eingangs des Abschnittes zitierten These.

3.4 Modellspezifikationen

Die Information, die ein Konstrukteur im Baufix–Szenario stets gewinnen muß, läßt sich in *einer* Frage formulieren: Welche Konstruktionshandlung soll mit welchen Objekten ausgeführt werden? Die Anweisungen eines Instrukteurs dienen letztlich der Beantwortung dieser Frage. Daher muß eine Sprachverstehenskomponente in diesem Szenario vor allem Benennungen von Objekten und Formulierungen zu Handlungsanweisungen verarbeiten können. Im weiteren Verlauf des Abschnittes werden diese Phänomene mit besonderer Intensität bearbeitet, um zu entsprechenden Modellen zu gelangen. Zusätzlich soll am Korpus studiert werden, wie die Instrukteure ihre Anweisungen als Ganzes strukturieren und ob sich besondere Erkenntnisse über den Dialogverlauf dieser Art der Mensch–Maschine–Kommunikation gewinnen lassen, die eine Grundlage für die Modelle bilden.

Als Korpus verwende ich im weiteren die Probandenäußerungen im Szenario–I der Wizard–of–Oz–Studie, deren Gesamtheit ich *Wizard–of–Oz–Korpus–I* nenne. Alle aufgeführten Äußerungen entstammen diesem Korpus. Die Bezeichnungen der einzelnen Äußerungen wurden von [Bri95b] übernommen.

3.4.1 Benennungen von Objekten

Objektbenennungen sind ein vielfach untersuchtes Phänomen (siehe zum Beispiel [Her76, MA92, Her94, Sch98]). Ich beziehe mich in der Definition von Objektbenennungen auf Herrmann, der als Ziel bei der Benennung von Objekten formuliert, „daß es einem Hörer gelingen soll, in einer bestimmten Situation einen bestimmten Gegenstand zu identifizieren“ [Her94, Seite 65]. Wie bereits ausgeführt haben sie in Konstruktionsinstruktionen einen hohen Stellenwert: beim Konstruieren werden Bauelemente zu einem Ganzen gefügt und dazu müssen die Bauelemente vom Konstrukteur identifiziert werden. Der propositionale Gehalt einer Anweisung hängt also entscheidend davon ab, wie es dem Instrukteur gelingt, die von ihm intendierten Objekte eindeutig zu benennen.

Benennungen von Basiselementen

Im Korpus finden sich vielfältige Bezeichnungen für die Basiselemente. Vorrangig verwenden die Instrukteure zur Benennung dieser Objekte relativ einfache Nominalphrasen wie

„Stecken Sie eine lange grüne Schraube durch den gelben Würfel." (03dl051)

In dieser Äußerung wird das eine Objekt außer über seinen Typ („Würfel") nur über seine Farbe („gelben") identifiziert. Da es in dem Baufix–Szenario nur gleichgroße Würfel gibt, reicht diese Benennung vollkommen aus. Im Gegensatz dazu wird in dieser Anweisung zur Disambiguierung der Schraube Farbe *und* Größe („lange grüne") verwendet, denn es existieren in dem Szenario ja Schrauben unterschiedlicher Farben, die mit „lang" bezeichnet werden könnten. Insbesondere der Gebrauch der Größenadjektive ist abhängig von der aktuellen Szene: eine drei Zentimeter lange Schraube wird im Kontext von zwei Zentimeter langen als „lang" bezeichnet werden können, während sie dann als „kurz" betrachtet werden könnte, wenn sich ansonsten nur noch längere Schrauben in der Szene befinden.

Neben den Farb– und Dimensionsadjektiven nutzen die Instrukteure auch die Form eines Basiselementes zur Identifizierung:

„Stecke die runde rote Schraube durch die Fünflochleiste." (16lo027)

Wie bei Benennungen mit Hilfe der Farbe wird eine Eigenschaft eines Bestandteils des Objektes zu dessen Identifizierung benutzt und auf das Ganze bezogen: nur der Kopf der Schraube ist rund oder eckig, trotzdem wird die Formulierung „runde Schraube" gewählt.

Schließlich verwenden die Sprecher auch die räumliche Lage eines Basiselementes zur seiner Identifizierung:

„Verbinde die Mutter mit der linken orangefarbenen Schraube." (28kd058)

Der Gebrauch von Lokalisationsadjektiven hängt stark von der konkreten Szene ab. Allerdings spielt der Einfluß des Blickpunktes des Sprechers auf die Wahl von Ortsbeschreibungen, wie sie im allgemeinen zu berücksichtigen sind, wegen des Versuchsaufbaus in der Wizard–of–Oz–Studie keine Rolle. Denn die Instruktionen wurden ja auf Grundlage eines zweidimensionalen Bildschirmfensters gegeben, welches beiden Dialogpartnern zur Verfügung stand.

Leider kann anhand des Korpus nicht überprüft werden, ob über– oder unterspezifizierte Benennungen vorliegen. Möglicherweise hätte ja in der obigen Äußerung 03dl051 auch die Formulierung „eine grüne Schraube" zur Objektidentifikation ausgereicht. Oder es ist unklar geblieben, ob eine eckige oder runde grüne Schraube gemeint war. Eine Einschätzung der Benennungen hinsichtlich dieses Aspektes könnte nur dann getroffen werden, wenn zu den Äußerungen die entsprechenden Szenenbilder vorlägen — das ist aber nicht der Fall.[5]

[5] Eikmeyer [Eik98] merkt an, daß ein Kontext intensionalen Charakter trägt. Darum reichten selbst Szenenbilder nicht aus. Zusätzlich wäre Information darüber nötig, welchen Kontext der Instrukteur in der jeweiligen Szene subjektiv wahrgenommen hat.

Für die Benennung der Basiselemente werden alternativ zu den Adjektiven auch synonyme Präpositionalphrasen verwendet:

> „Nimm eine dreilöchrige Leiste.“ (18kj003)

> „Nimm eine Leiste mit drei Löchern.“ (06al019)

Minimal spezifizierte definite Nominalphrasen wie

> „Fixiere die Schraube mit einer orangen Mutter.“ (18kj011)

gibt es nur dann, wenn aus dem Montagekontext klar ist, um welches Basiselement es sich handelt. Elliptische Objektbenennungen, die in den zwischenmenschlichen Konstruktionsdialogen aus dem SFB–Korpus auftreten (zum Beispiel Äußerung 06I140: „Dann mußt du den roten noch mal abschrauben.“), kommen im Wizard–of–Oz–Korpus–I praktisch nicht vor. Auch Relativsätze

> „Jetzt nimmst du eine Leiste mit fünf Löchern, eine gelbe Schraube mit einem Schlitz und eine Mutter, die orange ist.“ (11mm006)

zur Benennung eines Objektes gibt es nur sehr selten.

Mitunter werden *Referenzobjekte* zur Identifizierung eines Basiselementes herangezogen:

> „Schraube den grünen Block auf die orange Schraube neben der linken roten Schraube.“ (05sm028)

Einige Konstruktionen lassen sich nur durch die Verwendung von *Hilfsobjekten* bewerkstelligen. Die in

> „Weiter — befestige die aufliegende fünfbohrige Basisplatte mit der roten Schraube.“ (37sj030)

geforderte Verbindung kann ohne eine Schraube nicht hergestellt werden, die hier sogar noch näher („mit der roten Schraube“) spezifiziert wird. Häufig verzichten die Instrukteure allerdings darauf, das Hilfsobjekt überhaupt zu benennen:

> „Jetzt wird die kleine Leiste rechtwinkelig unter den roten Würfel geschraubt.“ (30kr019)

An dieser Äußerung wird offensichtlich, daß Instrukteure gewisses menschliches Allgemeinwissen auch beim Roboter voraussetzen. Daß es zum Schrauben einer Schraube bedarf, ist für uns Menschen eine solche Selbstverständlichkeit, daß das Hilfsobjekt gar nicht erwähnt wird.

Eine möglichst genaue Anweisung vereinfacht die Arbeit für den Konstrukteur erheblich. Daher bemühen sich die Instrukteure meist auch um exakte Detailangaben, wozu insbesondere *objektinterne Lokalisierungen* gehören:

„Nehmen Sie eine gelbe Schraube und drehen sie durch das zweite Loch von links des fünflöchrigen Stabes.“ (01pc024)

Vielfältig sind die Namen, welche für die Basiselemente gewählt werden. So werden die zum Bausatz gehörenden Leisten, von denen es nur drei unterschiedliche Arten gibt, mit 68 verschiedenen Namen belegt, die in Tabelle 3.1 aufgeführt sind. Dort zeigt sich exemplarisch die

Basisplatte	Brett	Brettchen	Dreibrett
Dreier	Dreierbrett	Dreierholzleiste	Dreierholzplättchen
Dreierholzstäbchen	Dreierleiste	Dreierloch	Dreierlochleiste
Dreierschiene	Dreierstäbchen	Dreierstück	Dreierstab
Dreierstange	Dreilochholzstück	Dreilochleiste	Dreilochschablone
Dreilochstück	Dreilochstab	Dreilochstange	Fünfer
Fünferbrett	Fünferholz	Fünferholzleiste	Fünferholzstäbchen
Fünferkette	Fünferleiste	Fünferlochbrett	Fünferlochleiste
Fünferschiene	Fünferstäbchen	Fünferstück	Fünferstab
Fünferstange	Fünflochleiste	Fünflochschablone	Fünflochscheibe
Fünflochschraube	Fünflochstab	Fünflochstange	Holzleiste
Holzplättchen	Holzplatte	Holzstäbchen	Leiste
Lochleiste	Plättchen	Platte	Schiene
Siebener	Siebenerbrett	Siebenerholzleiste	Siebenerholzstäbchen
Siebenerleiste	Siebenerschiene	Siebenerstäbchen	Siebenerstück
Siebenerstange	Siebenlochleiste	Siebenlochschablone	Siebenlochstange
Stäbchen	Stab	Stange	Streifen

Tabelle 3.1: Im Wizard–of–Oz–Korpus–I vorkommende Benennungen für die Baufix–Leisten

sehr große Variabilität der Namen für Objekte, deren jeweilige Auswahl durch die Sprecher Gegenstand der Forschung auch innerhalb des SFB 360 sind. In [Soc98] wird eine Umfrage im World Wide Web zur Benennung der Basiselemente vorgestellt, auf deren Grundlage Wahrscheinlichkeiten für die Wortwahl geschätzt werden können. Kessler berichtet in [Kes96] von psychologischen Experimenten, welche unter anderem der Verwendung von Devianzen bei der Referenzherstellung gewidmet sind, wie sie etwa in

„Nimm einen violetten Kreis.“ (12kg049)

enthalten sind. Einige der Basiselemente, insbesondere das vom Hersteller mit dem Namen „Mitnehmerbuchse“ versehene Teil, werden meist mit allgemeinen Namen bezeichnet wie in

„Drehe das weiße Plastikteil um.“ (22ra070)

oder durch ihre Funktion bei der Aggregierung charakterisiert:

„Nimm einen weißen Abstandhalter." (24ka061)

Einige Instrukteure verwenden Objektbenennungen, die bei einer sophistischen Betrachtung als falsch bezeichnet werden können, weil sie normalerweise ein ganz anderes Basiselement bezeichnen. Beispielsweise findet sich in mehreren Dialogen die Bezeichnung einer Schraube mit dem Wort „Mutter":

„Die beiden gelben Mutt äh Schrauben jetzt mit orangenen Muttern festschrauben" (05sm017)

Manchmal wird der Fehler von den Sprechern selbst entdeckt, was — wie im obigen Beispiel — zu Wort– oder Satzabbrüchen mit anschließender Reparatur führen kann oder dazu, daß im Verlauf des Dialogs plötzlich die korrekte Benennung verwendet wird.

Benennungen von Aggregaten

Aufgabe für die Instrukteure in den Wizard–of–Oz–Dialogen war die Anleitung der Konstruktion eines Flugzeugs. Dieses Ziel der Montage hatte erheblichen Einfluß auf die verwendeten Formulierungen. Wissen über Flugzeuge, ihre Gestalt und ihre Bestandteile setzten die Probanden bei dem Rechner voraus

„Drehe den Propeller des Flugzeugs um fünfundvierzig Grad." (09ho071)

oder mochten es mit einer einfachen Benennung vermitteln:

„Okay, das ist ein Propeller." (05sm050)

Im Verlaufe einer Konstruktion entstehen mitunter auch aus Basiselementen gebildete *Aggregate*, deren Aufgabe und Bedeutung für das Ganze zunächst nicht klar ist. Daher können sie auch nicht mit einem allgemein verständlichen Namen — wie zum Beispiel „Propeller" in der obigen Äußerung — versehen, sondern müssen zur Identifizierung umschrieben werden. Im Korpus finden sich dazu zwei Strategien: Entweder es erfolgt eine vergleichsweise sehr detaillierte Beschreibung des Aggregates

„Drehe das Objekt, das aus dem blauen Würfel, dem grünen Würfel und der grünen Schraube besteht, um neunzig Grad nach oben." (10hs053)

oder es bleibt der Verwendung ganz allgemeiner Namen:

„Schraube jetzt das gesamte Teil in den gelben Würfel." (22ra060)

Aus diesen Betrachtungen heraus wird für die Modellierung von Objektbenennungen in der Sprachverstehenskomponente folgende Spezifikation festgelegt:

1. Als Objektbenennungen im Baufix–Szenario sollen nur Benennungen von Basiselementen oder Aggregaten verstanden werden.

2. Es kann unterschieden werden zwischen der Benennung von intendierten Objekten, Hilfsobjekten und Referenzobjekten. Intendierte Objekte sind solche, mit denen eine Handlung ausgeführt werden soll. Hilfsobjekte werden zusätzlich benannt, wenn sie zur Ausführung dieser Handlung benötigt werden. Referenzobjekte sind Objekte, die in einer bestimmten räumlichen Relation zum intendierten Objekt liegen und zur Disambiguierung dieses Objektes benannt werden.

3. Objektbenennungen können durch Benennungen von objektinternen Lokalisationen ergänzt sein, um einen genauen Ort an dem Objekt zu spezifizieren.

3.4.2 Handlungsmodelle und Verben

Obgleich es im Baufix–Szenario nur vergleichsweise einfache Handlungsmöglichkeiten gibt, entstehen durch die Vielfalt ihrer Kombinationsmöglichkeiten sehr schnell komplexe Handlungsstrukturen. Aus diesem Grund und wegen des Reichtums an Formulierungsmöglichkeiten von Anweisungen[6] kann im Rahmen dieser Arbeit kein allumfassendes Modell für Handlungsanweisungen in diesem Szenario entwickelt werden. Vielmehr werde ich im folgenden die Handlungen im Baufix–Szenario, die von der Sprachverstehenskomponente verstanden werden sollen, festlegen und an Beispielen belegen, in welcher Art und mit welchen Verben im Korpus zu diesen Handlungen aufgefordert wird. Auf diese Weise kann das Modell zur Verarbeitung von Verben spezifiziert werden.

1. Herstellung einer unspezifizierten festen Verbindung

Das Verbinden von Basiselementen und Aggregaten ist das Anliegen der Konstruktion. Für das Ergebnis der Konstruktion ist es dabei von grundsätzlicher Bedeutung, ob eine feste oder lose Verbindung hergestellt werden soll. Eine feste Verbindung ist eine solche, bei der das Aggregat nicht wieder auseinanderfällt, wenn es an einer beliebigen Stelle angehoben wird. Diese werden angewiesen durch Formulierungen wie

> „Befestigen Sie bitte die beiden Schrauben in gelb am anderen Ende des Holzstückes.“ (40yy020)

Es sollen also Anweisungen verarbeitet werden können, in denen näher erläutert wird, auf welche Art und Weise die feste Verbindung herzustellen ist.

[6] In [Lob98, Seite 259 ff] wird ein wertvoller Eindruck davon gegeben.

2. Herstellung einer Schraubverbindung

Bemerkenswert bei den Anweisungen zur Herstellung einer Schraubverbindung ist die Vielfalt der Präfixe bei den verwendeten Verben. So wird zum Beispiel nicht nur „schrauben“ verwendet, wie in

> „Schraube nun eine orangene Mutter auf die Schraube.“ (20td029)

sondern mehrfach wird auch „anschrauben“, „einschrauben“, „festschrauben“, „zusammenschrauben“ sowie „verschrauben“ benutzt.

3. Herstellung einer Steckverbindung

Die Instrukteure in der Wizard–of–Oz–Studie haben sich häufig dazu entschieden, die Konstruktionsaufgabe so zu gliedern, daß Bauteile zunächst lose miteinander verbunden und dann verschraubt werden. Daher gibt es recht viele Formulierungen, die zur Herstellung einer zunächst instabilen Verbindung auffordern:

> „Stecke die rote Schraube durch das Brett und den roten Würfel.“ (21tt007)

4. Lösen einer Verbindung

Verbindungen sind immer dann zu lösen, wenn der Instrukteur seine Anweisung korrigieren will:

> „Entfernen Sie den Zylinder, drehen ihn um hundertachtzig Grad und stecken ihn wieder auf.“ (03dl043)

5. Nehmen eines Objektes

Jedes Konstruieren beginnt damit, daß das Konstruktionselement in die (Roboter–) Hand genommen werden muß. Das Nehmen eines Objektes bildet daher oftmals den Anfang eines Konstruktionsabschnittes:

> „Nehmen Sie jetzt ein lilafarbenes Röhrchen und schrauben Sie dieses auf die blaue Schraube drauf.“ (02gv100)

6. Positionieren eines Objektes

Häufig ist es notwendig, Objekte in einer bestimmten Art und Weise zu positionieren, etwa ein Teil vorübergehend abzulegen, um die (Roboter–)Hand frei zu haben:

> „Bitte dieses Teil links neben den blauen Würfel legen.“ (30kr080)

Das Positionieren kann auch eine andere Ausprägung haben, nämlich die ortsinvariante Veränderung der Lage des Objektes:

> „Drehe das Flugzeug so um, daß die Oberseite zu sehen ist.“ (09ho070)

Wenn Anweisungen zu diesen sechs Handlungen von der Sprachverstehenskomponente verarbeitet werden können, ist bereits ein umfangreiches Konstruieren möglich. Zur angemessenen Gestaltung der Mensch–Maschine–Kommunikation sollen zusätzlich Äußerungen verstanden werden, die folgende Aspekte betreffen:

7. **Vergabe eines Namens an ein Aggregat**
   Es soll möglich sein, während der Konstruktion an entstandene Aggregate Namen zu vergeben wie in dem bereits weiter oben angeführten Beispiel

   „Okay, das ist ein Propeller.“ (05sm050)

8. **Sofortige Unterbrechung der Ausführung**
   Ist während der Ausführung einer Handlung für den Instrukteur schon vor deren Ende sichtbar, daß etwas anderes passiert, als er gemeint hat, soll die Ausführung unterbrochen werden können. Im Korpus findet sich kein Beispiel dazu, denn der Aufbau war ja so organisiert, daß dem Probanden immer nur das Ergebnis der Ausführung dargeboten wurde, nicht aber ihr Verlauf.

9. **Beendigung des Konstruktionsprozesses**
   Damit der Konstruktionsprozeß kontrolliert beendet werden kann, sollen Äußerungen wie zum Beispiel

   „Ähm, das Zusammenbauen ist jetzt beendet.“ (30kr125)

   korrekt interpretiert werden.

Zusammenfassend kann also das Modell für die Verarbeitung von Verben wie folgt spezifiziert werden:

1. Die Sprachverstehenskomponente soll Verben, die bei der direkten Anweisung einer Konstruktion benutzt werden, verarbeiten. Die Verben sind den vorgestellten Handlungen zuzuweisen.

2. Zusätzlich sollen solche Verben verarbeitet werden, die bei der Vergabe eines Namens für Aggregate, der Unterbrechung einer Ausführung oder der Beendigung des Konstruktionsprozesses verwendet werden.

3.4.3 Plausibilität der Spezifikationen

Zur Absicherung der Spezifikationen für die Objektbenennungen und die Verarbeitung von Verben wurden diese von drei Linguistinnen exemplarisch überprüft. Dazu wurden ihnen die Spezifikationen erläutert, und sie wurden gebeten, eine Teststichprobe von zwanzig Äußerungen zu

bearbeiten.[7] Sie sollten die Stellen markieren, an denen nach ihrem Verständnis der Spezifikation Objektbenennungen (unterschieden nach intendiertem Objekt, Referenz– oder Hilfsobjekt und objektinterne Lokalisation) auftraten. Weiterhin sollten die in den Anweisungen enthaltenen Verben, wenn möglich, den vorgestellten Handlungen zugeordnet werden. Als Beispiel wurde ihnen die markierte Äußerung 23mc015 (siehe Abbildung 3.4) an die Hand gegeben. Ihre Mar-

In einem nächsten Schritt	steckst	du	die gelbe Schlitzschraube	vor dem Würfel	in das Loch.
	stecken		**intendiertes Objekt**	**Referenzobj.**	**objektint. Lokalisation**	

Abbildung 3.4: Beispiel für die Markierung einer Äußerung anhand der vorgestellten Spezifikationen

kierungen wurden verglichen mit dem Ergebnis einer von mir auf der gleichen Teststichprobe durchgeführten Bearbeitung, die insgesamt 65 markierte Stellen enthielt. Etwa 88 Prozent meiner Markierungen wurden von *allen drei* Expertinnen geteilt. Interpretiert man die Mehrheit der jeweiligen Expertinnenmeinung als maßgebend, so erhöht sich die Zahl der Übereinstimmungen sogar auf über 95 Prozent. Das bedeutet also, daß in weniger als fünf Prozent der Markierungen die Spezifikationen nicht ausreichten, um zu einem mehrheitlichen Ergebnis zu kommen. Damit scheinen sie ein gutes Fundament für die Modellbildung zu sein.

Mit der exemplarischen Überprüfung der Spezifikationen ist die Erarbeitung der Grundlagen für die Modelle von Objektbenennungen und Verben abgeschlossen. Nun soll die Aufmerksamkeit auf den Aufbau der einzelnen Äußerungen und den Dialogverlauf im Korpus gerichtet werden.

3.5 Aufbau der Äußerungen

In der modernen Mensch–Maschine–Kommunikation ist die sprachliche Freiheit der Systembenutzer ein häufig postuliertes Ziel [Abe97]. Um unter diesem Paradigma herauszufinden, wie Menschen eine Maschine in einer Konstruktionsszene anweisen, erhielten die Probanden in der Wizard–of–Oz–Studie keinerlei Hinweise oder gar restriktive Anweisungen bezüglich ihrer Wortwahl und Formulierung. Im vorherigen Abschnitt zeigte sich die Konsequenz dieses Studiendesigns in der Vielfalt der Benennungen — den Probanden war eben kein Inventar von Objektnamen an die Hand gegeben worden und daher mußten sie Umschreibungen wählen oder eigene Namen für die Objekte erfinden. Ähnliches gilt auch für den Aufbau der Äußerungen. Da keine „erlaubte Grammatik“ vorgegeben war, finden sich sehr vielfältige Formulierungen und Satzkonstruktionen. Bezüglich des Aufbaus der Äußerungen möchte ich folgende Aspekte herausstellen:

[7] Die Äußerungen der Teststichprobe wurden zufällig aus dem Wizard–of–Oz–Korpus–I ausgewählt.

- Das häufige Auftreten von Imperativsätzen ist angesichts des Konstrukteur–Instrukteur–Szenarios sicherlich wenig überraschend. Allerdings finden sich auch viele Anweisungen, welche die Form eines Aussagesatzes haben:

  „Der weiße Reifen wird um das rote Rad gelegt.“ (07ac025)

  Bereits die Auswertung der Dialoge der Trainingsphase zeigte, daß im Unterschied zu den im Vorfeld der Studie untersuchten Mensch–Mensch–Dialogen relativ wenig Fragen zur Objektspezifikation gestellt wurden [Bri95a, Seite 14]. Dies bestätigt sich auch beim Studium der Dialoge des Szenarios I. Das bereits im Zusammenhang der Objektbenennungen erwähnte sehr seltene Vorkommen von Relativsätzen läßt sich für das Korpus insgesamt bestätigen.

- Die Instrukteure bemühten sich um eine knappere Formulierung, wenn sie als Dialogpartner eine Maschine annahmen. Während nämlich eine Äußerung im Szenario I durchschnittlich aus etwa zwölf Wörtern bestand, verwendeten die Probanden im Szenario II, die ja einen Menschen als Konstruktionspartner annahmen, im Schnitt fast 23 Wörter pro Äußerung. Allerdings ist die Streuung recht groß: der wortkargste Sprecher im Szenario I kommt mit nur knapp sieben Wörtern pro Äußerung aus. Im wortreichsten Dialog dagegen werden im Durchschnitt fast 32 Wörter pro Äußerung benutzt.

- Nur ein Instrukteur verfällt extrem in eine unnatürliche „Maschinensprache“:

  „Führe zusammen Schraubenmutter, Würfel grün, Bauteil mit drei Löchern.“ (15wk043)

  Vereinzelt gibt es solche Tendenzen aber auch in anderen Dialogen. Sie zeigen sich zum Beispiel darin, daß die Artikel in den Objektbenennungen einfach weggelassen werden. Anscheinend gibt es Menschen, die sich in der Kommunikation mit einer Maschine auf das ihrer Meinung nach wichtigste konzentrieren möchten und glauben, der Maschine zu helfen, wenn sie nur Satzfragmente übermitteln.

Betrachtet man das Korpus unter der Aufgabenstellung, ein Verarbeitungsmodell für gesprochene Sprache zu gewinnen, können keine weiteren Besonderheiten über den Aufbau der Äußerungen festgestellt werden. Es gibt weder ein präferiertes Satzmuster noch eine Regel bezüglich der Stellung der verwendeten Konstituenten zueinander.

Insgesamt ergibt sich also, daß das Modell für die Äußerungen als Ganzes keine engen Restriktionen bezüglich der syntaktischen Realisierung einer Anweisung enthalten darf. Vielmehr muß vorgesehen werden, daß die beschriebenen Objektbenennungen und Verben praktisch in freier Stellung zueinander auftreten können.

3.6 Dialogstruktur

Nachdem in den beiden vorangegangenen Abschnitten die Bestandteile einer Äußerung sowie der Aufbau derselben dargelegt wurden, soll nun der Blick auf den Dialogverlauf im Ganzen gerichtet werden. Zunächst stelle ich die verschiedenen Dialogakte vor. Dann untersuche ich, welche Abfolgen von Dialogakten zu beobachten sind und gehe schließlich der Frage nach, inwieweit sich die Instrukteure während des Dialogs an das Wizard–of–Oz–System anpassen.

3.6.1 Dialogakte

Den einzelnen Äußerungen lassen sich Dialogakte zuordnen. Ein Dialogakt ist nach [Sch95b] im Unterschied zu den illokutiven Akten aus [Aus62] und der damit verbundenen Sprechakttheorie [Sea69] eine Abstraktion auf pragmatischer Ebene. Er charakterisiert eine Äußerung unabhängig von ihrer grammatikalischen Realisierung. Diese Charakterisierung ist aber für die automatische Sprachverarbeitung viel aussagekräftiger als diejenige von Sprechakten [Sch95b, Seite 35ff].

Anhand des Korpus können für die Konstruktionsanweisungen die nachfolgenden Dialogakte bestimmt werden, die der Natur der Sache nach teilweise mit den bereits vorgestellten Handlungen zusammenhängen:

1. **Bauvorgang**
   Der Instrukteur bezieht sich auf einen Bauvorgang im Ganzen.

   „Wiederhole den letzten Bauvorgang.“ (35al046)

   Diese Äußerungen tragen häufig auch zur Strukturierung der Konstruktionsaufgabe und –ausführung bei.

2. **Verbindung herstellen**
   Der Instrukteur fordert zur Herstellung einer Verbindung auf. Dieser Dialogakt findet sich am häufigsten, weil die Verbindung das elementare Konstruktionsprinzip ist.

3. **Verbindung lösen**
   Der Konstrukteur wird aufgefordert, eine bestehende Verbindung wieder zu lösen.

4. **Allgemeine Bauanweisung**
   Der Instrukteur gibt eine nicht näher spezifizierte allgemeine Bauanweisung, wie zum Beispiel

   „Unter den grünen Würfel baust du jetzt den gelben Würfel.“ (23mc040)

   Solche Anweisungen werden häufig im Fortgang des Dialogs näher spezifiziert.

5. **Lageveränderung**
   Der Instrukteur möchte eine Lageveränderung eines Objektes bewirken, die die korrekte Ausführung einer Konstruktion erst ermöglicht. Dazu gehören insbesondere Anweisungen, die zum Nehmen oder Positionieren eines Objektes auffordern (siehe Seite 38).

6. **Beschreibung**
   Der Instrukteur beschreibt ein Aggregat. Dieser Dialogakt dient entweder zur Disambiguierung von Objekten oder dazu, beim Konstrukteur eine bildliche Vorstellung über das zu Bauende zu erzeugen und somit die Instruktion verständlicher zu machen, wie beispielsweise in

   > „Unter die Leiste mit fünf Löchern muß eine Leiste mit drei Löchern, so daß ein Loch übersteht.“ (33gs017)

Diese Dialogakte beziehen sich nur auf Anweisungen im engeren Sinne und decken Äußerungen wie

> „Aha, muß man jeden Schritt abwarten.“ (14lt002)

nicht ab, denn solche Äußerungen, deren direkter propositionaler Gehalt in Bezug auf die Konstruktionsaufgabe gering ist, spielen im Kontext dieser Arbeit eine untergeordnete Rolle.

In [Fis95] werden für ein und dasselbe Szenario noch weitere Dialogakte festgelegt, die in zwischenmenschlichen Dialogen beobachtet wurden. So gibt es in den Mensch–Mensch–Dialogen aus dem SFB–Korpus beispielsweise Klärungsfragen von Konstrukteuren, die im Wizard–of–Oz–System nicht implementiert waren. Auch ihr Gegenstück, die Beantwortung von Klärungsfragen durch den Instrukteur, findet sich in der simulierten Mensch–Maschine–Kommunikation nicht.

3.6.2 Dialogmodell

Aus dem Studium von Sequenzen von Dialogakten kann insbesondere für Auskunfts– oder Informationsdialoge häufig ein sehr schönes Ablaufmodell für die Dialoge der jeweiligen Domäne gefunden werden. Mast beispielsweise stellt ein detailliertes Dialogmodell für Zugauskunftsdialoge vor [Mas93], das hauptsächlich aus realen Dialogen mit der Regensburger Bahnauskunft (FACID–Korpus [Hit88]) gewonnen wurde. Diese Dialoge sind insofern sauber strukturiert, als daß nach einer optionalen Begrüßungsphase der Dialogpartner das Ziel des Dialogs eindeutig festgelegt ist: es geht darum, daß der Bahnbeamte die Informationen bekommt, die er benötigt, um eine gewünschte Verbindung herauszusuchen. Dazu braucht er im einfachsten Fall nur einen Abfahrtsort, einen Ankunftsort, sowie den Zeitpunkt der Reise. In welcher Reihenfolge diese

Informationen gegeben werden, ist belanglos. Von vornherein ist klar, daß die gewünschte Auskunft das Ziel des Gesprächs ist. Darin liegt ein großer Unterschied zu den Konstruktionsdialogen. Dort hat der Instrukteur sehr viele Möglichkeiten, das Konstruktionsziel zu erreichen, welches dem Konstrukteur möglicherweise zu Beginn sogar unbekannt ist. Der Instrukteur ist weder festgelegt in der Reihenfolge der Konstruktionen noch gibt es feste Reglements auf welche Art und Weise die Aggregate hergestellt werden müssen. Manche Instrukteure gehen sehr ins Detail, andere geben eher allgemeinere Anweisungen. Die Entwicklung eines aussagekräftigen, deterministischen Ablaufmodells für die Dialoge des Korpus erfordert daher eine sehr aufwendige, detaillierte Untersuchung der Dialogaktsequenzen, die zu diesem Zweck sicher noch verfeinert werden müßten, und war aus Zeitgründen im Rahmen dieser Arbeit nicht möglich. Vielmehr muß ich im weiteren davon ausgehen, daß jede Sprechhandlung auf jede andere folgen kann und sich diesbezüglich keine Restriktionen festlegen lassen.

Wenn auch kein sinnvolles Ablaufmodell für die Dialoge gewonnen werden kann, stellt sich dennoch die Frage, in welcher Weise sich die einzelnen Instruktionen aufeinander beziehen. Der Auszug in Abbildung 3.5 entstammt dem Mensch–Mensch–Dialog B1_02 aus dem SFB–

Instrukteur 1: „So, jetzt nimmst du die eine Fünferstange."

Konstrukteur 1: „Mhm."

Instrukteur 2: „Und eine Dreierstange. Jetzt baust du die so untereinander, daß du ähm die Fünferstange sozusagen verlängerst, indem du die Dreierstange mit zwei Löchern dadran schraubst."

Konstrukteur 2: „Mit zwei Löchern, das ist also eine Sechser oder wie ?"

Instrukteur 3: „Genau, daß es eine Sechser wird und dafür nimmst du die zwei gelben Schrauben, um das festzumachen. Und dazu als Mutter diese orangefarbenen Rauten."

Abbildung 3.5: Aushandlungsbeispiel aus dem Dialog B1_02 des SFB–Korpus [SFB94]

Korpus. Hier erfolgt ein echtes Aushandeln der Konstruktion. Der Konstrukteur ist nicht nur passiv Ausführender, sondern trägt selbst zur Klärung der Anweisung bei, indem er eine eigene Benennung für das zu bauende Aggregat vorschlägt („eine Sechser" in seiner zweiten Äußerung). Diese Benennung wird vom Instrukteur wieder aufgegriffen und somit zur gemeinsamen Grundlage für den weiteren Verlauf des Dialogs. Instrukteur und Konstrukteur beziehen sich also nicht nur auf das, was sie selbst formuliert haben, sondern auch auf den Gesprächspartner. Bei den simulierten Mensch–Maschine–Dialogen gibt es dieses Phänomen nicht, wie das Beispiel in Abbildung 3.6 verdeutlicht. Dieser Ausschnitt bezieht sich auf den gleichen Konstruktionsabschnitt wie im obigen zwischenmenschlichen Dialog: es sollen zwei Leisten in bestimmter Weise miteinander verschraubt werden. Der Instrukteur in der Mensch–Maschine–Kommunikation

Instrukteur 1: „Verschraube ähm Holzleiste mit drei Löchern und roten Würfel mit roter Sechskantschraube.“

System 1: „Ich habe verstanden. Ihre Anweisung wird bearbeitet. Einen Augenblick bitte.“

Instrukteur 2: „Lege fünflöchrige Holzleiste auf die beiden freien Löcher der dreilöchrigen Holzleiste.“

System 2: „Ich habe verstanden. Ihre Anweisung wird bearbeitet. Augenblick bitte.“

Instrukteur 3: „Verschraube das mittlere Loch der dreilöchrigen Holzleiste mit einer gelben Schlitzschraube und einem orangem — und einer orangen Mutter.“

Abbildung 3.6: Auszug aus dem Dialog 35al des Wizard–of–Oz–Korpus–I

geht aber im Kontrast zu den Mensch–Mensch–Dialogen in keiner Weise auf die Äußerungen des Dialogpartners ein — die Systemausgaben geben auch keinen Anlaß dazu. Er bezieht sich aber sehr wohl auf seine eigenen vorhergehenden Anweisungen. Die Beschreibung „die beiden freien Löcher der dreilöchrigen Holzleiste“ in seiner zweiten Anweisung kann nur dann verstanden werden, wenn sie im Kontext der ersten Anweisung gesehen wird. In der letzten Instrukteursanweisung ist auch sichtbar, daß auf bereits bekannte Objekte mit einem definiten Artikel verwiesen wird („der dreilöchrigen Holzleiste“), während neu zu benutzenden Basiselemente mit einem indefiniten Artikel („mit einer gelben Schlitzschraube“) versehen sind.

3.6.3 Systemadaption

In der Testreihe zur Evaluierung des Terminabsprachesystems (siehe Seite 20ff) war zu beobachten, daß die Benutzer schon nach kurzer Zeit herausgefunden hatten, welche Art von Formulierungen das System besonders gut verarbeiten konnte. Aus diesem Grund ergab sich die Vermutung, daß die Instrukteure ihre Konstruktionsstrategie, Formulierung und Wortwahl an das Wizard–of–Oz–System adaptieren. Insbesondere hatte ich erwartet, daß die knapp und einfach formulierten Systemantworten eine ebensolche Sprache der Instrukteure herausfordern würde. Diese Vermutung hat sich nicht bestätigt. Abbildung 3.7 zeigt die Anzahl der Wörter, die die Instrukteure in den jeweils ersten 30 Äußerungen gesprochen haben. Sie hat selbst im Anfangsbereich keine monoton abfallende Steigung, die auf einen etwa konstanten Wert führt und widerspricht damit einer Anpassungsthese im Baufix–Szenario zumindestens hinsichtlich der knappen und einfachen Formulierungen. Es scheint also keine dialogeinleitende Phase der Orientierung und Gewöhnung an das System zu geben.

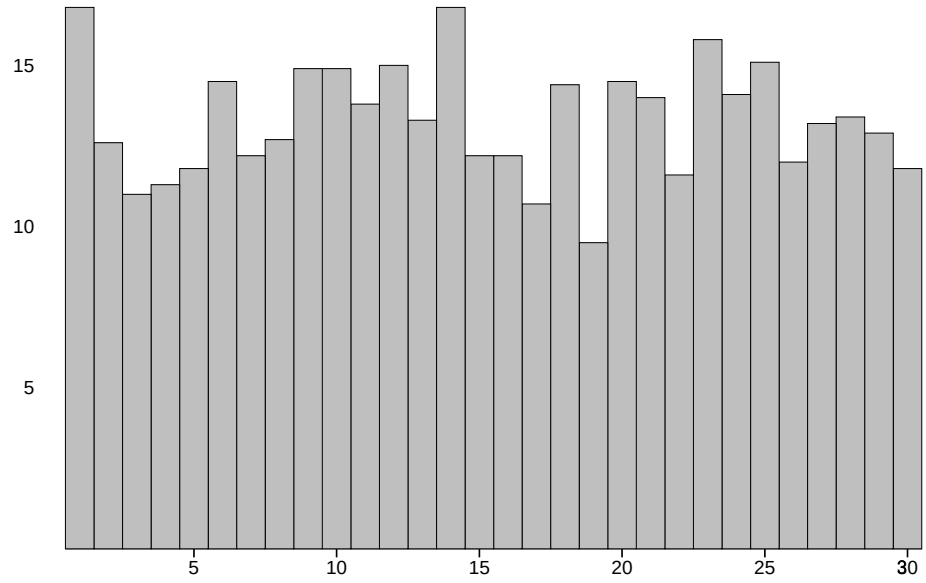

Abbildung 3.7: Durchschnittliche Anzahl der Wörter in den ersten 30 Äußerungen im Szenario I der Wizard–of–Oz–Studie

Zusammenfassend lassen sich aus der Korpusuntersuchung folgende Aspekte als Grundlagen der Modellbildung für die Dialogführung festhalten:

- Es lassen sich anhand des Korpus zwar Dialogakte festlegen, es kann aber kein aussagekräftiges Ablaufmodell für diese gefunden werden.
- Gleichwohl muß vorgesehen werden, daß sich die Instrukteure auch in der Mensch–Maschine–Kommunikation auf vorhergehende Anweisungen beziehen.
- Es kann nicht davon ausgegangen werden, daß es eine Adaption des Sprechers an das System bezüglich der Länge und Kompaktheit der Anweisungen gibt.

3.7 Zusammenfassung

In diesem Kapitel sind die Grundlagen für die Modellbildung der Sprachverstehenskomponente gelegt worden.

Ausgangspunkt dazu ist die Festlegung der Methodik. In dieser Arbeit wird der Weg gewählt, aus der Anforderung an die zu realisierende Sprachverstehenskomponente festzulegen, welcher Modelle es bedarf. Anhand eines Korpus wird untersucht, wie die Modelle realisiert sind, was schließlich zu einer Spezifikation der Modelle führt. Diese bilden die Grundlage für die Umsetzung der Modelle in einen Formalismus.

Die zugrunde liegende Domäne ist das Baufix–Szenario. Mit einfachen Bauelementen können Konstruktionsaufgaben bewerkstelligt werden. Als Korpus dienen im Rahmen des SFB 360 gewonnene Instruktionen, die in einer Wizard–of–Oz–Studie aufgenommen wurden.

Die wichtigsten Informationen für eine erfolgreiche Konstruktion bergen die Objektbenennungen und die Verben, die zu einer Handlung auffordern. Daher werden für sie Spezifikationen entwickelt, die von der Sprachverstehenskomponente zu erfüllen sind. Die Spezifikationen werden von drei Expertinnen auf einer zufällig ausgewählten Teststichprobe überprüft — es ergibt sich eine Übereinstimmung von über 95 Prozent.

Bezüglich des Aufbaus der Äußerungen lassen sich keine engen Restriktionen für die Modellierung angeben. Das gleiche gilt für das Dialogmodell. Obgleich Dialogakte festgelegt werden können, kann kein aussagekräftiges Ablaufmodell für diese Dialogakte gefunden werden.

Im folgenden Kapitel stelle ich den Formalismus vor, in dem die Modelle realisiert werden, bevor im übernächsten Kapitel die konkrete Realisierung ausführlich beschrieben wird.

Kapitel 4

Wissensrepräsentation mit ERNEST++

Das, was du sagst, soll wahr sein,
Das, wie du's sagst, soll klar sein.
Friedrich Güll

Möchte man ein System zur automatischen Sprachverarbeitung erstellen, ergibt sich — wie bei anderen großen Computersystemen auch — die Frage nach der Wissensrepräsentation. Zum allgemeinen Studium dieser Frage gibt es eine Menge grundlegender Literatur (wie zum Beispiel [Win83, Sow84, Cha85, Hey88]). Ich werde in diesem Kapitel nur auf die von mir hauptsächlich zur Wissensrepräsentation verwendete Netzwerksprache ERNEST++ eingehen. Bei ihrer Darstellung beziehe ich mich primär auf [Kum98b].

ERNEST++ ist ein objektorientiertes semantisches Netzwerksystem, welches aus ERNEST (**Er**langer **Ne**tzwerk**s**ys**t**em) [Sag85, Sag90, Kum92, Kum93, Sag97] hervorgegangen ist. ERNEST wurde speziell zur Interpretation von Sensordaten entwickelt. Bei der Portierung der Konzeption und Implementation des prozedural organisierten ERNEST in die objektorientierte Sichtweise von ERNEST++ [BP95] wurde der ursprüngliche Sprachumfang nahezu erhalten und später um einige Aspekte erweitert [Löm98, Löm99].

Zunächst erläutere ich die Grundidee der Modellierung mit semantischen Netzen im allgemeinen. Danach stelle ich die wichtigsten Komponenten von ERNEST++ und die Inferenzregeln zur Wissensnutzung vor. Anschließend erläutere ich die Verarbeitungsstrategie und die Einflußmöglichkeiten des Modellierers auf die Abarbeitung des Eingangssignals. Sodann lege ich einige Aspekte der objektorientierten Implementation von ERNEST++ dar. Eine Zusammenfassung beschließt das Kapitel.

4.1 Semantische Netze

Semantische Netze wurden von [Qui68] als einfaches Modell des menschlichen Gedächtnisses eingeführt. Die Grundelemente dieses Formalismus zur Wissensrepräsentation sind Knoten und Kanten. Knoten repräsentieren Begriffe, Objekte oder Ereignisse[1], während Kanten die Beziehung zwischen diesen darstellen. So kann elementares Wissen über eine Familie in einem einfachen semantischen Netz modelliert werden. Abbildung 4.1 zeigt eine solche Wissensbasis. In ihr ist modelliert, daß die Familie aus Eltern, Kindern und einem Haustier besteht. Es werden einige Beziehungen zwischen den Begriffen und auch die real existierenden Instanzen angegeben, die in ihrer Gesamtheit die Familie bilden.

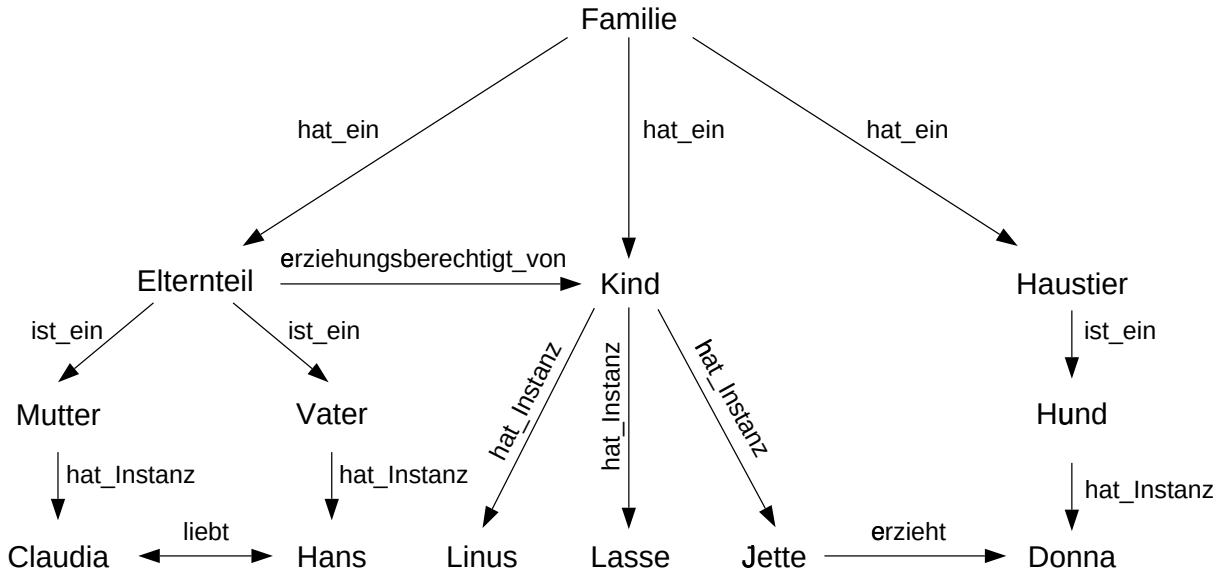

Abbildung 4.1: Semantisches Netz, das Wissen über eine Familie repräsentiert

In diesem Netz werden ganz spezielle Kanten wie *liebt* oder *erzieht* verwendet. Semantische Netze bieten also die Möglichkeit, beliebige Beziehungen zwischen Begriffen oder Entitäten durch die Markierung einer Kante, ihre *Rolle*, auszudrücken. Das Beispielnetz umfaßt auch Kantentypen, die in vielen semantischen Netzwerkformalismen benutzt werden und denen im allgemeinen eine feste Bedeutung zugeschrieben ist:

- Die *hat_ein*–Kante beschreibt eine Bestandteilsbeziehung zwischen den im Ursprungs– und Zielknoten der Kante repräsentierten Begriffe.

- Die *ist_ein*–Kante etabliert eine Generalisierung. In obigem Netz wird mit Hilfe dieses Kantentyps ausgesagt, daß ein Hund ein spezielles Haustier ist. Üblicherweise werden

[1] Im weiteren werde ich die Vokabel „Begriff“ abkürzend für „Begriff, Objekt oder Ereignis“ verwenden.

die Informationen über einen Begriff mit Hilfe dieses Kantentyps vererbt: im Beispielnetz erben die Knoten *Mutter* und *Vater* die Erziehungsberechtigung vom *Elternteil*–Knoten.

- Die *hat_Instanz*–Kante zeigt an, daß der Zielknoten eine Instanz des im Ursprungsknoten modellierten Begriffs darstellt: Linus ist ein bestimmtes Kind in der Familie. Als Instanz des *Kind*–Knotens erfüllt er die dort formulierten allgemeinen Eigenschaften und Beziehungen zu anderen Knoten.

Schon an diesem kleinen Netz lassen sich die positiven Eigenschaften dieses Wissensrepräsentationsformalismus erkennen (siehe auch [Kum92, Seite 41f]):

Modularität: In einem Knoten findet sich zentralisiert das Wissen über einen Begriff. Somit ist eine modulare Organisation des Wissens garantiert. Dieses Merkmal ist vor allem bei der Modellierung und Erweiterung großer Wissensbasen von Vorteil.

Wohlstrukturiertheit: Durch den Knoten–Kanten–Aufbau ergibt sich eine wohlstrukturierte Wissensbasis. Sie ist — insbesondere wegen der Möglichkeit der graphischen Darstellung — sehr leicht lesbar.

Umsetzbarkeit von Expertenwissen: Die Umsetzung von menschlichem Wissen in semantische Netze ist für einen Experten recht einfach, denn der Formalismus entstand ja gerade als Modell des menschlichen Gedächtnisses.

Kompaktheit: Der Vererbungsmechanismus ermöglicht eine sehr kompakte Form der Wissensrepräsentation.

Gleichwohl haben semantische Netze im allgemeinen auch einige Schwächen, die sich aus der völligen Freiheit in der Definition der Knoten und Kanten ergeben:

- Es existiert keine einheitliche Regelung oder allgemeine Übereinkunft, welches Wissen in Knoten und welches in Kanten zu repräsentieren ist. Nur die Kantentypen *hat_ein*, *ist_ein* und *hat_Instanz* bilden — wie bereits erwähnt — eine Ausnahme.

- Den Knoten und Kanten ist keine allgemeine Bedeutung zugeordnet — ihre Semantik ist völlig frei definierbar. Daher lassen sich auch keine allgemeinen Inferenz– oder Ableitungsregeln zur Wissensnutzung aufstellen, und es kann kein allgemeingültiges Abarbeitungsverfahren zur Problemlösung angegeben werden.

- Der Entwickler eines semantischen Netzes hat selbst darauf zu achten, daß eine logische Konsistenz im Netz herrscht, denn der Formalismus selbst kann dies nicht garantieren.

Der ERNEST++–Formalismus basiert auf den Ideen der semantischen Netze. Allerdings werden einige Definitionen und Restriktionen festgelegt. Beispielsweise können mit den Kanten nicht beliebige Relationen zwischen den Knoten ausgedrückt werden. Zwar bedeutet dieses eine Einschränkung in der Modellierungsfreiheit, es bringt aber den großen Vorteil mit sich, daß Regeln zur Wissensnutzung definiert werden können, die unabhängig von den Gegebenheiten eines speziellen Netzes in allen ERNEST++–Netzen anwendbar sind. Somit existiert eine Grundlage für eine problemunabhängige Verarbeitungsstrategie, die eine Analyse von Sprache oder Bildern mit Hilfe von semantischen Netzen ermöglicht. ERNEST++ bleibt also (genauso wie schon ERNEST) nicht bei der reinen Repräsentation von Wissen stehen, sondern ist explizit als Werkzeug zur Musteranalyse konzipiert.

4.2 Grundlegende Komponenten von ERNEST++

Wie in allen semantischen Netzwerkformalismen bilden die Knoten und Kanten auch in ERNEST++ die grundlegenden Komponenten zur Wissensrepräsentation. Sie werden daher zunächst vorgestellt. Zur Modellierung etwas komplexerer Sachverhalte reichen sie jedoch bei weitem nicht aus. Die hierfür im wesentlichen benötigten Komponenten stelle ich anschließend vor. Anschaulich machen möchte ich die Komponenten anhand eines Beispielnetzes, das in der Abbildung 4.2 dargestellt ist. Das Beispielnetz könnte zur Analyse von Bildern dienen, auf denen Bäume zu sehen sind. Zu diesem Zweck ist im Netz einiges Wissen über Bäume, ihre Gestalt und ihre

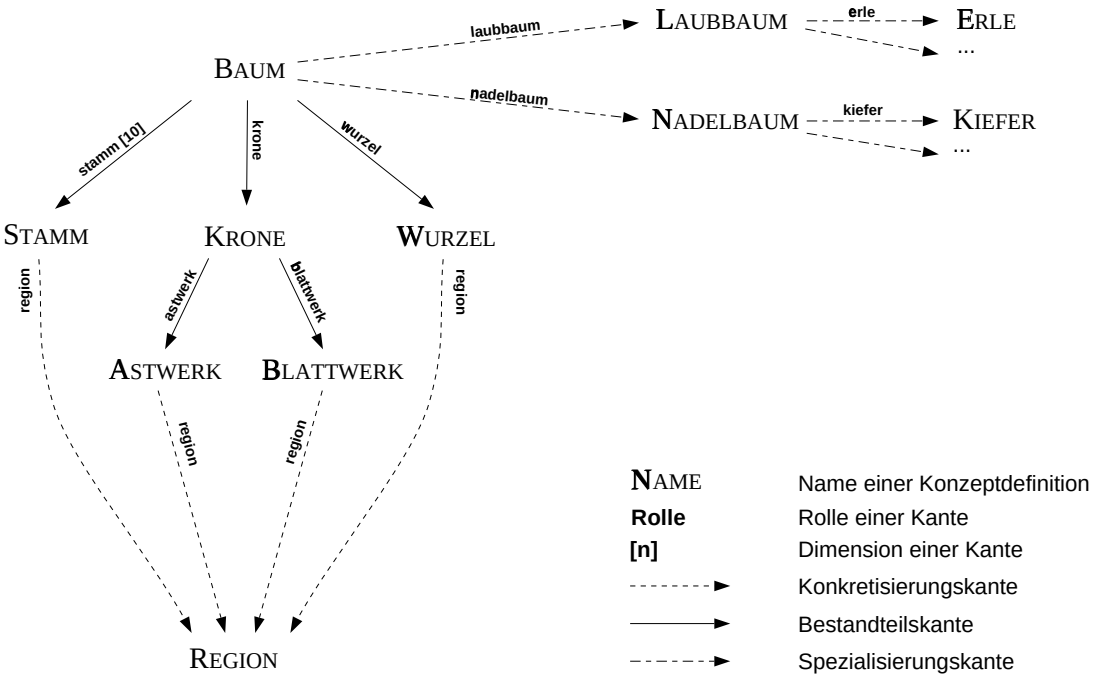

Abbildung 4.2: Überblick über ein ERNEST++–Netz zur Wissensrepräsentation

mögliche Verwendung eingetragen. Es wird im weiteren noch näher erläutert.

Konzeptdefinitonen und Netzknoten

Die Modellierung eines Begriffs beginnt in ERNEST++ mit der Definition des Begriffs in einer *Konzeptdefinition*. Sie enthält all das Wissen über den Begriff, das man modellieren möchte. In Abbildung 4.3 ist einführend ein vergröberter Auszug aus der Konzeptdefinition für den Begriff „Baum“ zu sehen. Sie beschreibt zunächst, daß ein Baum aus Stamm, Krone und Wurzel besteht und in zwei speziellen Arten auftritt — als Laubbaum und als Nadelbaum. Jeder Baum besitzt zwei weitere, interne Eigenschaften, nämlich seine Höhe und die Zugehörigkeit zu einer Pflanzenfamilie.

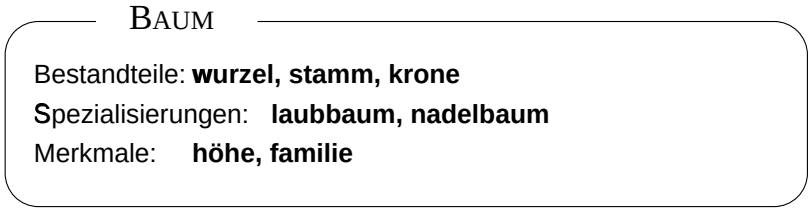

Abbildung 4.3: Ausschnitt aus der Konzeptdefinition zum Begriff „Baum“

Die Gesamtheit der Konzeptdefinitionen bestimmen die Struktur der Wissensbasis und sind eine Grundlage für die Analyse. Während der Abarbeitung eines Sensorsignals entstehen *Netzknoten* zu den Konzeptdefinitionen. Sie entsprechen den Knoten in einem allgemeinen semantischen Netz.[2] Im Unterschied zu allgemeinen semantischen Netzen sind in ERNEST++ allerdings drei Typen von Netzknoten festgelegt. *Konzepte* sind diejenigen Netzknoten, die keine weiteren Informationen als die Konzeptdefinition tragen. Sie umfassen nur die allgemeine Beschreibung des repräsentierten Begriffs und seiner Eigenschaften, unabhängig von dem konkreten Signal. Das Ziel der Analyse besteht darin, den Konzeptdefinitionen Signalausschnitte zuzuordnen, das heißt für die allgemeinen Beschreibungen konkrete Werte zu berechnen. Diese konkreten Werte sind in einer *Instanz* enthalten. Während also in einem Konzept zu BAUM festgehalten ist, daß jeder Baum einer Pflanzenfamilie angehört, ist in einer Instanz zu BAUM ein konkreter, vom jeweiligen Sensordatum abhängiger und aus ihm berechneter Wert eingetragen, beispielsweise *Rosengewächs*. Somit ist aus dem Konzept eine Instanz geworden — das Konzept wurde *instantiiert*. In den meisten Fällen ist es nicht möglich, daß die Analyseergebnisse in einem Schritt berechnet werden. *Modifizierte Konzepte* repräsentieren daher die während der Analyse eines Eingangssignals erreichten Zwischenergebnisse. Wenn also auf einem Bild zwar schon die Kro-

[2] Die Einführung des Begriffs „Netzknoten“ ist deshalb nötig, weil sie unterschieden werden müssen von *Suchbaumknoten*, die es in ERNEST++ ebenfalls gibt (siehe Abschnitt 4.4). Auf welche Art und Weise die Netzknoten entstehen ist ebenfalls Gegenstand des Abschnittes 4.4 und soll im Moment unberücksichtigt bleiben.

ne eines Baumes, der Stamm aber noch nicht detektiert wurde, so ist dieses Zwischenergebnis in einem modifizierten Konzept von BAUM vermerkt.

Im weiteren gelte folgende Schreibweise: wenn „Baum“ der zu modellierende Begriff ist, dann denotiert

BAUM	die zum Begriff gehörende Konzeptdefinition,
K_i(BAUM)	das i-te zu BAUM gehörende Konzept,
MK_j(BAUM)	das j-te zu BAUM gehörende modifizierte Konzept,
I_l(BAUM)	die l-te zu BAUM gehörende Instanz und
N_k(BAUM)	der k-te zu BAUM gehörende Netzknoten, unabhängig davon, welchen Typ er besitzt.

Kanten

Im Unterschied zu allgemeinen semantischen Netzen stellt ERNEST++ dem Modellierer nur drei verschiedene Kantentypen mit einer festgelegten Bedeutung zur Verfügung.

Bestandteilskanten realisieren die Dekomposition eines Begriffs in Subbegriffe. Im Beispielnetz in Abbildung 4.2 ist die Beziehung von BAUM zu STAMM, KRONE und WURZEL folglich durch Kanten diesen Typs ausgedrückt. Eine Bestandteilskante ist also mit der allgemein bekannten hat_ein–Kante vergleichbar. In ERNEST++ besteht die Möglichkeit, eine Bestandteilskante als *kontextabhängig* zu markieren. Damit wird zum Ausdruck gebracht, daß der Zielknoten der Kante stets im Kontext des Ursprungsknotens zu sehen ist. So sollte im Beispielnetz die Kante von BAUM zu KRONE als kontextabhängig gekennzeichnet sein, um zu verdeutlichen, daß eine Baumkrone nur im Kontext eines Baumes als solche zu identifizieren ist. Nur über kontextabhängige Bestandteilskanten kann aus dem Zielknoten heraus Wissen, welches im Ursprungsknoten enthalten ist, abgerufen werden. Ein modifiziertes Konzept MK_i(KRONE) könnte beispielsweise in einem mit ihm verbundenen modifizierten Konzept MK_j(BAUM) nachsehen, ob für dieses bereits die Pflanzenfamilie bestimmt werden konnte und diese Information bei der eigenen Instantiierung nutzen.

Soll ein genereller Begriff spezialisiert werden, sind *Spezialisierungskanten* zu verwenden. Im Netz in Abbildung 4.2 wird also ausgesagt, daß eine Kiefer ein ganz bestimmter Nadelbaum ist. Wie über die ist_ein–Kante in anderen semantischen Netzwerkformalismen werden in ERNEST++ die Eigenschaften eines Begriffs durch Spezialisierungskanten an den Zielknoten der Kante vererbt.

Konkretisierungskanten erleichtern den Aufbau von wohlstrukturierten Wissensbasen, indem sie dem Modellierer die Möglichkeit bieten, verschiedene Abstraktionsebenen einzuführen. Begriffe, die zu einer Begriffswelt gehören, können somit gruppiert werden. Beispielsweise finden die

abstrakten Konzeptdefinitionen STAMM, KRONE und WURZEL ihre jeweilige Entsprechung in der Konzeptdefinition REGION, die eine zusammenhängenden Region gleicher Farbe und Textur auf einem Bild repräsentiert. Im weiteren verwende ich folgende Bezeichnung: eine Konzeptdefinition A ist *abstrakter* als eine Konzeptdefinition B, wenn sie direkt oder indirekt über eine Konkretisierungskante mit B verbunden ist. Wenn A abstrakter ist als B, dann ist B *konkreter* als A.

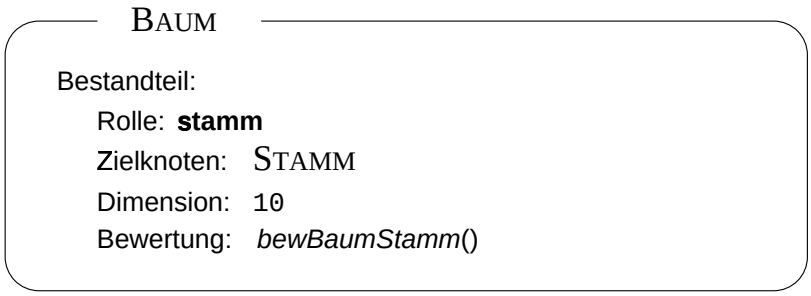

Abbildung 4.4: Definition der Kante *stamm* in der Konzeptdefinition BAUM

In Abbildung 4.4 ist die Definition einer Kante innerhalb einer Konzeptdefinition anhand der Kante von BAUM zu STAMM dargestellt. Jede Kante hat zu ihrer eindeutigen Identifizierung eine Rolle. Die Kante mit der Rolle „stamm“ in der Konzeptdefinition BAUM wird von nun an als BAUM.*stamm* aufgeschrieben. In der Kante ist außerdem der Name des Zielknotens festgehalten. Weil es Bäume mit mehr als einem Stamm gibt, ist diese Kante *hochdimensional*, das heißt mehrere (bis zu zehn) Netzknoten N_i(STAMM) können über genau eine Kante mit einem Netzknoten N_j(BAUM) verbunden sein. In der Bewertungsfunktion *bewBaumStamm* wird beurteilt, wie gut der oder die Zielknoten der Kante zum Ursprungsknoten passen. Ist also zum Beispiel als Familie in N_j(BAUM) bereits *Buchengewächs* ermittelt, sollten die Eigenschaften von N_i(STAMM) dem nicht widersprechen.

In den folgenden Abbildungen dieses Kapitels, die Ausschnitte aus Konzeptdefinitionen oder Netzknoten wiedergeben, werden dieselben Schrifttypen und –arten wie in Abbildung 4.4 verwendet. Sie bedeuten im einzelnen:

NAME	Namen von Konzeptdefinitionen oder Netzknoten
rolle	Rollen von ERNEST++–Komponenten
`Wert`	Werttypen oder Werte
Funktion	Namen von Funktionen

Merkmale

Eigenschaften von Begriffen werden in ERNEST++ durch *Merkmale* ausgedrückt, die in den Konzeptdefinitionen enthalten sind. Die Berechnung der Merkmale ist ein wesentlicher Beitrag zur

Erreichung des Ziels der Analyse. Denn durch ihre Berechnung erfolgt die Ermittlung von konkreten Werten für die Instanzen. Daher macht insbesondere das in den Berechnungsfunktionen enthaltene Wissen einen Großteil der Leistungsfähigkeit des gesamten Analysesystems aus.

Abbildung 4.5 zeigt beispielhaft die Merkmalsdefinitionen in der Konzeptdefinition STAMM. Das Merkmal *länge* beschreibt die Länge eines Stammes, *festigkeit* die Festigkeit seine Holzes und *verwendung* bestimmt schließlich, ob aus dem Stamm ein Möbelstück angefertigt werden kann oder ob er besser zu anderen Zwecken verwendet wird. Im Merkmal *lage* wird die Lage des Stammes im Bild durch seinen Schwerpunkt charakterisiert. Dieses Merkmal dient also ausschließlich zu Analysezwecken, denn der Schwerpunkt eines Stammes gehört im allgemeinen sicher nicht zu seinen wesentlichen Eigenschaften. Jedes Merkmal besitzt eine *Rolle* zur eindeutigen Identifikation und Erläuterung der funktionalen Rolle. Die Rolle wird auch zur Notation verwendet: das Merkmal „länge“ in der Konzeptdefiniton STAMM soll nun als STAMM.*länge* aufgeschrieben werden. Der *Werttyp* eines Merkmals legt die Wertemenge der Berechnungsfunktion fest. Manchmal sollen aber nicht alle Werte der Wertemenge als Ergebnis der Berechnung in Frage kommen. Mit dem Eintrag *Restriktion* besteht daher die Möglichkeit, a priori den

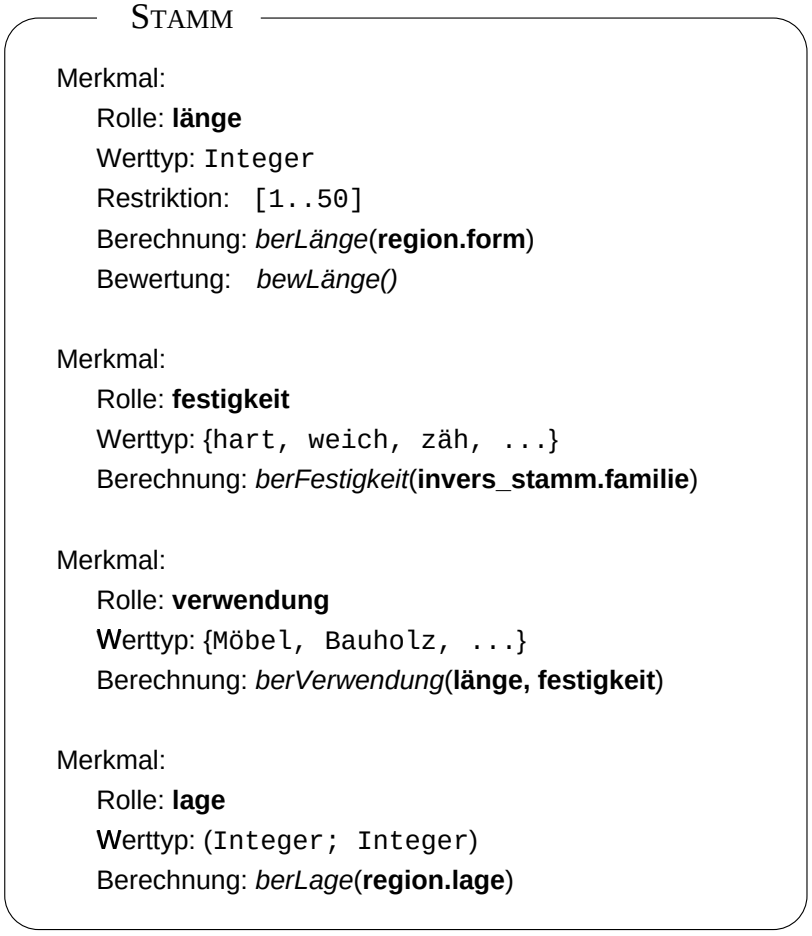

Abbildung 4.5: Definition der Merkmale in der Konzeptdefinition STAMM

Wertebereich eines Merkmals einzuschränken. So wird in dem Merkmal STAMM.*länge* ausgesagt, daß als Werte für die Merkmalsberechnung nur Zahlen zwischen eins und fünfzig sinnvoll sind. In jedem Merkmal ist im Eintrag *Berechnung* die Funktion anzugeben, welche die Berechnung durchführt. Ihre Argumente werden mit Hilfe der Rollen von anderen ERNEST–Komponenten spezifiziert. Die Argumente können Merkmale aus der selben Konzeptdefinition sein (wie zum Beispiel im Merkmal *verwendung*) oder aus Konzeptdefinitionen stammen, die Bestandteile oder Konkretisierungen der Konzeptdefiniton sind, zu dem das Merkmal gehört: STAMM.*länge* bekommt über die Kante mit der Rolle *region* das Merkmal mit der Rolle *form* von REGION als Argument. Außerdem kann, wie bereits erwähnt, auf Konzeptdefinitionen zugegriffen werden, die einen Kontext etablieren. Über die Kante *invers_stamm* greift das Merkmal *festigkeit* in der Konzeptdefinition STAMM auf das Merkmal BAUM.*familie* als Argument der Berechnungsfunktion zu. Die Eintrag *Bewertung* gibt an, wie gut der berechnete Wert zu den Erwartungen für dieses Merkmal paßt. In *bewLänge* kann beispielsweise beurteilt werden, wie gut die berechnete Stammlänge innerhalb der angegebenen Restriktion liegt.

Relationen

Der Vergleich von Eigenschaften ist oftmals ein angemessenes Mittel, um zusätzliche Informationen zu erhalten oder Plausibilitäten zu überprüfen. In ERNEST wird ein solcher Vergleich durch *Relationen* bewerkstelligt. In Relationen können Merkmalswerte miteinander verglichen werden. Als Ergebnis der Relationsberechnung wird während der Analyse in einer Bewertungsstruktur repräsentiert, wie gut die Werte in den Merkmalen der Netzknoten die angenommene Beziehung zwischen ihnen erfüllen. Beispielsweise kann die in BAUM eingefügte Relation *anordnung* (siehe Abbildung 4.6) überprüfen, ob die Anordnung der im Bild gefundenen Bestand-

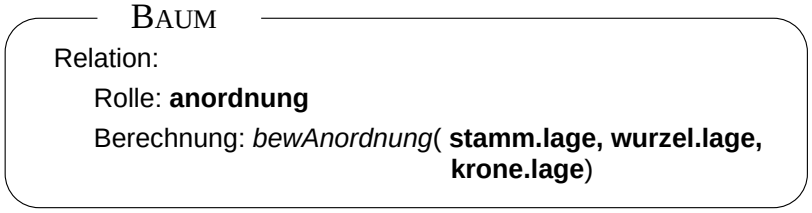

Abbildung 4.6: Definition der Relation *höhe_plausibel* in der Konzeptdefinition BAUM

teile des Baumes plausibel ist. Dazu greift *bewAnordnung* auf die Schwerpunkte der Bestandteile zu und vergleicht sie. Liegt etwa der Schwerpunkt der Wurzel oberhalb des Schwerpunktes des Stammes, scheint während der Analyse eine fehlerhafte Zuordnung der Regionen vonstatten gegangen zu sein. Dann sollte das Gütemaß in der Bewertungsstruktur einen entsprechend schlechten Wert zugewiesen bekommen.

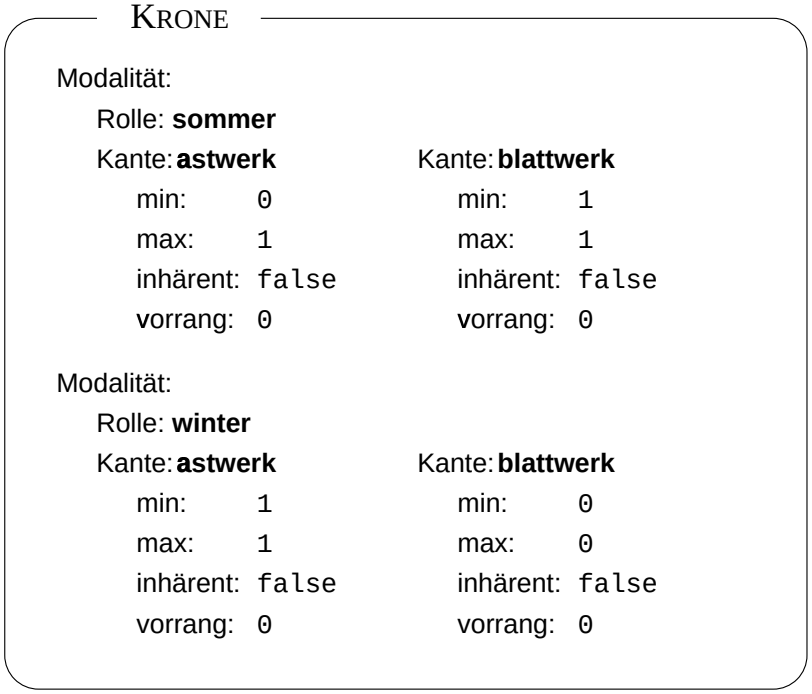

Abbildung 4.7: Definition der Modalitäten in der Konzeptdefinition KRONE

Modalitäten

Begriffe können verschiedene Ausprägungen haben. Die Krone eines Laubbaumes besteht im Winter nur aus Ästen, während im Sommer nur Blätter, eventuell um einige Äste ergänzt, zu sehen sind. Die Kanten zwischen KRONE, ASTWERK und BLATTWERK sind also nicht immer alle erforderlich — zur Modellierung einer Krone im Winter braucht man keine Kante KRONE.*blattwerk*, weil es keine Blätter gibt. ERNEST bietet zur kompakten Modellierung der Ausprägungen eines Begriffs die *Modalitäten* an. In ihnen wird für jede Kante festgehalten, ob sie obligatorisch, optional oder inhärent ist. In KRONE sind daher die Modalitäten wie in Abbildung 4.7 angegeben. In der Modalität KRONE.*sommer*[3] wird bestimmt, daß die Kante *blattwerk* genau einmal vorkommt. Dies geschieht, indem die minimale Anzahl *min* genauso wie die maximale Anzahl *max* den Wert eins erhält. Die Kante *astwerk* ist dagegen in dieser Modalität als optional markiert, denn in *min* ist der Wert null und in *max* der Wert eins eingetragen. In der Modalität KRONE.*winter* wird die Kante *blattwerk* als nicht existent markiert, indem die Einträge *min* und *max* den Wert null erhalten. Der Eintrag *inhärent* hat dann den Wert *true*, wenn man modellieren möchte, daß der Zielknoten der Kante nicht ausdrücklich vorzuliegen braucht. So könnte in einer Modalität von BAUM festgehalten werden, daß die Kante BAUM.*wurzel* inhärent ist. Damit würde ausgedrückt, daß ein Baum auf alle Fälle eine Wurzel besitzt — selbst wenn diese auf einem Bild nicht zu sehen ist. Der Modalitäteneintrag *vorrang* beeinflußt die in Ab-

[3] Analog zu den Kanten und Merkmalen wird im weiteren eine Modalität mit der Rolle „rolle“ in der Konzeptdefinition BEGRIFF als BEGRIFF.*rolle* aufgeschrieben.

schnitt 4.4 dargelegte Verarbeitungsstrategie.

Bewertungen

Eine große Bedeutung kommt in Systemen zur Musteranalyse der Bewertungsproblematik zu. Die signalnahen Module (beispielsweise die Bildaufnahme oder die Berechnung von Regionen gleicher Farbe) können vor allem aufgrund von äußeren Einflüssen (zum Beispiel Lichtverhältnisse) ungenaue oder sogar fehlerhafte Ergebnisse produzieren. Folglich sind die der Interpretation zugrunde liegenden Daten stets mit einer gewissen Unsicherheit behaftet. Dieser Unsicherheit wird in ERNEST und ERNE$\overset{++}{\text{S}}$T mit einer Bewertungsstruktur Rechnung getragen. In ERNEST steht dem Entwickler eines Netzes ein zehnelementiger Vektor zur Bewertung der Analysesituation zur Verfügung. Die Bedeutung der einzelnen Komponenten des Bewertungsvektors kann in jedem Netz unterschiedlich sein. ERNE$\overset{++}{\text{S}}$T nutzt die Möglichkeiten der objektorientierten Implementation zur konsequenten Weiterentwicklung dieses Gedankens aus und erlaubt dem Entwickler die freie Definition von Bewertungsklassen. In ein und demselben Netz können durchaus mehrere solcher Bewertungsklassen festgelegt werden[4], denn es müssen ja ganz unterschiedliche Aspekte bewertet werden, beispielsweise die Güte von Merkmalsberechnungen, die jeweiligen Netzknoten als Ganzes oder der Fortgang der Analyse insgesamt.

4.3 Inferenzregeln

Nachdem ich im letzten Abschnitt die wichtigsten ERNE$\overset{++}{\text{S}}$T–Komponenten zur Wissensrepräsentation erläutert habe, stellt sich nun die Frage, wie das modellierte Wissen zu nutzen ist, das heißt, in welcher Weise die Zuordnung der Konzeptdefinitionen zu dem Sensorsignal geschieht. In ERNE$\overset{++}{\text{S}}$T existieren sechs Regeln zur Wissensnutzung. Sie fußen ausschließlich auf den definierten Netzknoten– und Kantentypen und können daher in jedem ERNE$\overset{++}{\text{S}}$T–Netz angewendet werden. Die Inferenzregeln sind also problemunabhängig und eliminieren somit einen der genannten Nachteile der Modellierung mit semantischen Netzen. Die Inferenzregeln realisieren drei Arten der Wissensnutzung:

1. Erzeugung und Erweiterung von Instanzen
   Etwas vergröbert ausgedrückt, kann eine Instanz immer dann gebildet werden, wenn die Zielknoten aller als obligatorisch deklarierten Kanten selbst Instanzen sind. Eine Instanz kann erweitert werden, falls Instanzen existieren, die über optionale Kanten mit ihr verbunden werden können. Eine Ausnahme bilden die sogenannten *holistischen Instanzen*. Als Erweiterung des Sprachumfangs von ERNEST besteht in ERNE$\overset{++}{\text{S}}$T nämlich die Möglichkeit, die mit einem holistischen Erkenner erzielten Ergebnisse direkt als Instanz einzubinden (siehe Seite 63).

[4] Der Entwickler muß lediglich die Vergleichs– und Kombinationsoperatoren für die Klassen definieren.

2. Datengetriebene Erzeugung von modifizierten Konzepten
   Diese Regel besagt, daß ein neues modifiziertes Konzept $MK_n(A)$ aus einem modifiziertes Konzept $MK_j(A)$ oder einem Konzept $K_i(A)$ immer dann berechnet werden kann, wenn ein neuer Netzknoten $N_l(B)$ über eine Bestandteils– oder Konkretisierungskante an $MK_j(A)$ beziehungsweise $K_i(A)$ gebunden wurde. Bei der Berechnung von $MK_n(A)$ wird das in $N_l(B)$ enthaltene Wissen genutzt. Durch die datengetriebene Erzeugung von modifizierten Konzepten wird also Information von signalnäheren Netzknoten an abstraktere Netzknoten gegeben. Sie wird daher auch *Bottom–Up–Regel* genannt.
3. Modellgetriebene Erzeugung von modifizierten Konzepten
   Wird ein Konzept $K_i(B)$ oder ein modifiziertes Konzept $MK_j(B)$ an einen bestehenden abstrakteren Netzknoten $N_l(A)$ über eine Bestandteils– oder Konkretisierungskante gebunden, so führt dies zu der Berechnung eines neuen modifizierten Konzeptes $MK_n(B)$, bei der die in dem abstrakteren Netzknoten $N_l(A)$ enthaltene Information nach „unten“, daß heißt zu dem konkreteren modifizierten Konzept hin, propagiert wird. Diese Regel heißt deswegen auch *Top–Down–Regel*.

Bei der Erzeugung beziehungsweise Erweiterung modifizierter Konzepte oder Instanzen werden die in ihnen enthaltenen Merkmale und Relationen neu berechnet und bewertet. Außerdem werden die Bewertungsfunktionen für die Kanten und für den Netzknoten insgesamt angestoßen. Auf diese Weise beinhalten die in einer Analysesituation aktuell enthaltenen modifizierten Konzepte und Instanzen stets den derzeit erreichten Stand der Analyse.

4.4 Verarbeitungsstrategie

In ERNEST wird die Analyse von Sensorsignalen als Suchproblem aufgefaßt. Jeder Zustand der Analyse wird repräsentiert in einem Ensemble von Netzknoten, deren jeweilige Gesamtheit einen Knoten im Analyse–Suchbaum[5] bilden. So entstehen Situationen, in denen die im vorigen Abschnitt vorgestellten Regeln anwendbar sind. Oftmals können in einem Suchbaumknoten aber mehrere Regeln gleichzeitig angewandt werden. Damit die Konsistenz der Analysesituation gewahrt bleibt, ist die grundlegende Verarbeitungsstrategie jedoch so angelegt, daß stets nur eine Regel ausgeführt wird. Nach der Ausführung der Regel werden die dabei neu entstehenden Suchbaumknoten bewertet. Auf Grundlage dieser Bewertung findet der A*–Algorithmus [Nil71] Anwendung, der bei einer optimistischen Restschätzung der anfallenden Kosten eine optimale Lösung garantiert.

[5] Im weiteren muß also immer genau unterschieden werden zwischen den *Netz*knoten (Konzepte, modifizierte Konzepte und Instanzen) und den *Suchbaum*knoten, die eine Analysesituation widerspiegeln!

◇ Funktion: ERNEST++BasisKontrolle

▷ Parameter: Konzeptdefinitionen *K*

```
OFFEN := Suchbaum_Initialisierung(K);
initialisiere_problemabhängig();
while (OFFEN ≠ ∅) do
    sbk_i := wähle_bestbewerteten_Suchbaumknoten(OFFEN);
    if (Analyseziel_erreicht(sbk_i)==TRUE) then
        präsentiere_Analyseergebnis(sbk_i);
        Analyse_Terminierung();
    elseif (datengetriebene_Bindung_erwünscht(sbk_i) == TRUE) then
        sbk_i.Datengetriebene_Bindung();
    elseif (ungebundene_Konzepte_erwünscht(sbk_i)==TRUE) then
        sbk_i.Ergänzung_um_ungebundene_Konzepte();
    elseif (sbk_i.holistische_Instantiierung_möglich() == TRUE) then
        sbk_i.Holistische_Instantiierung();
    elseif (holistische_Dekomposition_erwünscht(sbk_i) == TRUE) then
        sbk_i.Holistische_Dekomposition();
    elseif (sbk_i.Instantiierung_möglich() == TRUE) then
        sbk_i.Netzknoten_Instantiierung();
    elseif (sbk_i.modellgetriebene_Bindung_möglich() == TRUE) then
        sbk_i.Modellgetriebene_Bindung();
    else
        sbk_i.Expansion_optionaler_Kanten_und_Spezialisierungen();
Analyse_Terminierung();
```

Algorithmus 4.1: ERNEST++–Basiskontrolle

In Algorithmus 4.1 wird dargelegt, wie ein Modellierer in ERNEST++ auf die Verarbeitungsstrategie Einfluß nehmen kann und wann welche Regel angewendet wird. Dieser Algorithmus wird auch ERNEST++*–Basiskontrolle* genannt. Die Analyse beginnt mit der Initialisierung des Suchbaums. Dabei werden modifizierte Konzepte zu vom Modellierer initial angegebenen Konzeptdefinitionen berechnet, entsprechende Suchbaumknoten angelegt und der Liste *OFFEN* zugefügt. Somit befinden sich dann nur die Ausgangsknoten der Analyse, die sogenannten *Wurzelknoten*, in der Liste OFFEN. Danach erhält der Modellierer mit der Funktion *initialisiere_problemabhängig*[6] die Möglichkeit, eine problembezogene Initialisierung, beispielsweise das Einlesen eines Bildes, durchzuführen. Während der Analyse wird die Liste OFFEN

[6] Funktionen, die ein Modellierer zu schreiben hat, sind in Algorithmus 4.1 *kursiv* gedruckt.

in einer while–Schleife, der *Basisschleife*, abgearbeitet. Zunächst wird mittels der Funktion *wähle_bestbewerteten_Suchbaumknoten* der Suchbaumknoten sbk_i ausgewählt. Er repräsentiert die aktuell bestbewertete Interpretation des Eingangssignals. Falls dieses Analyseergebnis bereits dem Analyseziel entspricht, kann das Ergebnis präsentiert werden und der Algorithmus terminiert. Im anderen Fall wird genau eine *Basisaktion* zur Erweiterung der Interpretation in sbk_i durchgeführt. Zum Abschluß einer jeden Basisaktion werden die dabei entstandenen Suchbaumknoten bewertet und der Liste OFFEN hinzugefügt. Sodann wird die Basisschleife erneut ausgeführt. Ist die Prämisse der Basisschleife nicht mehr erfüllt, konnte das Analyseziel nicht erreicht werden. Im folgenden werden die ERNEST–Basisaktionen in der Reihenfolge ihres Auftretens in der Basisschleife dargelegt.

Datengetriebene Bindung

Aufgabe der datengetriebenen Bindung ist die Herstellung einer Verbindung zwischen einem Netzknoten N_i(A) zu einem abstrakteren Netzknoten N_j(B). In Abbildung 4.8 ist ein Beispiel zu sehen. Entscheidet sich der Algorithmus durch die problemabhängige Funktion *datengetrie-*

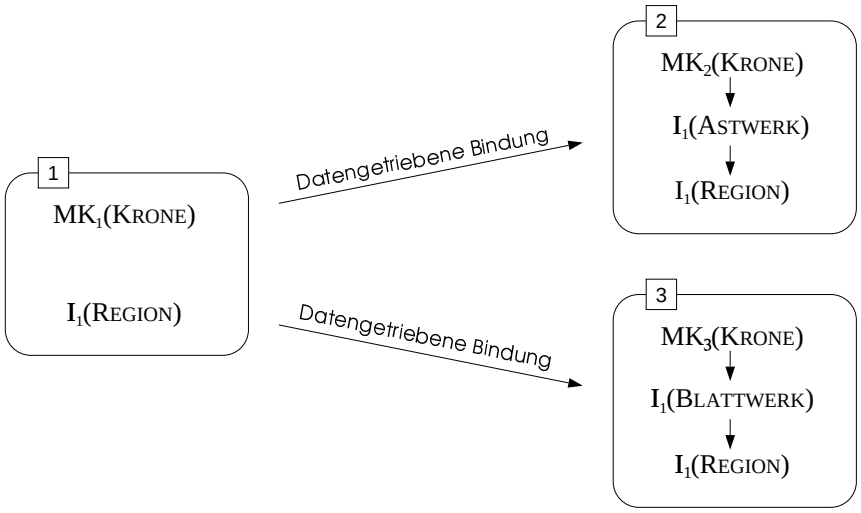

Abbildung 4.8: Datengetriebene Bindung eines Netzknotens

bene_Bindung_erwünscht im Suchbaumknoten [1], die Instanz I_1(REGION) an das modifizierte Konzept MK_1(KRONE) zu binden, so ergeben sich zwei Möglichkeiten, die zu den konkurrierenden Nachfolgern [2] und [3] des Ausgangs–Suchbaumknotens führen. In [2] ist der Pfad über ASTWERK, in [3] der alternative Weg über BLATTWERK gegangen worden. Jeder dieser Wege besteht aus zwei Einzelschritten: von REGION zu ASTWERK beziehungsweise BLATTWERK und dann weiter von dem neu entstandenen Netzknoten zu dem modifizierten Konzept von KRONE, das jeweils neu berechnet wird. Vor jedem Einzelschritt prüft eine problemabhängige Funktion, ob der beabsichtigte Weg eingeschlagen werden soll. Bei der Expansion der Einzelschritte werden die jeweiligen Netzknoten erzeugt und die entsprechende Regel aktiviert. Im

Beispiel können also ASTWERK und BLATTWERK jeweils instantiiert werden. Existiert das angegebene Ziel der Bindung noch nicht, so wird automatisch ein neues modifiziertes Konzept dieses Ziels erzeugt. Für das Beispiel bedeutet dies: gäbe es MK_1(KRONE) nicht, *datengetriebene_Bindung_erwünscht* fordert aber trotzdem eine Bindung von I_1(REGION) an KRONE, würde ein neues modifiziertes Konzept von KRONE erzeugt und I_1(REGION) an dieses gebunden.

Ergänzung um ungebundene Konzepte

Eine andere Möglichkeit, den aktuellen Suchbaumknoten zu erweitern, besteht darin, ungebundene Konzepte hinzuzufügen. Mit der Funktion *ungebundene_Konzepte_erwünscht* werden die Namen der Konzepte spezifiziert, die dem Suchbaumknoten einfach hinzugefügt werden (in Abbildung 4.9 beispielsweise K_2(REGION)). Erst wenn der Suchbaumknoten zu einem späteren

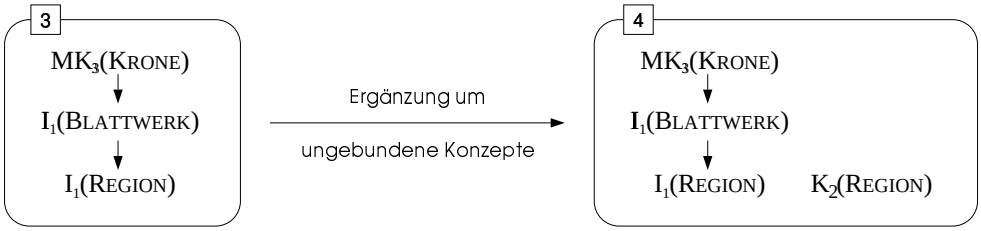

Abbildung 4.9: Ergänzung um ungebundene Konzepte

Zeitpunkt in der Basisschleife erneut zur Bearbeitung kommt, wird entschieden, ob und gegebenenfalls in welcher Art und Weise der neue Netzknoten in die Analyse eingebunden wird.

Holistische Instantiierung

Das Wesen eines holistischen Erkenners besteht darin, daß er ein Objekt als einheitliches Ganzes erkennt. Er erkennt keine Substrukturen des Objektes, auf das er trainiert ist. Im Zusammenspiel mit der Analyse durch ERNEST$^{++}_{+}$ eignen sich holistische Erkenner daher besonders dazu, Ergebnisse für relativ abstrakte Konzeptdefinitonen zu produzieren, die später bei Bedarf verfeinert werden können. Dazu muß ein solcher Erkenner in eine Konzeptdefinition eingetragen werden, für die dann holistische Instanzen gebildet werden können. Stellt die Basisschleife durch die Funktion *holistische_Instantiierung_möglich* fest, daß ein Konzept oder modifiziertes Konzept zu einer solchen Konzeptdefinition in dem aktuellen Suchbaumknoten vorliegt, wird der holistische Erkennungsprozeß durchgeführt und sein Ergebnis in einer holistischen Instanz festgehalten. Hätte man zum Beispiel ein künstliches neuronales Netz, das Baumkronen in einem Bild erkennt, so könnte dessen Ergebnis — wie in Abbildung 4.10 zu sehen — direkt von einem modifizierten Konzept MK_m(KRONE) zu einer holistischen Instanz I^h_n(KRONE) führen[7], ohne

[7] Die Notation erfolgt in Analogie zu der auf Seite 54 vorgestellten Schreibweise, wobei das hochgestellte „h“ für „holistisch“ steht.

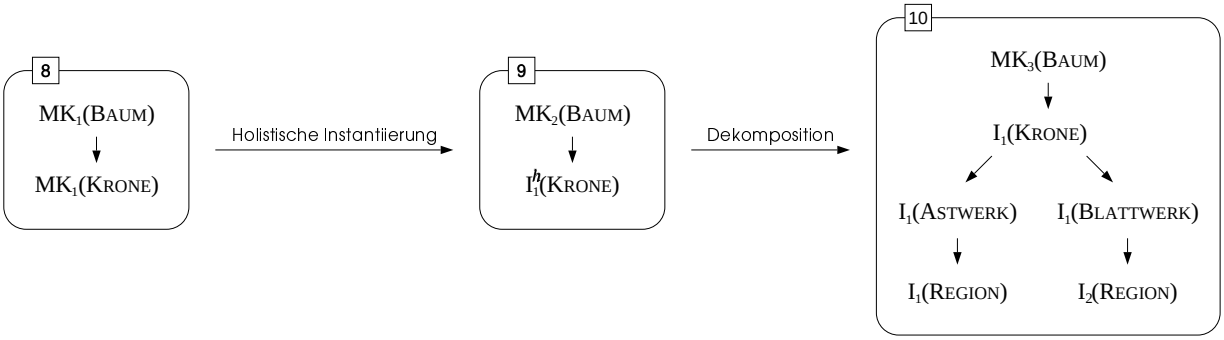

Abbildung 4.10: Holistische Instantiierung und Dekomposition einer holistischen Instanz

daß Instanzen zu ASTWERK und BLATTWERK vorliegen müssen. In der weiteren Abarbeitung der Basisschleife wird I_n^h(KRONE) wie jede andere Instanz behandelt.

Holistische Dekomposition

Reicht das in einer holistischen Instanz I_n^h(H) enthaltene ganzheitliche Resultat nicht aus, so kann diese Instanz dekomponiert werden. Dabei werden die in den Konzeptdefinitionen festgelegten Bestandteile und Konkretisierungen modellgetrieben bis zu den Signalkonzepten erzeugt. Das in I_n^h(H) enthaltene Analyseergebnis wird dabei zur Restringierung der neu entstehenden Netzknoten genutzt. Abbildung 4.10 zeigt beispielhaft die Dekomposition von I_1^h(KRONE).

Instantiierung

Kann ein Konzept oder modifiziertes Konzept instantiiert werden, so wird es in den nachfolgenden Suchbaumknoten durch die neu berechneten Instanzen ersetzt. Zusätzlich wird die in der jeweiligen Instanz neu gewonnene Information in alle anderen Netzknoten des Suchbaumknotens propagiert. Auf diese Weise können Restriktionen in die Netzknoten eingetragen werden, welche unter Umständen die weitere Analyse wesentlich effizienter gestalten. Das modifizierte

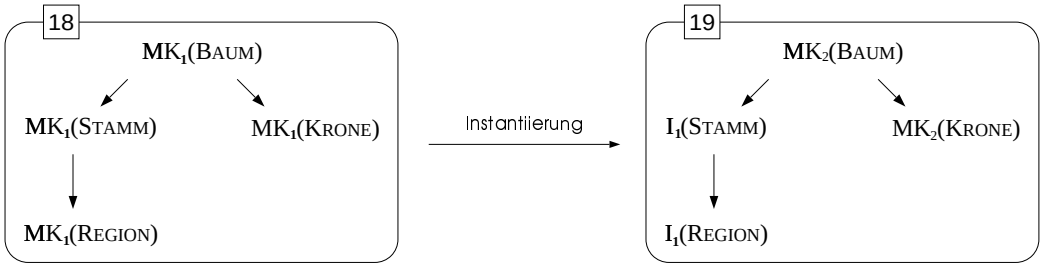

Abbildung 4.11: Instantiierung mit Restringierung angebundener Netzknoten

Konzept MK_1(REGION) im Knoten 18 der Abbildung 4.11 kann instantiiert werden. Die Propagierung der neuen Instanz I_1(REGION) in 19 hat zur Folge, daß auch die Instanz I_1(STAMM) berechnet werden kann, denn alle Konkretisierungen und Bestandteile von STAMM sind instantiiert. Die neue Information führt auch zu den neuen modifizierten Konzepten MK_2(BAUM) und

MK_2(KRONE). Beispielsweise kann in MK_2(KRONE) der Bereich, in dem nach einer Baumkrone gesucht werden soll, durch die Lage des Stammes im Bild eingeschränkt werden.

Modellgetriebene Bindung

Falls in der Basisschleife kein Netzknoten instantiiert werden kann, wird als nächstes überprüft, ob eine modellgetriebene Bindung eines Netzknotens durchgeführt werden kann. Das ist genau dann der Fall, wenn ein Netzknoten eine Modalität enthält, in der eine Bestandteils– oder Konkretisierungskante als obligatorisch markiert ist und diese Kante noch nicht expandiert wurde. Existiert im Suchbaumknoten kein Netzknoten, der als Zielknoten der Kante in Frage kommt, wird bei der modellgetriebenen Bindung dieser Kante ein neues modifiziertes Konzept erzeugt und an den Ursprungsknoten der Kante gebunden. Somit realisiert die modellgetriebene Bindung die Verbindung von abstrakten Konzepten hin zu signalnahen Konzepten. Falls bei der Berechnung des neuen modifizierten Konzeptes konkurrierende Ergebnisse berechnet werden, führen diese — wie in Abbildung 4.12 beispielhaft am modifizierten Konzept MK_1(WURZEL) dargestellt — zu konkurrierenden Nachfolgern des aktuellen Suchbaumknotens.

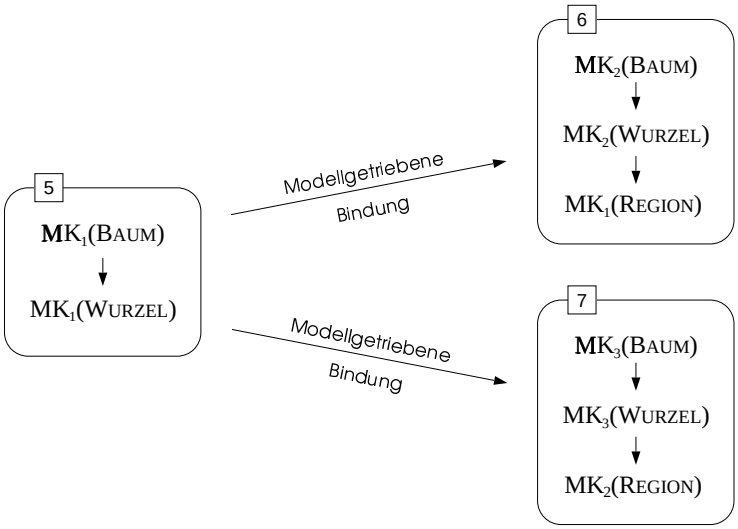

Abbildung 4.12: Modellgetriebene Bindung

Expansion optionaler Kanten und Spezialisierungen

Die letzte Möglichkeit zur Weiterentwicklung der Interpretation ist die Expansion von optionalen Kanten und Spezialisierungskanten. Algorithmus 4.2 zeigt das Verfahren, das in der Funktion *Expansion_optionaler_Kanten_und_Spezialisierungen* als Methode der Klasse *E_Suchbaumknoten*, welche die Suchbaumknoten repräsentiert, implementiert ist. Alle Instanzen $\mathcal{I}$ des Suchbaumknotens werden betrachtet. Zunächst geht es um die Expansion der optionalen Kanten. Die Funktion *hole_alle_optionalen_Kanten* erzeugt die Menge der zur Expansion anstehenden optionalen Kanten der Instanz I_j. Das sind all diejenigen Kanten, die bezüglich der

◇ Funktion: E_Suchbaumknoten::Expansion_optionaler_Kanten_und_Spezialisierungen

```
I := hole_alle_Instanzen();
for (I_j ∈ I )
    K := I_j.hole_alle_optionalen_Kanten()
    for (K_n ∈ K )
        if (Expansion_optionaler_Kante_erwünscht(this, I_j, K_n)) then
            expandiere_optionale_Kante(I_j, K_n);
for (I_j ∈ I )
    S := I_j.hole_alle_Spezialisierungskanten()
    for (S_n ∈ S )
        if (Expansion_Spezialisierungskante_erwünscht(this, I_j, S_n)) then
            expandiere_Spezialisierungskante(I_j, S_n);
```

Algorithmus 4.2: Algorithmus zur Expansion optionaler Kanten und Spezialisierungen in einem Suchbaumknoten

Modalität von I_j noch nicht vollständig expandiert[8] sind. Die problemabhängige Funktion *Expansion_optionaler_Kante_erwünscht* entscheidet unter Berücksichtigung des jeweiligen Suchbaumknotens (*this*) und der aktuell betrachteten Instanz I_j für alle Kanten K_n, ob eine Expansion durchgeführt werden soll. Die Expansion optionaler Kanten erfolgt dann analog der modellgetriebenen Bindung: der Knoten 13 in Abbildung 4.13 entsteht als Nachfolger von 11, indem die optionale Kante *wurzel* von I_1(BAUM) expandiert wurde. Nachdem alle optionalen Kanten abgearbeitet sind, werden in ähnlicher Weise die Spezialisierungskanten betrachtet. Für jede Spezialisierungskante entscheidet *Expansion_Spezialisierungskante_erwünscht*, ob eine Expansion durchgeführt werden soll. Die Expansion einer Spezialisierungskante hat zur Folge, daß der allgemeine Netzknoten durch den spezielleren ersetzt wird. In Abbildung 4.13 wird beim Schritt von Suchbaumknoten 11 zu Suchbaumknoten 12 die Instanz I_1(BAUM) spezialisiert zu I_1(LAUBBAUM). Dabei werden alle Kanten und Merkmale von I_1(BAUM) auf I_1(LAUBBAUM) übertragen.

Zusammenfassend kann das Zusammenspiel der ERNEST++–Basisaktionen während der Analyse eines Signals wie folgt charakterisiert werden. Sie garantieren auf der einen Seite eine wohldefinierte Abarbeitung eines Eingangssignals durch ein semantisches Netz. Andererseits lassen sie dem Modellierer genügend Freiraum für eine flexible Verarbeitungsstrategie, um den unterschiedlichen Anforderungen, die sich aus verschiedensten Problemstellungen ergeben, gerecht zu werden.

[8] Eine Kante ist vollständig expandiert, wenn der in der Modalität notierte Wert für die maximale Anzahl von Zielknoten erreicht ist.

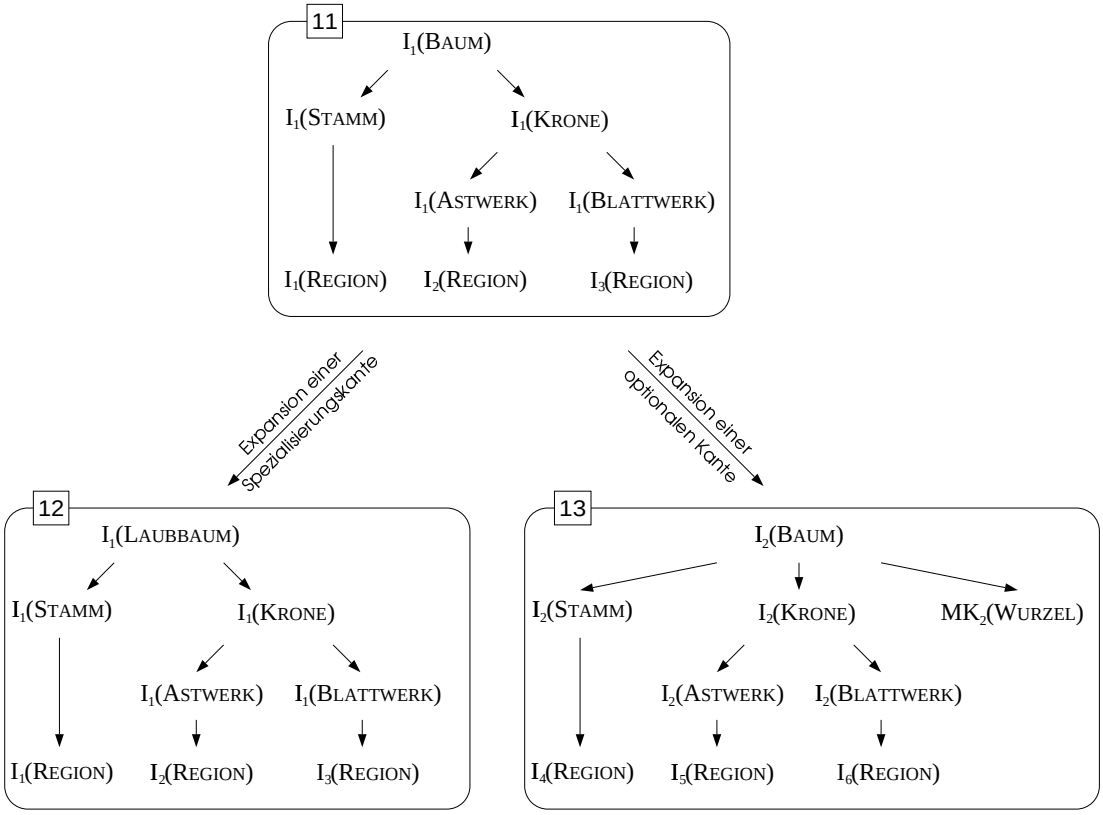

Abbildung 4.13: Expansion optionaler Kanten und Spezialisierungskanten

4.5 Aspekte der objektorientierten Implementation

Der Grundgedanke der objektorientierten Implementation von ERNEST^{++} besteht darin, jede ERNEST^{++}–Komponente auf eine eigene C++–Klasse abzubilden. Daher führt zum Beispiel jede ERNEST^{++}–Konzeptdefinition zu jeweils einer C++–Klasse. Die während der Analyse entstehenden Netzknoten sind C++–Instanzen dieser Klassen. In der Abbildung 4.14 ist ein Ausschnitt aus der Klassendefinition zu BAUM (siehe Seite 53 und 57) zu sehen, in der die Komponenten von BAUM wiederzufinden sind, die ganz spezifisch für diese Konzeptdefinition sind. Die

```
class Baum : public E_Netzknoten {
Baum_wurzel          *wurzel;       // Kante wurzel
Baum_stamm           *stamm;        // Kante stamm
Baum_krone           *krone;        // Kante krone
Baum_laubbaum        *laubbaum;     // Kante laubbaum
Baum_nadelbaum       *nadelbaum;    // Kante nadelbaum
Baum_familie         *familie;      // Merkmal familie
Baum_höhe            *höhe;         // Merkmal höhe
Baum_anordnung       *anordnung;    // Relation anordnung
...
};
```

Abbildung 4.14: C++–Klassendefinition zu BAUM

Klassendefinitionen werden automatisch aus den Konzeptdefinitionen erzeugt. Die Informationen, die nicht automatisch gewonnen werden können, müssen bei der Implementation explizit angegeben werden. Dazu gehört insbesondere die Festlegung der Werttypen der Merkmale.

Jeweils gleichartige Klassen erben durch eine übergeordnete Klasse die gemeinsamen Einträge. Beispielsweise haben alle ERNEST++–Merkmale Rollen — folglich befindet sich in der Basisklasse E_Merkmal ein Eintrag *rolle*, der in den abgeleiteten Klassen mit einem Wert belegt wird. Abbildung 4.15 zeigt diese generellen Klassen.[9]

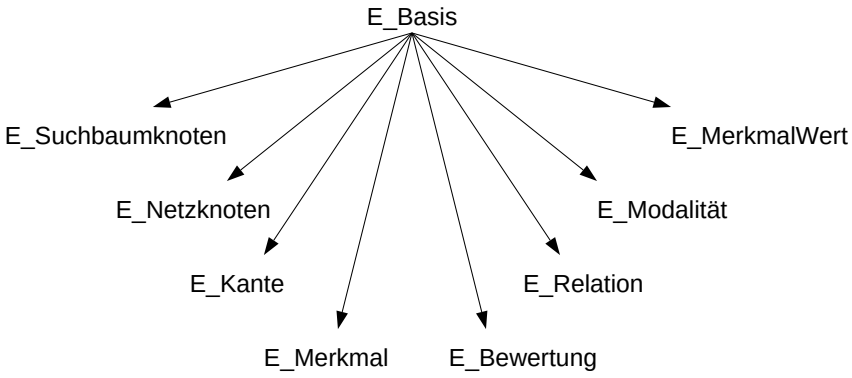

Abbildung 4.15: Generelle Klassenstruktur in ERNEST++

4.6 Zusammenfassung

ERNEST++ ist ein semantisches Netzwerksystem, das speziell zur Interpretation von Sensorsignalen entwickelt wurde und die positiven Eigenschaften von semantischen Netzwerkformalismen im allgemeinen wie Modularität, Wohlstrukturiertheit und Kompaktheit besitzt. Begriffe werden in ERNEST++ in Konzeptdefinitionen modelliert, die während der Analyse der Daten ihre Entsprechung in Netzknoten finden. Es gibt drei Typen von Netzknoten: Konzepte, modifizierte Konzepte und Instanzen, wobei letztere eine vollständige Zuordnung eines Signalauschnittes zu einer Konzeptdefinition repräsentieren. Zur Modellierung von Beziehungen von Begriffen stehen dem Modellierer die drei Kantentypen Konkretisierung, Bestandteil und Spezialisierung zur Verfügung. Begriffsinterne Eigenschaften werden in ERNEST++ durch Merkmale realisiert, die durch Relationen in Beziehung zueinander gesetzt werden können. In eigenen Bewertungsstrukturen kann die Qualität und Sicherheit der Interpretation umfassend bewertet werden. Durch die Definition fester Knoten– und Kantentypen ist ERNEST++ in der Lage, problemunabhängige Inferenzregeln bereit zu stellen, was die Beseitigung einer wesentlichen Schwäche von semantischen Netzwerkformalismen im allgemeinen bedeutet. Die Verarbeitungsstrategie basiert auf dem A*–Algorithmus. Durch die Bereitstellung verschiedener Funktionen hat der Modellierer in jedem

[9] In der Abbildung sind nur die Klassen zu den in dieser Arbeit vorgestellten ERNEST++–Komponenten dargestellt.

Suchbaumknoten die Möglichkeit, auf den Fortgang der Analyse Einfluß zu nehmen. Somit kann eine an das Problem angepaßte Verarbeitungsstrategie entwickelt werden.

Die klare, explizite Wissensrepräsentation einerseits und die flexible Verarbeitungsstrategie andererseits machen ERNEST++ zu einem attraktiven Werkzeug zur Interpretation gesprochener Sprache.

Kapitel 5

Die Realisierung der Sprachverstehenskomponente

Die Dinge sollten so einfach wie möglich gemacht werden. Aber nicht einfacher.
Albert Einstein

In diesem Kapitel stelle ich die Sprachverstehenskomponente vor, die im Rahmen dieser Arbeit für den Einsatz in einem Konstruktionsszenario konzipiert und implementiert wurde. Der grundlegende Anspruch an die Sprachverstehenskomponente ergibt sich dabei aus den Untersuchungen in Kapitel 3: den Instrukteuren sollen keinerlei Restriktionen bei der Formulierung ihrer Anweisungen gegeben werden. Im Kontext der gesamten Szene soll der sprachverarbeitende Teil des gesamten Konstruktionssystems auch dann angemessen reagieren, falls eine Äußerung nicht oder nur teilweise erkannt oder verstanden wurde. Dieser Anspruch an die Verstehenskomponente deckt sich mit den Vorstellungen, die Allen wie folgt formuliert:

> „The bottom line for a dialogue system is that it should never give up.“ [All95a]

Es geht mir in dieser Arbeit also *nicht* darum, einen Beitrag zur linguistischen Forschung in dem Sinne zu leisten, daß ich einen ganz speziellen Aspekt in Konstruktionsdialogen untersuche und ein ausgefeiltes Modell für ihn entwickle, das alle möglichen Variationen dieses Aspektes berücksichtigt. Ich sehe die Idee des hier vorgestellten Ansatzes vielmehr auf einer Linie mit dem Herangehen der flachen Analyse im Verbmobil–Projekt (siehe Abschnitt 2.1), deren Ziel die mit wenig Aufwand verbundene und daher schnelle Analyse einer Äußerung ist. In diesem Kapitel stelle ich daher ein effizientes sprachverstehendes System vor, das den Anforderungen in einem kooperativ gestalteten Dialog zwischen Mensch und Maschine adäquat begegnet.

Das Kapitel ist wie folgt aufgebaut. Zunächst beschreibe ich die Stellung der Verstehenskomponente in dem realisierten Gesamtsystem. Danach erläutere ich das Zusammenwirken von Spracherkennung und Sprachverstehen. Sodann stelle ich den Aufbau der ERNEST++-Wissensbasis vor und lege anschließend die Verarbeitungsstrategie dar. Einige besondere Aspekte des ERNEST++–Netzes sind Gegenstand des nachfolgenden Abschnitts. Es folgt ein Ausblick, bevor ein Resümee das Kapitel beendet.

5.1 Sprachverarbeitung im Konstruktionsszenario

Die Sprachverstehenskomponente ist konzipiert für ein Gesamtsystem, durch das ein Mensch einen Roboter anweisen kann, mit Baufix–Basiselementen und –Aggregaten Konstruktionsaufgaben zu erfüllen. Dieser Anspruch an das Gesamtsystem gliedert sich in drei Aufgaben. Zum einem muß die Szene visuell analysiert werden, zum zweiten muß der Inhalt der menschlichen Instruktion im Kontext des Dialogs verstanden und letztendlich muß die intendierte Handlung ausgeführt und überwacht werden. Im weiteren Verlauf des Abschnittes stelle ich das Gesamtsystem vor und spezifiziere die Aufgabenstellung für die Sprachverstehenskomponente.

5.1.1 Das Gesamtsystem im Überblick

Das Gesamtsystem besteht aus drei *Säulen*: der Bildverarbeitungssäule, der Sprachverarbeitungssäule und der Robotiksäule, deren innerer Aufbau und Beziehung zueinander in Abbildung 5.1 zu sehen ist.

Aufgabe der Bildverarbeitungssäule ist es, die Szene visuell zu explorieren. Insbesondere geht es darum, Hypothesen für die in der Szene befindlichen Baufix–Objekte zu gewinnen. Nach der Aufnahme der Szene wird dazu zunächst eine Farbklassifikation durchgeführt. Das farbklassifizierte Bild bildet die Grundlage für die Segmentierung von Regionen gleicher Farbe. Aus deren Formparametern wird bestimmt, welche Objekte sich in der Szene befinden. In dem hybriden Verfahren wird ein semantisches mit einem künstlichen neuronalen Netz kombiniert. Genaue Beschreibungen finden sich in [Hei96], [Kum98a] und [Bau98].

Die Robotiksäule ist verantwortlich für die Ausführung der Instruktionen und deren Überwachung. Dabei ist daran gedacht, daß die Handlungen sowohl von einem Manipulator [Zha98] als auch virtuell [Jun97] durchgeführt werden können. Der derzeitige Stand des Systems erlaubt allerdings nur die Konstruktion in der virtuellen Welt.

Den Aufgaben der Sprachsäule ist der folgende Abschnitt 5.1.2 gewidmet. Der Informationsaustausch der Säulen läuft über eigene Module. Das Modul *Planung* [Fri99] vermittelt zwischen der Sprach– und der Robotiksäule. Die Aufgabe dieses Moduls besteht vor allem darin, aus der

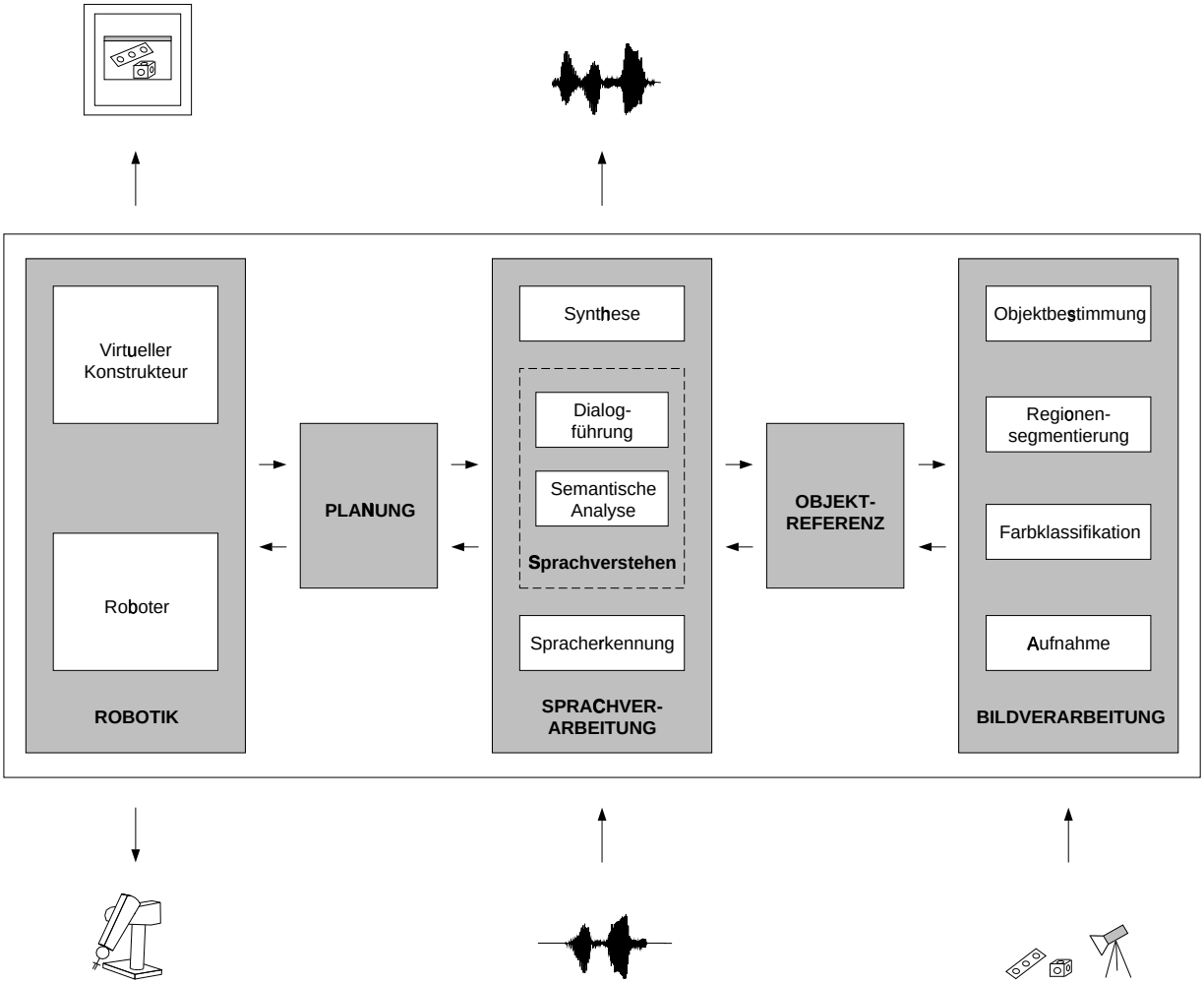

Abbildung 5.1: Das Gesamtsystem im Überblick

semantischen Repräsentation der Instruktion eine Folge von einfachen Roboteranweisungen zu generieren. So muß aus der Anweisung „Steck' den Ring auf die Schraube." eine Sequenz von Roboteranweisungen der folgenden Art erzeugt werden:

1. nimm den Ring in die linke Hand
2. nimm die Schraube in die rechte Hand
3. stecke das Objekt in der linken Hand auf das Objekt in der rechten Hand

Bei der Planung der Handlungsausführung ist insbesondere zu beachten, ob die intendierte Handlung in der jeweiligen Situation überhaupt durchführbar ist. Gegebenenfalls muß die Undurchführbarkeit der Sprachsäule mitgeteilt werden, damit eine Nachfrage oder ein Hilfegesuch an den Instrukteur gestellt werden kann. Sollte etwa im obigen Beispiel der Ring außerhalb der Reichweite des Roboters liegen, gibt dieser eine entsprechende Nachricht an das Planungsmodul, welches sie zur Sprachsäule weiterleitet, die dann den Instrukteur bitten kann einzugreifen.

Die Aufgabe des Planungsmoduls besteht zusätzlich darin, die Ausführung der Handlung zu überwachen, um einen angemessenen Fortgang der Konstruktion insbesondere in Ausnahmesituationen zu gewährleisten. Wenn etwa dem Roboter während eines Schraubvorganges die Schraube aus der Hand gleitet und er das Mißgeschick nicht selbst beheben kann, so sollte das Planungsmodul des Systems darauf reagieren und den Roboter ansteuern, um die Schraube wieder aufzunehmen oder die Sprachverstehenskomponente anweisen, den Instrukteur um Hilfe zu bitten.[1]

Der Informationsaustausch zwischen der Bildverarbeitungs– und der Sprachsäule findet über das Modul *Objektreferenz* [Wac99] statt. Hierbei geht es um das Finden von Referenten in der Szene. Dazu werden die Objektbenennungen neben der visuellen Auswertung als weitere partielle Repräsentation der gesamten Szene betrachtet. Die visuelle und sprachliche Interpretation der Szene werden in den selben Formalismus — nämlich Nachbarschaftsgraphen — überführt, um sie anschließend mit Hilfe von dynamisch aufgebauten Bayes–Netzen [Pea88] abgleichen zu können. Als Ergebnis liefert das Modul eine oder mehrere bewertete Hypothesen für die Referenten der benannten Objekte.

Zur Kommunikation zwischen den Modulen wird im Gesamtsystem das im SFB 360 entwickelte Kommunikationssystem *Distributed Applications's Communication System (DACS)* benutzt [Fin95b, Jun98]. DACS unterstützt die Entwicklung sehr sicherer und einfach zu programmierender Schnittstellen. Dabei bleibt der einzelne Entwickler davon verschont, sich mit den Problemen kommunizierender Rechner im Detail zu beschäftigen. DACS hält verschiedene Kommunikationssemantiken bereit, um den unterschiedlichen Anforderungen in einem komplexen System gerecht zu werden und hat sich auch unter großer Last bewährt.

5.1.2 Aufgaben und grundlegender Aufbau der Sprachverarbeitung im Gesamtsystem

Die Aufgabe der Sprachsäule im Gesamtsystem besteht darin, die sprachlichen Instruktionen entgegenzunehmen und zu verstehen, eine entsprechende Systemreaktion einzuleiten und eigene sprachliche Systemausgaben zu produzieren. Diese Aufgabenstellung gliedert sich in folgende inhaltlichen Punkte:

1. Die Objektbenennungen, die in den Äußerungen enthalten sind, müssen intern repräsentiert und gegebenenfalls disambiguiert werden. Eventuell sind dazu Rückfragen an den Instrukteur zu stellen. Die bei der Untersuchung zur Modellbildung (siehe Abschnitt 3.4.1)

[1] Ich wende mich dagegen, eine solche Situation als „Fehlersituation" zu bezeichnen, sondern verwende den Begriff „Ausnahmesituation". Denn diese Situationen gehören zum ganz normalen Leben — auch Menschen kann etwas aus der Hand rutschen und sie sind in der Lage, diese Situation zu meistern, ohne in einen „Fehlerstatus" zu schalten.

gewonnenen Erkenntnisse über die Formulierungen der Instrukteure müssen dabei berücksichtigt werden.

2. Es müssen Handlungsaufforderungen und ihr Zusammenhang über mehrere Äußerungen hinweg erkannt werden. Im Unterschied zu Auskunftssystemen ist dem System dabei im vorhinein nicht bekannt, welches Ziel im Dialog verfolgt wird und wie der Instrukteur dieses Konstruktionsziel erreichen möchte.

3. Dem Instrukteur muß Gelegenheit gegeben werden, die Ausführung einer Handlung in einer neuen Anweisung zu korrigieren beziehungsweise direkt mit einer Intervention zu unterbrechen.

4. In Abstimmung mit den anderen Komponenten muß eine angemessene Systemreaktion bestimmt werden. Ergibt sich aus der Anweisung unmittelbar eine Rückfrage, ist diese zu generieren. Andernfalls muß der Planungsprozeß angestoßen werden, dessen Ergebnis seinerseits wiederum eine Rückfrage erfordern kann.

Bei der Umsetzung dieser Aufgabenstellung darf nicht außer acht gelassen werden, daß die Sprachsäule die Hauptschnittstelle zum Instrukteur ist. Darum ist stets eine angemessene Gestaltung der Mensch–Maschine–Kommunikation als Ganzes besonders zu berücksichtigen. Dem Instrukteur sollte immer klar sein, was das System gerade tut beziehungsweise was es seinerseits vom Instrukteur erwartet. Das ist zweifellos eine der wichtigsten Voraussetzungen für den erfolgreichen Dialog zwischen Mensch und Maschine.

Der Aufbau der Sprachsäule ist ebenfalls in Abbildung 5.1 zu sehen. Die *Spracherkennung* erzeugt aus dem Sprachschall eine Folge von Worthypothesen. Die *semantische Analyse* extrahiert deren Bedeutung und die *Dialogführung* ist für die Herstellung des Dialogkontextes und die Auswahl der nächsten Systemreaktion zuständig. Die beiden letztgenannten Module bilden die Sprachverstehenskomponente und sind in einem ERNEST^{++}–Netz realisiert. Die *Synthese* schließlich produziert die Sprachausgaben des Systems.

Die Modulstruktur der Sprachsäule erinnert zunächst an die aus Kapitel 2 bekannte typische Struktur von sprachverarbeitenden Systemen, deren Aufbau und Strategie meist streng hierarchisch in horizontalen Ebenen organisiert ist: der Spracherkennung folgt in typischen Systemen die syntaktisch–semantische Analyse, bevor eine Dialogkomponente den Dialogkontext herstellt und eine Systemreaktion anstößt. Die Kommunikation der einzelnen Ebenen findet in solchen Systemen über wohldefinierte Schnittstellen meist unidirektional nur von den signalnäheren zu den abstrakteren Ebenen statt. Der Vorteil einer solchen Architektur liegt darin, daß jede Ebene unabhängig von den anderen modelliert und optimiert werden kann. Der große Nachteil besteht allerdings darin, daß Wissen, das in abstrakteren Ebenen enthalten ist, den signalnäheren nicht zur Verfügung steht und für die Bildung sinnvoller Restriktionen auf diesen Ebenen daher nicht

verwendet werden kann. Aus diesem Grund wird in dieser Arbeit eine *vertikale Organisation des Wissens* realisiert. Die Idee ist, daß auf allen Ebenen der Analyse auch Wissen aus anderen Ebenen benutzt wird. Das heißt beispielsweise, daß Wissen über die Konstruktionsdomäne auch schon bei der Spracherkennung explizit Verwendung findet und sie damit verbessert. Dieser vertikale Ansatz in der Sprachverarbeitung erfordert allerdings, daß die einzelnen Wissensbausteine sehr gut aufeinander abgestimmt sind.

Wie die Module der Sprachsäule im einzelnen funktionieren und kommunizieren, ist Gegenstand der folgenden Abschnitte. Im Kern der Darstellung steht dabei die Realisierung der Sprachverstehenskomponente als angekündigtes Ergebnis dieser Arbeit.

5.2 Spracherkennung und Sprachverstehen

In diesem Abschnitt geht es um das Zusammenwirken von Spracherkennung und Sprachverstehen in der Sprachsäule. Zum besseren Verständnis der Konzeption dieser Interaktion möchte ich zunächst einen kleinen Exkurs in die Sprach*erkennung* machen. Auf den nächsten beiden Seiten erläutere ich das traditionelle Herangehen an die automatische Spracherkennung und stelle einen interessanten Ansatz zur Verbesserung der Spracherkennungsergebnisse vor.

In der heutigen Spracherkennungstechnologie folgen praktisch alle Verfahren dem statistischen Verarbeitungsparadigma [Bah83]. Dabei wird die Sprachproduktion und –erkennung als ein informationstheoretischer Prozeß angesehen (siehe Abbildung 5.2). Eine Sequenz $\boldsymbol{w}$ von gespro-

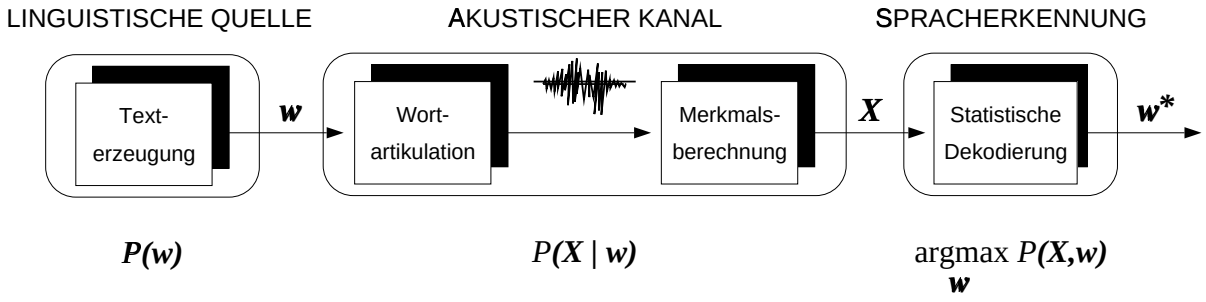

Abbildung 5.2: Das informationstheoretische Kanalmodell der Spracherzeugung und –erkennung (nach [ST95])

chenen Wörtern $w_1, \ldots, w_m$ wird in diesem Modell durch einen akustischen Kanal in eine Folge $\boldsymbol{X}$ von Merkmalsvektoren $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$ verschlüsselt. Die Aufgabe der Spracherkennung besteht in der Dekodierung von $\boldsymbol{X}$, das heißt in der Ermittlung einer Wortfolge $\boldsymbol{w}^*$, die eine möglichst gute Näherung der gesprochenen Wortfolge $\boldsymbol{w}$ sein sollte. Die Berechnungsvorschrift für $\boldsymbol{w}^*$ lautet mit Hilfe der Bayesregel:

$$\boldsymbol{w}^* = \underset{\boldsymbol{w}}{\operatorname{argmax}}\, P(\boldsymbol{w}|\boldsymbol{X}) = \underset{\boldsymbol{w}}{\operatorname{argmax}}\, \frac{P(\boldsymbol{w}) \cdot P(\boldsymbol{X}|\boldsymbol{w})}{P(\boldsymbol{X})} = \underset{\boldsymbol{w}}{\operatorname{argmax}}\, P(\boldsymbol{w}) \cdot P(\boldsymbol{X}|\boldsymbol{w}) \quad (5.1)$$

$\boldsymbol{w}^*$ ist also die Wortfolge, die mit maximaler Wahrscheinlichkeit produziert wurde, unter der Voraussetzung, daß die Folge der Merkmalsvektoren $\boldsymbol{X}$ gegeben ist. Die Kunst der Spracherkennung besteht nun darin, die benötigten Wahrscheinlichkeiten zu berechnen. Die Wahrscheinlichkeit der Folge der Merkmalsvektoren $P(\boldsymbol{X})$ kann dabei vernachlässigt werden, denn sie ist unabhängig von $\boldsymbol{w}$ und trägt folglich zur Maximierung nichts bei. Zur Gewinnung der *akustischen Wahrscheinlichkeit* $P(\boldsymbol{X}|\boldsymbol{w})$ werden in der Regel Hidden–Markov–Modelle [Hua90] benutzt. $P(\boldsymbol{w})$ gibt die linguistisch motivierte *a priori Wahrscheinlichkeit* der Wortfolge an. In den meisten Fällen wird sie faktorisiert:

$$P(\boldsymbol{w}) = P(w_1) \cdot P(w_2|w_1) \cdot \prod_{i=3}^{m} P(w_i|w_1 \ldots w_{i-1}) \tag{5.2}$$

Um der kombinatorischen Explosion bei der Bestimmung von $P(\boldsymbol{w})$ zu begegnen, muß man sich in der Praxis allerdings auf die Betrachtung nur eines kleinen Teils der Vorgänger des letzten Wortes w_m einer Wortfolge $\boldsymbol{w}$ beschränken. Häufig bezieht man sogar nur das unmittelbare Vorgängerwort mit ein und kommt so auf *Bigramm–Modelle*:

$$P(\boldsymbol{w}) = P(w_1) \cdot \prod_{i=2}^{m} P(w_i|w_{i-1}) \tag{5.3}$$

Selbst bei einer Einschränkung der Geschichte des Wortes w_m ist eine Schätzung von $P(\boldsymbol{w})$ dann sehr schwer, wenn für eine relativ inhomogene Domäne nur ein kleines Trainingskorpus vorliegt. Die knapp neun Stunden gesprochener Sprache in den beiden Wizard–of–Oz–Korpora aus Kapitel 3 beispielsweise bestehen in ihrer Summe insgesamt aus 50.114 Wörtern. Die Größe des von den Sprechern dabei benutzten Vokabulars beträgt 2.146. Von den theoretisch möglichen 4,6 Millionen Bigrammen, kommen nur 14.753 — das sind etwa 0,3 Prozent — tatsächlich vor, 65 Prozent davon sogar nur einmal. Das ist keine optimale Grundlage für die Schätzung von a priori Wahrscheinlichkeiten. Häufig hat man keine andere Wahl als die bestimmten Bigrammwahrscheinlichkeiten im Nachhinein zu glätten [Kat87] und dabei insbesondere auszuschließen, daß für eine Wortkombination $w_i w_j$ gilt: $P(w_j|w_i) = 0$ und diese somit gänzlich als unzulässig bewertet würde.

Ein anderes Verfahren bezüglich des Einbringens von linguistischem Wissen in den spracherkennenden Prozeß wird von Wachsmuth in [Wac97] vorgestellt. Während die a priori Wahrscheinlichkeiten implizit das Sprachfragment bei der Spracherkennung nutzen, wird mit dieser Arbeit die Möglichkeit geschaffen, linguistisches Wissen über das Fragment dem Spracherkenner explizit zur Verfügung zu stellen. Damit können den Schwächen von rein statistischen Ansätzen bei einer kleinen Trainingsdatenbasis gezielt begegnet und die Spracherkennungsergebnisse verbessert werden.

Die Grundidee des Verfahrens in [Wac97] besteht darin, die statistischen Sprachmodelle mit einer deklarativen LR(1)–Grammatik (siehe zum Beispiel [Aho86]) zu kombinieren und direkt in den spracherkennenden Prozeß einfließen zu lassen. Für spontan gesprochene Sprache kann

aufgrund ihrer Vielfalt allerdings keine Grammatik für ganze Sätze festgelegt werden. Daher wird ein Parser benutzt, der in der Grammatik definierte Konstituenten akzeptiert. Somit wird es möglich, die Grammatik weich einzusetzen, das heißt sie entscheidet nicht nach Abschluß einer Äußerung hart über deren Zulässigkeit oder Nichtzulässigkeit. Vielmehr beeinflußt die Grammatik während der Abarbeitung des Sprachsignals den Fortgang der Analyse. Mit einem Strafterm werden diejenigen Wortfolgen versehen, die der Grammatik nicht entsprechen. Es ergibt sich dann eine Gesamtbewertung B für die Wortfolge $\boldsymbol{w}$:

$$B(\boldsymbol{w}) = P(\boldsymbol{X}|\boldsymbol{w}) \cdot [P(\boldsymbol{w}) \cdot B_G^{fehler}(\boldsymbol{w})]^{\gamma} \tag{5.4}$$

wobei der Strafterm $B_G^{fehler}(\boldsymbol{w})$ zwischen null und eins liegt und auch berücksichtigt, in welcher Art und Weise (beispielsweise Abbruch einer begonnenen Konstituente) die Wortfolge der Grammatik G nicht entspricht. Mit dem Gewicht γ kann der Einfluß der linguistisch motivierten Bewertung von $\boldsymbol{w}$ bestimmt werden. In [Wac98] werden die mit diesem Ansatz erzielten Verbesserungen vorgestellt. Auf einer unabhängigen Teststichprobe wurde eine Verringerung der Wortfehlerrate [Lee89] um über elf Prozent erreicht, als eine Grammatik zusätzlich zum Bigramm eingesetzt wurde.

Soweit der Exkurs in die Spracherkennung.

Bei der Realisierung der Verstehenskomponente nutze ich das von Wachsmuth entwickelte Verfahren. Dazu betrachte ich es aber aus einem ganz anderen Blickwinkel, nämlich vom automatischen Sprachverstehen her. Es ergeben sich dann zwei interessante Aspekte:

1. Typischerweise besteht eine der ersten Aufgaben der Sprachverstehenskomponente darin, eine syntaktische Analyse vorzunehmen. Dazu bedarf es einer Syntaxmodellierung. Wenn man aber diese Modellierung in den oben vorgestellten Formalismus übersetzt, so kann die syntaktische Analyse in wesentlichen Teilen bereits während der Sprach*erkennung* erledigt werden. Die vom Parser während der Abarbeitung eines Sprachsignals aufgebauten Strukturen können nämlich vom Spracherkenner mit ausgegeben werden und stehen also der Verstehenskomponente zur Verfügung.

2. Die Definition der LR(1)–Grammatik kann natürlich unter verschiedenen Aspekten geschehen. Der naheliegendste Ansatz ist wohl der, die bekannten syntaktischen Konstituenten wie beispielsweise eine Nominalphrase oder eine Verbalphrase zu modellieren. Man hat aber auch die Möglichkeit, direkt pragmatisches und domänenspezifisches Wissen in die Grammatik zu importieren, indem man entsprechende Nichtterminale definiert. Durch diese vertikale Organisation der Wissensbasis kann schon die Spracherkennung die Besonderheiten von Konstruktionsanweisungen berücksichtigen.

Abbildung 5.3 zeigt beispielhaft, wie diese Aspekte umgesetzt werden. Man sieht einen Ausschnitt aus der LR(1)–Grammatik, die der Spracherkenner benutzt. Die von der Grammatik

```
1  $$SEGMENT:                    $INT_OBJEKT |
                                 $AKTION |
                                 $REF_OBJEKT |
                                 $OBJEKT_SPEZIFIKATION |
                                 $HILFSOBJEKT |
                                 $INTERVENTION |
                                 ... ;
2  $INT_OBJEKT:                  $$objekt_nom |
                                 $$objekt_gen |
                                 $$objekt_dat |
                                 $$objekt_akk ;
3  $$objekt_nom:                 $$det_e $$adj_e_liste |
                                 $$det_e $$adj_e_liste $$objektnomen_nom_sing_fem |
                                 $$det_er $$adj_e_liste |
                                 $$det_er_$$adj_e_liste $$objektnomen_nom_sing_mas |
                                 ... ;
4  $$adj_e_liste:                $$adj_e |
                                 $$adj_e_list $$adj_e ;
5  $$adj_e:                      $$farb_adj_e |
                                 $$form_adj_e |
                                 $$gross_adj_e |
                                 $$lokal_adj_e ;
6  $$farb_adj_e:                 blaue | dunkle | rote | ... ;
7  $$objektnomen_nom_sing_fem:   Buchse | Holzlatte | Scheibe | ... ;
8  $$det_e:                      die | diese | eine | ... ;
```

Abbildung 5.3: Auszug aus der Segmentdefinition

akzeptierten Wortfolgen nenne ich von nun an *Segmente*, die Grammatik in ihrer Gesamtheit *domänenspezifische Segmentdefinition*.[2] Die Terminalsymbole der Segmentdefinition, die alle dem Vokabular des Systems entstammen, sind fett gedruckt. Die Segmente sind alle in Großbuchstaben angegeben und beginnen mit einem einzelnen $–Zeichen. Die Substrukturen der Segmente, die *Subsegmente*, sind in kleinen Buchstaben angegeben und beginnen mit $$.

Das Symbol $$SEGMENT ist das Startsymbol der Segmentdefinition. Es kann ersetzt werden durch die Segmente, von denen die wichtigsten in Abbildung 5.3 angegeben sind.[3] Sie entsprechen den im Abschnitt 3.4 entwickelten Spezifikationen für die Modelle. $INT_OBJEKT steht für einfache Objektbenennungen wie „diese rote Scheibe“, $AKTION dient zur Ableitung von Verben, welche zu einer Konstruktionshandlung auffordern („nimm“). $REF_OBJEKT bezeichnet ein Referenzobjekt zur näheren Bestimmung eines Objektes („neben dem grünen Klotz“) und $OBJEKT_SPEZIFIKATION steht für eine nähere Beschreibung eines Objektes („mit dem

[2] Ich vermeide in diesem Zusammenhang den Begriff „Konstituente“ und „Konstituentengrammatik“, weil es Segmente gibt, die im linguistischen Sinne keine syntaktischen Konstituenten bilden und andersherum manche Konstituenten kein unmittelbares Gegenstück in der Segmentdefinition haben.

[3] Im Anhang A.1 können alle verwendeten Segmente nachgelesen werden.

eckigen Kopf“). Ein $HILFSOBJEKT ist ein Hilsfmittel zur Ausführung einer Handlung („mit der Schraube“). Eine $INTERVENTION deckt Äußerungsteile ab, in denen der Roboter aufgefordert wird, die Ausführung einer Instruktion zu unterbrechen („halt stop!“).

Anhand von $INT_OBJEKT wird die Idee der Segmentdefinition näher erläutert. In Regel 2 ist formuliert, daß jede Objektbenennung in einem bestimmten Kasus auftritt. Objektbenennungen im Nominativ werden in Regel 3 näher beschrieben. Unter Berücksichtigung der Deklinationsregeln für Adjektive nach [Dro84] besteht eine Objektbenennung im Nominativ zum Beispiel aus einem femininen Artikel ($$det_e), gefolgt von einer Adjektivliste, deren Elemente alle auf ’e’ enden ($$adj_e_liste), und einem Nomen, welches im Singular steht und den Genus Femininum sowie den Kasus Nominativ besitzt ($$objektnomen_nom_sing_fem). In der Regel 7 ist allerdings näher spezifiziert, daß für eine Benennung eines Objektes nicht jedes Nomen des Vokabulars in Frage kommt. Vielmehr werden als Terminale nur solche Nomen zugelassen, die auch tatsächlich zur Benennung der Objekte benutzt werden. Dieses Wissen wurde ja aus der Korpusbetrachtung (siehe Seite 32ff) gewonnen. In der Definition der Grammatik für den Spracherkenner ist auf diese Weise das pragmatische und Domänenwissen repräsentiert — eine vertikale Organisation des Wissens für den gesamten Prozeß der automatischen Sprachverarbeitung ist realisiert.

Durch die Segmentdefinition wird nicht nur gemäß den Gleichungen 5.1 bis 5.4 die Suche der besten Wortfolge beeinflußt, sondern auch die Ausgabe des Spracherkenners. Wie bereits erwähnt bleiben die während der Analyse aufgebauten Strukturen erhalten. Das Ergebnis des Spracherkenners ist also strukturiert. Die Anweisung

> „Jetzt nimm doch bitte die grüne Schraube mit dem eckigen Kopf neben ähm neben der langen Leiste.“ (5.5)

führt zum Beispiel bei einer erfolgreichen Erkennung zu folgender Ausgabe des Spracherkenners:

> jetzt (nimm: $AKTION) doch bitte (die grüne Schraube: $INT_OBJEKT) (mit dem eckigen Kopf: $OBJEKT_SPEZIFIKATION) neben ähm (neben der langen Leiste: $REF_OBJEKT) (5.6)

Die Segmente sind also als Einheiten markiert, und jene Teile der Äußerung, die in der Segmentdefinition nicht modelliert sind, werden ohne Angabe eines Segmentes ausgegeben. Dieses Beispiel belegt auch, daß nicht genau ein Segment immer genau eine syntaktische Konstituente abdeckt. So wird die Nominalphrase „die grüne Schraube mit dem eckigen Kopf“ durch zwei Segmente ($INT_OBJEKT und $OBJEKT_SPEZIFIKATION) erfaßt. Der Grund für diese Modellierung liegt in folgendem: erkennt der Spracherkenner aufgrund schlechter Signalqualität oder undeutlicher Aussprache ein Wort innerhalb eines definierten Segmentes nicht, so kann der Parser das Segment als Ganzes nicht detektieren und gibt die erkannten Wörter unmarkiert ein-

zeln aus. Je mehr Substrukturen ein Segment enthält, umso größer ist die Wahrscheinlichkeit, daß eine davon nicht erkannt wird und das ganze Segment zerbricht. Die Modellierung zu langer Segmente hat daher im Ergebnis oft das Gegenteil des Beabsichtigten zur Folge: statt domänenspezifisch motivierter Einheiten liefert der Spracherkenner nur unstrukturierte Einzelergebnisse. Aus diesem Grund finden in der Segmentdefinition sowohl die Interessen der Spracherkennung als auch die Ansprüche der Sprachverstehenkomponente Berücksichtigung.

Nachdem nunmehr dargelegt ist, wie Aspekte des Sprachverstehens direkt in den spracherkennenden Prozeß einfließen und somit bereits strukturierte Ergebnisse vom Spracherkenner produziert werden, geht es in den folgenden beiden Abschnitten darum, wie diese Ergebnisse weiterverarbeitet werden. Ich stelle dazu zunächst die verwendete ERNEST–Wissensbasis vor.

5.3 ERNEST–Wissensbasis der sprachverstehenden Komponente

Die ERNEST–Wissensbasis zum Sprachverstehen ist sehr flach hierarchisch aufgebaut und gliedert sich in drei Abstraktionsebenen[4], die in Abbildung 5.4 zu sehen sind. Die signalnächste

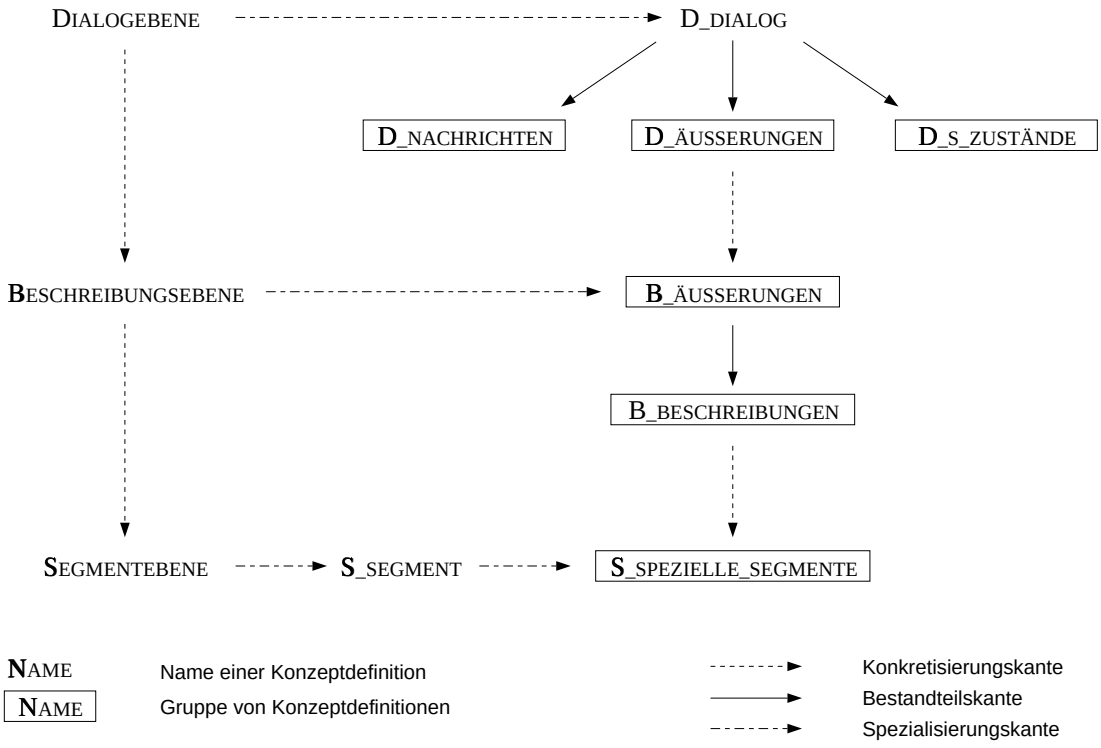

Abbildung 5.4: Die ERNEST–Wissensbasis zum Sprachverstehen im Überblick

[4] Anhang A.2 enthält alle Konzeptdefinitonen im Überblick.

Ebene ist die *Segmentebene*. Die Konzeptdefinition S_SEGMENT repräsentiert ein vom Spracherkenner erhaltenes Segment. Die Spezialisierungen von S_SEGMENT[5] stehen für die speziellen Segmente. Sie finden ihre Entsprechung auf der *Beschreibungsebene* in den Konzeptdefinitionen B_BESCHREIBUNGEN, welche eine semantische Interpretation der einzelnen Segmente bestimmen. Die einzelnen Beschreibungen werden in den Konzeptdefinitionen B_ÄUSSERUNGEN zu einer Beschreibung der ganzen Äußerung zusammengefaßt. Auf der *Dialogebene* befindet sich die Konzeptdefinition D_DIALOG zur Modellierung des Dialogs als Ganzes. Seine Bestandteile sind Instrukteursäußerungen, repräsentiert durch D_ÄUSSERUNGEN, mit verschiedenen Ausgaben verbundene Systemzustände, modelliert in D_S_ZUSTÄNDE, sowie Nachrichten der anderen Säulen, die in D_NACHRICHTEN zusammengefaßt sind.

In den folgenden Abschnitten werden die einzelnen Ebenen näher erläutert.

5.3.1 Die Segmentebene als Schnittstelle zum Spracherkenner

Auf der Segmentebene des Netzes wird die Schnittstelle zum Spracherkenner realisiert. Alle in der Segmentdefinition enthaltenen Segmente sind auf der Segmentebene des semantischen Netzes durch je eine Konzeptdefinition modelliert (siehe Abbildung 5.5). Alle diese Konzeptde-

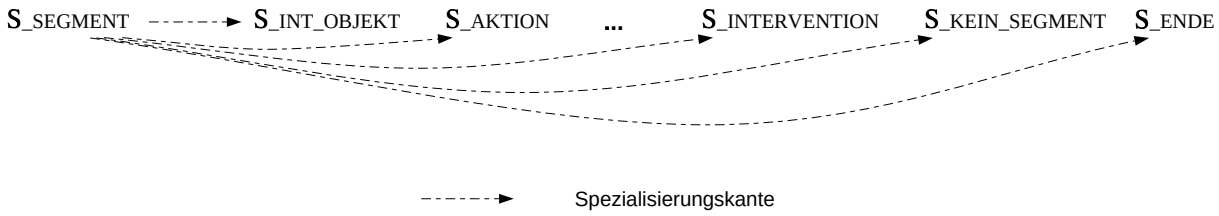

Abbildung 5.5: Die Segmentebene der Wissensbasis

finitionen sind Spezialisierungen von S_SEGMENT, das für die Segmente im allgemeinen steht. In S_SEGMENT ist das Merkmal *hypothese* eingetragen, welches die Ergebnisse des Spracherkenners inkrementell einliest. In S_KEIN_SEGMENT sind diejenigen Äußerungsteile repräsentiert, die keinem Segment zugeordnet werden können. Dazu zählen Äußerungsteile, die von der Segmentdefinition nicht abgedeckt sind und solche, die der Spracherkenner nicht als vollständiges Segment erkannt hat. Wenn also während des spracherkennenden Prozesses ein Segment zerbrochen ist, kann bei Bedarf die in den einzelnen Wörten enthaltene Information für den Interpretationsprozeß noch genutzt werden. Das Ende einer Äußerung ist durch die Konzeptdefinition S_ENDE modelliert. Eine spezielle Modellierung für das Äußerungsende ist unerläßlich, weil die meisten Systemreaktionen erst nach dem Abschluß einer Äußerung sinnvoll sind.

[5] In Abbildung 5.4 und den nachfolgenden Abbildungen sind zur besseren Übersicht teilweise mehrere Konzeptdefinitionen zusammengefaßt und durch eine Einrahmung kenntlich gemacht.

Die Konzeption der Segmente als besondere Einheit im automatischen Sprachverstehen ist vergleichbar mit anderen Ansätzen des partiellen Parsings [Jac91, Kaw96, Kaw98]. Ihnen gemeinsam ist die Idee, daß sich ein großer Teil der notwendigen Informationen in sogenannten Schlüsselphrasen (*key–phrases*) verbirgt. Für solche Phrasen lassen sich im Gegensatz zu ganzen Sätzen oder gar Äußerungen zuverlässige Modelle entwickeln. Das partielle Parsing ist daher besonders geeignet, die Robustheit des sprachverstehenden Systems zu vergrößern. Denn bei Satzgrammatiken besteht stets die Gefahr, daß das Parsen einer Äußerung wegen eines möglicherweise sehr kleinen Erkennungsfehlers komplett fehlschlägt. Die genannten Ansätze bleiben aber bei der Detektion der Schlüsselphrasen stehen und betten sie nicht in ein Dialogsystem ein. Daher ist in diesen Vorschlägen auch nicht beschrieben, wie Äußerungsteile verarbeitet werden, die keiner Schlüsselphrase entsprechen.

Das im Terminabsprachesystem (siehe Abschnitt 2.3) erläuterte und mit Hilfe von semantischen Hidden–Markov Netzwerken realisierte Sprachverstehen ist der Konzeption der Verstehenskomponente ebenfalls verwandt. Genau wie die Segmente stehen die SHMNs für eine vertikale Modellierung des Wissens und liefern strukturierte Ergebnisse. Allerdings muß das linguistische Wissen komplett in der ERNEST–Wissensbasis aufwendig modelliert sein, bevor die SHMNs daraus automatisch abgeleitet werden können. Die Modellierung in einer deklarativen Segmentdefiniton ist dagegen deutlich schlanker und daher wesentlich einfacher erweiterbar und wartungsfreundlicher. Durch die Verwendung von SHMNs ist es möglich, vom Spracherkenner direkt Instanzen zu Konzeptdefinitionen der Semantik– oder sogar Pragmatikebene[6] zu bekommen. Die darunterliegenden Ebenen, insbesondere die eigentliche Schnittstellenebene, die Hypothesenebene, verlieren damit ihren Sinn. Insofern stellt die hybride Modellierung mit Segmentdefinitionen eine konsequente Weiterentwicklung der Idee der SHMNs dar: das Wissen über die Struktur semantischer beziehungsweise pragmatischer Einheiten ist direkt in der Segmentdefinition enthalten, und die Segmente bilden eine sauber modellierte Schnittstelle zwischen Spracherkennung und Sprachverstehen.

5.3.2 Beschreibungsebene

Aufgabe der Beschreibungsebene ist es, aus den Segmenten eine interne Repräsentation der Äußerungssemantik zu extrahieren. Die grundlegende Modellierungsidee besteht darin, jeder Konzeptdefiniton der Segmentebene eine entsprechende Konzeptdefinition auf der Beschreibungsebene zuzuordnen, welche die Bedeutung des Segmentes ermittelt. Abbildung 5.6 gibt einen Überblick.[7]

[6] Gemeint sind die Ebenen, wie sie in Abbildung 2.6 auf der Seite 17 zu sehen sind.

[7] In Abbildung 5.6 (und auch den folgenden Abbildungen, die ERNEST++–Netz–Auschnitte zeigen) sind alle nicht mit einer Dimension versehenen Kanten eindimensional, das heißt sie haben maximal einen Zielnetzknoten.

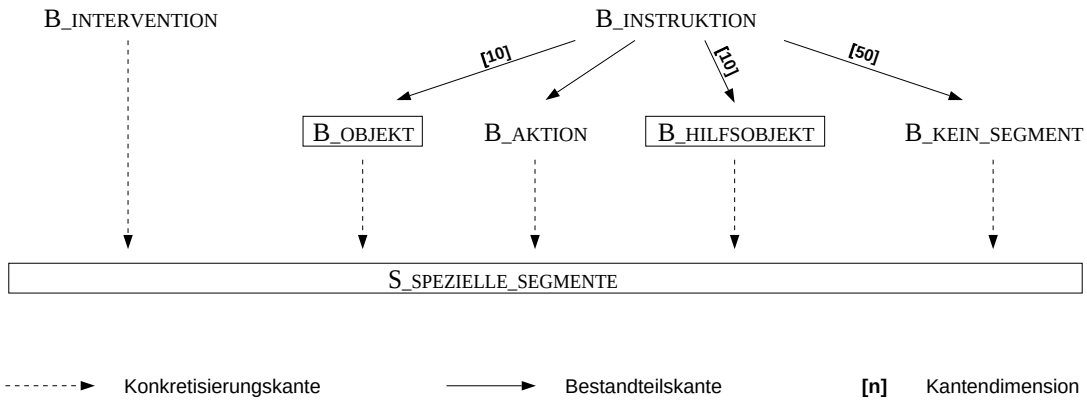

Abbildung 5.6: Überblick über die Beschreibungsebene und ihre Anbindung an die Segmentebene

Es sind zunächst die direkten Pendants zu den Segmenten zu sehen:

- B_AKTION repräsentiert ein handlungsanweisendes Verb. Diese Konzeptdefinition besitzt keine Dekomposition, sondern ist so einfach strukturiert, daß eine unmittelbare Interpretation des entsprechenden Segmentes vorgenommen werden kann. Beispielsweise kann aus einer Instanz von S_AKTION, die den Äußerungsteil „nimm“ enthält, sofort die Bedeutung gewonnen werden, daß eine Aufforderung zum Nehmen eines Objektes vorliegt. Insgesamt sind entsprechend der Modellspezifikation in der Sprachverstehenskomponente derzeit folgende Aktionen als mögliche Roboterhandlungen definiert: befestigen, schrauben, stecken, entfernen, nehmen, positionieren und beenden. Alle direkten Instruktionen werden auf diese Aktionen abgebildet. Zusätzlich können auch einfache Benennungen („Das nenne ich jetzt Propeller.“) verarbeitet werden.

- B_KEIN_SEGMENT ist den Äußerungsteilen gewidmet, denen kein Segment zugeordnet werden kann.

- Auch die Modellierung von kurzen Interventionen erfolgt innerhalb eines einzigen Segmentes, auf das B_INTERVENTION direkt zugreifen und eine Interpretation vornehmen kann. Unter Interventionen verstehe ich ausschließlich kurze Äußerungen, die darauf gerichtet sind, die Ausführung der Handlung durch den Roboter sofort zu unterbrechen.[8]

Die Modellierung von Basiselement– und Aggregatbenennungen in B_OBJEKT und Hilfsobjekten zur Ausführung einer Handlung durch B_HILFSOBJEKT muß dagegen aufwendiger sein, weil sie aus mehreren Segmenten bestehen.

[8] Der Interventionsbegriff ist hier also sehr eng gefaßt. In anderen Arbeiten, beispielsweise in [Pet98, Lob98], werden alle kurzen Anweisungen, die unmittelbar in die Ausführung einer gerade stattfindenden Handlung eingreifen sollen, als Interventionen betrachtet.

Die Modellierung für die Objektbenennungen — also die detaillierte Darstellung von B_OBJEKT — ist in Abbildung 5.7 dargelegt. Eine umfassende Objektbeschreibung (B_OBJEKTBESCHREIBUNG) kann in diesem Modell aus einem benannten intendierten Ob-

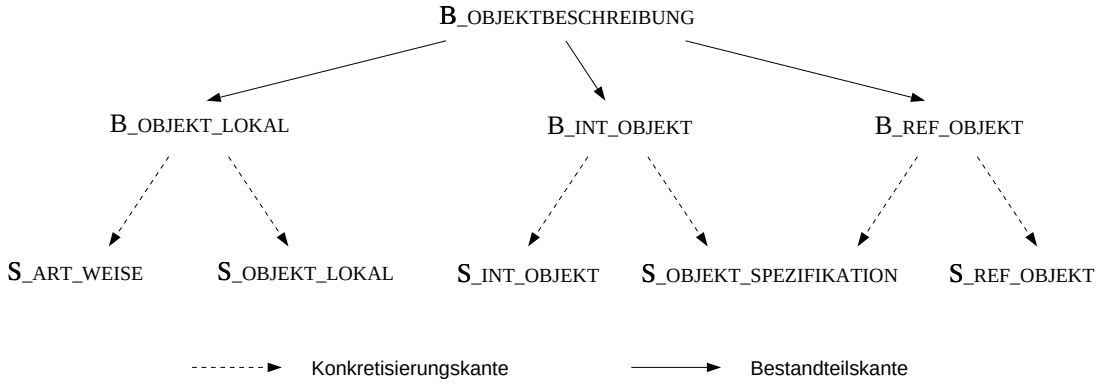

Abbildung 5.7: Modellierung für Objektbenennungen mit objektinterner Lokalisierung

jekt (B_INT_OBJEKT) und einem Referenzobjekt (B_REF_OBJEKT) bestehen. Beide können jeweils durch eine S_OBJEKT_SPEZIFIKATION näher beschrieben sein. Ihre Konkretisierungen S_INT_OBJEKT und S_REF_OBJEKT stehen für einfach strukturierte Benennungen wie „die rote runde Schraube“ beziehungsweise „neben dem weißen Reifen“, die in genau einem Segment erfaßt werden. Zusätzlich kann eine Objektbeschreibung auch die Benennung einer objektinternen Lokalisierung (B_OBJEKT_LOKAL) beinhalten.[9] Die objektinternen Lokalisationen umfassen sowohl die Angabe eines Ortes („das dritte Loch von links“) als auch die Art und Weise, wie gegebenenfalls ein objektinterner Ort zu erreichen ist („von unten“). Die Informationen über die objektinternen Lokalisationen werden vom Merkmal B_OBJEKT_LOKAL.*beschreibung* berechnet und in der Klasse *Objekt_Lokalisation*[10] eingetragen (siehe Abbildung 5.8). Die in der Instruktion „Steck’ die Schraube von unten durch das dritte Loch von links der Leiste.“ enthal-

```
class Objekt_Lokalisation : public E_MerkmalWert {
EnumTyp                 typ;            // {Loch | Schaft | ...}
EnumArtWeise            art_weise;      // {unten | oben | links | ...}
EnumInternPosition      position;       // {links | rechts | mittig | ...}
int                     anzahl;         // {1, 2, 3, ... 7}
};
```

Abbildung 5.8: Klasse zur Repräsentation objektinterner Lokalisationen

tene Benennung eines objektinternen Ortes führt zu folgender Belegung der Einträge: *typ* erhält den Wert *Loch* und *art_weise* bekommt den Wert *unten*. Dem Eintrag *position* wird der Wert

[9] Das ist auch der Grund, warum ich den Konzeptnamen B_OBJEKTBESCHREIBUNG und nicht B_OBJEKTBENENNUNG verwende — die Angabe einer objektinternen Lokalisierung geht ja über die reine Benennunge eines Objektes zum Zwecke seiner Referenzierung hinaus.

[10] Die Gestaltung dieser Klasse entstand auf Anregung des C1–Projektes im SFB 360.

links und *anzahl* die Zahl drei zugewiesen, denn drei Einheiten von links her gezählt beschreibt die objektinterne Lage des intendierten Loches.

Die Hilfsobjekte zur Ausführung einer Handlung sind im Baufix–Szenario selbst wieder Basiselemente. Daher sind sie im wesentlichen wie die Objektbeschreibungen modelliert. Die Modellierung von B_HILFSOBJEKT unterscheidet sich nur in zwei Punkten von der Modellierung der Objektbeschreibungen: zum einen wurde auf die Integration von B_OBJEKT_LOKAL und den damit verbundenen Konzeptdefinitionen verzichtet. In den Wizard–of–Oz–Korpora findet sich kein einziges Beispiel, in dem ein Instrukteur eine Hilfsobjektbenennung mit einer internen Lokalisation angereichert hätte. Zum zweiten wird nicht auf S_INT_OBJEKT konkretisiert sondern auf das S_HILFSOBJEKT, das explizit einfache Hilfsobjektbenennungen wie „mit der langen Schraube“ repräsentiert.

Die Berechnung der semantischen Repräsentationen in den Merkmalen der Konzeptdefinitionen geschieht im wesentlichen durch den Zugriff auf ein Lexikon, welches unter anderem Grundformen und Synonyme[11] bereitstellt. Alle Wörter werden dadurch auf ein Vokabular abgebildet, welches das Modul Objektreferenz zur Referentenbestimmung verwendet. In B_OBJEKTBESCHREIBUNG werden die einzelnen Repräsentationen zu einer umfassenden Objektbeschreibung zusammengeführt. Die entsprechende Klasse *Objekt_Beschreibung* ist in Abbildung 5.9 zu sehen. In dieser Klasse werden die zur eindeutigen Diskriminierung benötig-

```
class Objekt_Beschreibung : public E_MerkmalWert {
string                    typ;          // benannter Typ
string                    *farbe;       // benannte Farbe
string                    *form;        // benannte Form
string                    *größe;       // benannte Größe
string                    *lokal;       // benannter Ort in der Szene
Ref_Objekt_Beschreibung   **ref_objekt; // benannte Referenzobjekte
Objekt_Lokalisation       **intern_lokal; // objektinterne Lokalisationen
...
};
```

Abbildung 5.9: Klasse zur Repräsentation von Objektbeschreibungen

ten Eigenschaften von Objekten, also ihr Typ sowie die Farbe, Form und Größe, festgehalten. Motiviert vom Aufbau des Gesamtsystems wurde der Eintrag *lokal* zur Verarbeitung weiterer räumlicher Angaben, wie sie beispielsweise in „die hintere Leiste“ enthalten sind, hinzugenommen. Benannte Referenzobjekte werden in *ref_objekt* eingetragen. Die dafür bereitgestellte Klasse *Ref_Objekt_Beschreibung* enthält einen Eintrag vom Typ *Objekt_Beschreibung*, der das Referenzobjekt repräsentiert und außerdem eine Liste von Präpositionen, welche die räumliche Relation zu dem intendierten Objekt beschreiben. Objektinterne Lokalisationen werden in *intern_lokal* notiert.

[11] Zum Beispiel werden die Wörter „lila“, „lilane“, „lilanen“ und weitere Flektionsformen auf das Synonym „violett“ abgebildet.

```
class Instruktion_Beschreibung : public E_MerkmalWert {
string                  aktion;     // benannte Aktion
Objekt_Beschreibung     **objekt;   // beteiligte(s) Objekt(e)
};
```

Abbildung 5.10: Klasse zur Repräsentation einer Instruktion

Die Beschreibungen der einzelnen Äußerungsteile werden schließlich in B_INSTRUKTION zur Interpretation der ganzen Anweisung zusammengefaßt. Durch die Dimensionen der Bestandteilskanten (siehe Abbildung 5.6 auf Seite 84) wird festgelegt, wieviele der einzelnen Bestandteile höchstens vorkommen dürfen. Eine Instruktion besteht also in diesem Modell im wesentlichen aus bis zu einer Aktionsnennung, zehn Objektbeschreibungen und Hilfsobjektbeschreibungen und bis zu fünfzig Wörtern, die keinem Segment zugeordnet werden konnten. Alle Bestandteilskanten sind außerdem in einer Modalität von B_INSTRUKTION als optional markiert. Damit können auch unvollständige Anweisungen verarbeitet werden: eine Instanz von B_INSTRUKTION kann auch dann entstehen, falls etwa die Spracherkennung in der Äußerung kein Aktionssegment ausfindig machen konnte. Das ist die Grundlage für eine qualifizierte Rückfrage an den Instrukteur und ein wesentliches Merkmal für eine angemessene Gestaltung der Mensch–Maschine–Kommunikation. Das Merkmal B_INSTRUKTION.*beschreibung* trägt die Interpretation der Anweisung zusammen und benutzt dazu die Klasse *Instruktion_Beschreibung* (Abbildung 5.10). Eine Beschreibung einer Anweisung wird darin einfach aufgefaßt als eine auszuführende Aktion und die Beschreibung der beteiligten Objekte.

Abschließen möchte ich die Darstellung der Beschreibungsebene mit einem zusammenfassenden Beispiel. Abbildung 5.11 zeigt die interne Repräsentation der Äußerung 5.5 von Seite 80. Sie

```
Instruktion_Beschreibung {
aktion:    nehmen;
objekt:    typ:          Schraube;
           farbe:        grün;
           form:         eckig;
           ref_objekt:   relation:   neben;
                         objekt:     typ:      Loch_Leiste;
                                     form:     lang;
};
```

Abbildung 5.11: Interne Repräsentation einer Anweisung

ist als Analyseergebnis der Beschreibungsebene in einer Instanz von B_INSTRUKTION enthalten. Abbildung 5.12 zeigt die vollständigen Instanzenbaum, der bei Zuordnung der bisher vorgestellten Konzeptdefinitionen zu der Beispieläußerung entsteht. Auf der untersten Ebene stehen die Instanzen der Segmentebene, denen jeweils ein Teil des Sprachsignals zugeordnet ist. Auf der Beschreibungebene werden zunächst I_1(S_INT_OBJEKT) und I_1(S_OBJEKT_SPEZIFIKATION)

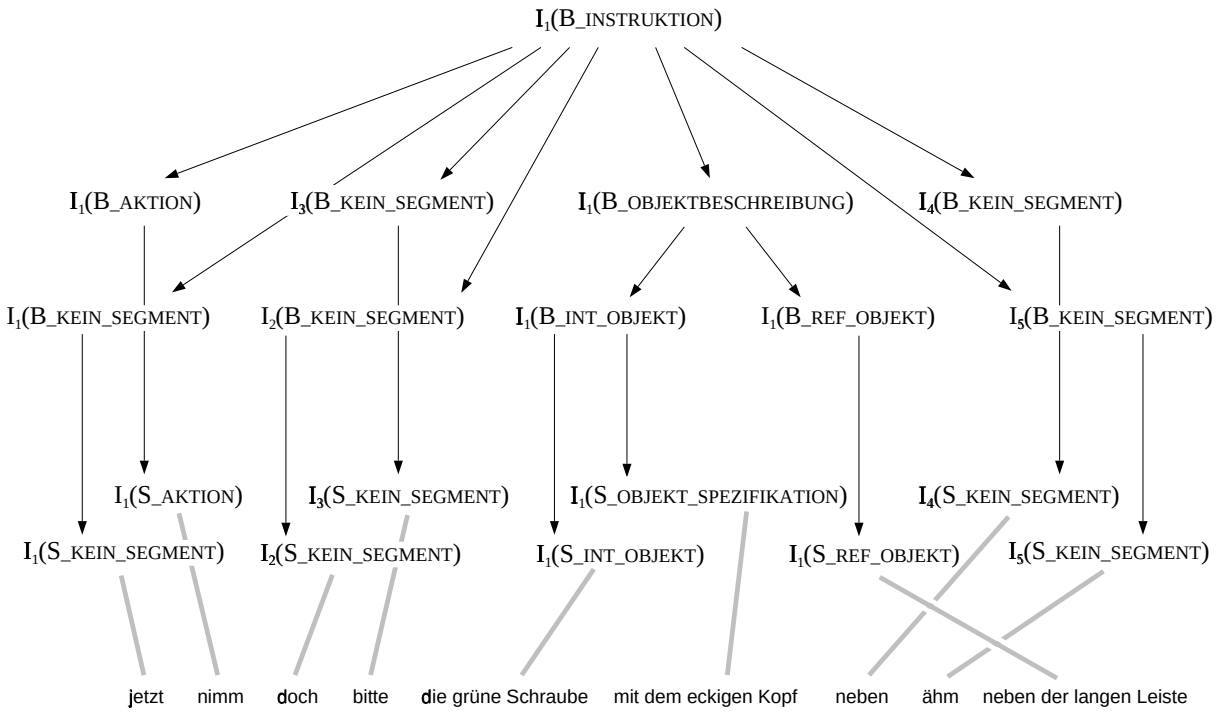

Abbildung 5.12: Instanzenbaum für eine Anweisung

durch I_1(B_INT_OBJEKT) sowie I_1(S_REF_OBJEKT) durch I_1(B_REF_OBJEKT) interpretiert und dann in I_1(B_OBJEKTBESCHREIBUNG) zu einer Beschreibung des benannten Objektes zusammengefaßt. Gemeinsam mit I_1(B_AKTION), welche die Interpretation der Handlungsaufforderung enthält, und den fünf Instanzen von B_KEIN_SEGMENT bildet sie die Grundlage für die Instanz I_1(B_INSTRUKTION), welche die Interpretation der gesamten Anweisung enthält. In diesem Beispiel genügt die Analyse von I_1(B_AKTION) und I_1(B_OBJEKTBESCHREIBUNG), um eine in sich geschlossene Interpretation aus dem Sprachsignal zu extrahieren. Daher brauchen die in den Instanzen I_j(B_KEIN_SEGMENT) enthaltenen Informationen nicht weiter ausgewertet zu werden. In I_1(B_INSTRUKTION) ist schließlich das in Abbildung 5.11 dargelegte Interpretationergebnis enthalten. Dieses Ergebnis bildet den Ausgangspunkt für die Interpretation der Anweisung im Kontext des Dialogs und der aktuellen Szene.

5.3.3 Dialogebene

Auf der Dialogebene werden die einzelnen sprachlichen Instruktionen mit dem Dialogkontext verbunden. Außerderm werden Informationen, welche die anderen Säulen bereitstellen, ausgenutzt und man erhält somit eine adäquate Auswertung der aktuellen Konstruktionsszene aus der

Sicht der Dialogführung.[12] Auf dieser Ebene geht es also auch um die Integration mehrerer Informationskanäle, damit eine geeignete Systemreaktion bestimmt werden kann. Aus diesem Grund reicht es nicht aus, ein reines Dialogmodell zu entwickeln, in dem ausschließlich die Abfolge der sprachlichen Kommunikation des Instrukteurs mit dem System konzipiert ist. Im folgenden stelle ich daher zunächst ein Ablaufmodell für die Mensch–Maschine–Kommunikation vor, das auch die Ergebnisse der Bildverarbeitungs– und der Robotiksäule berücksichtigt. Danach lege ich die Realisierung dieses Modells in der ERNEST++–Wissensbasis dar.

Ablaufmodell

Die Idee des Modells besteht darin, Systemzustände zu definieren, die mit festgelegten Aktionen des Systems, beispielsweise mit einer erklärenden Sprachausgabe oder dem Anstoßen des Planungsprozesses, verbunden sind. Im Modell in Abbildung 5.13 sind die möglichen Zustände

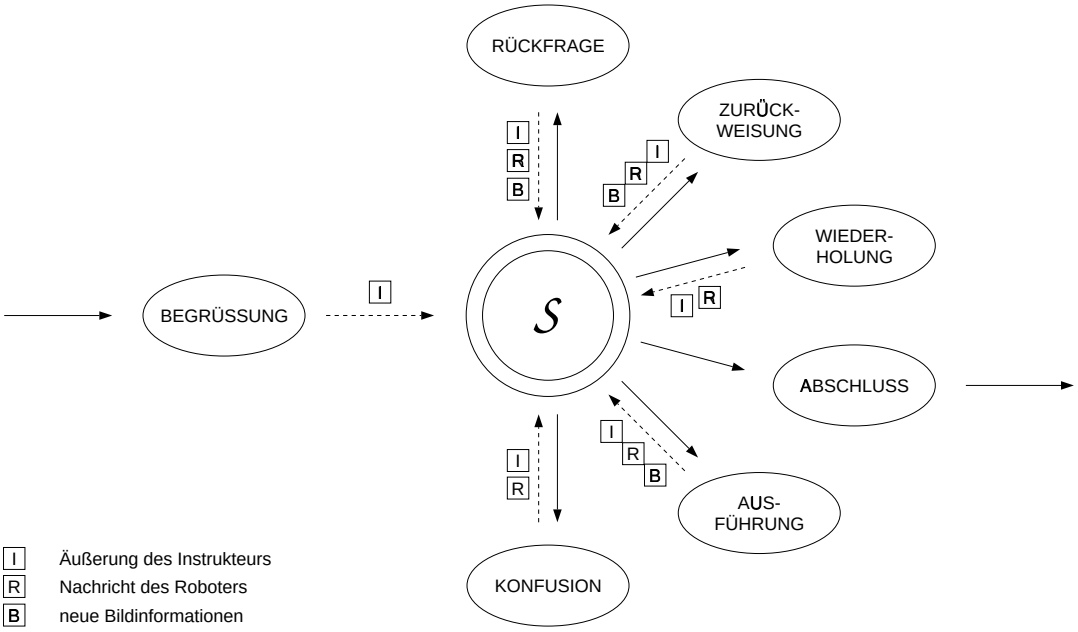

Abbildung 5.13: Ablaufmodell für die Mensch–Maschine–Kommunikation in dem Gesamtsystem

der Verstehenskomponente in Ellipsen dargestellt. Der doppelt umrandete Knoten $\mathcal{S}$ in der Mitte[13] realisiert die Auswahl des nächsten Zustandes. Das Verlassen eines Zustandes — in Abbildung 5.13 durch gestrichelte Pfeile angedeutet — wird ausgelöst durch eine Äußerung des Instrukteurs [I], eine interne Nachricht des Roboters an die Sprachverstehenskomponente [R] oder

[12] Diese Szenenauswertung geschieht also unter einem ganz bestimmten und eingeschränkten Blickwinkel. Andere Module des Gesamtsystems benötigen eine ganz andere Szenenauswertung, in der beispielweise die exakte Lage der Objekte und ihre geometrischen Daten berechnet werden.

[13] Der Buchstabe $\mathcal{S}$ steht für **S**zeneauswertung.

neue aus dem Bild gewonnene Informationen [B]. Übergänge zwischen den Systemzuständen sind nur über den Knoten $\mathcal{S}$ möglich, der dem Wesen nach die Übergangsfunktion

$$\mathcal{S}\colon \mathrm{Z} \times \mathrm{I} \times \mathrm{R} \times \mathrm{B} \longrightarrow \mathrm{Z}$$

darstellt, wobei Z die Zustände, I die Instrukteursäußerungen, R die möglichen Robotermeldungen und B neue Bildinformationen repräsentieren. Die Pfade durch das Modell ergeben somit die möglichen Abläufe der Mensch–Maschine–Kommunikation. Im weiteren wird gezeigt werden, daß es fast keine Restriktionen im Ablauf gibt: mit Ausnahme der Zustände *BEGRÜSSUNG* und *ABSCHLUSS* kann jeder Zustand auf jeden folgen. Darin spiegelt sich auch die Erkenntnis aus Kapitel 3 wider, daß kein aussagekräftiges Ablaufmodell für die Dialoge im Wizard–of–Oz–Korpus–I gefunden werden konnte.

Jede Konstruktion beginnt mit einer Begrüßung durch das System. Im Systemzustand *RÜCKFRAGE* wird eine klärende Rückfrage (beispielsweise „Meinst Du die rote oder die blaue Schraube?“) an den Instrukteur gestellt. Im Normalfall wird man erwarten, daß der Instrukteur die Rückfrage beantwortet. Aber auch neue Informationen der Bildverarbeitung (beispielsweise durch die Analyse einer Zeigegeste des Instrukteurs) führen zum Verlassen dieses Zustandes, weil eine erneute Szeneauswertung dann sinnvoll ist und die Rückfrage sich gegebenenfalls erledigt hat. Genausogut kann eine Robotermeldung ein Verlassen des Zustandes bewirken. Wenn beispielsweise der Roboter mitteilt, daß er ein Teil aus der Hand verloren hat, sollte zunächst diese Situation vom System gemeistert werden. Dazu muß es den aktuellen Zustand verlassen und im Knoten $\mathcal{S}$ die angemessene Reaktion, das heißt den geeigneten Folgezustand bestimmen. Diese Ausnahmesituation kann in jedem Systemzustand auftreten und führt daher stets zum Verlassen des aktuellen Zustandes. Im Zustand *WIEDERHOLUNG* wird um eine Wiederholung der letzten Anweisung gebeten, und im Zustand *ZURÜCKWEISUNG* wird die Ausführung der Instruktion durch das System zurückgewiesen. Der Zustand *AUSFÜHRUNG* steht für das Senden einer Anweisung an das Planungsmodul und deren Ausführung durch den Roboter. Die Notbremse des Systems ist im Zustand *KONFUSION* modelliert. Dem Instrukteur wird mitgeteilt, daß das System im Moment nicht weiß, was zu tun ist. Es nimmt eine angemessene Ausgangsstellung ein und bittet den Instrukteur um eine neue Anweisung ausgehend von dieser neuen Situation. Der Systemzustand *ABSCHLUSS* beendet das gemeinsame Konstruieren von Mensch und Maschine.

Im folgenden stelle ich vor, wie dieses Modell in den ERNEST–Formalismus abgebildet wurde. Dabei erläutere ich die einzelnen Zustände noch näher und beantworte außerdem die Frage, wann welcher Zustand aus welchem Grunde eingenommen wird.

Konzeptdefinitionen auf der Dialogebene

Auf der Dialogebene des ERNEST–Netzes (siehe Abbildung 5.14) finden sich die Konzeptdefinitionen zur Realisierung des Ablaufmodells. Die Grundidee besteht darin, die einzelnen Sy-

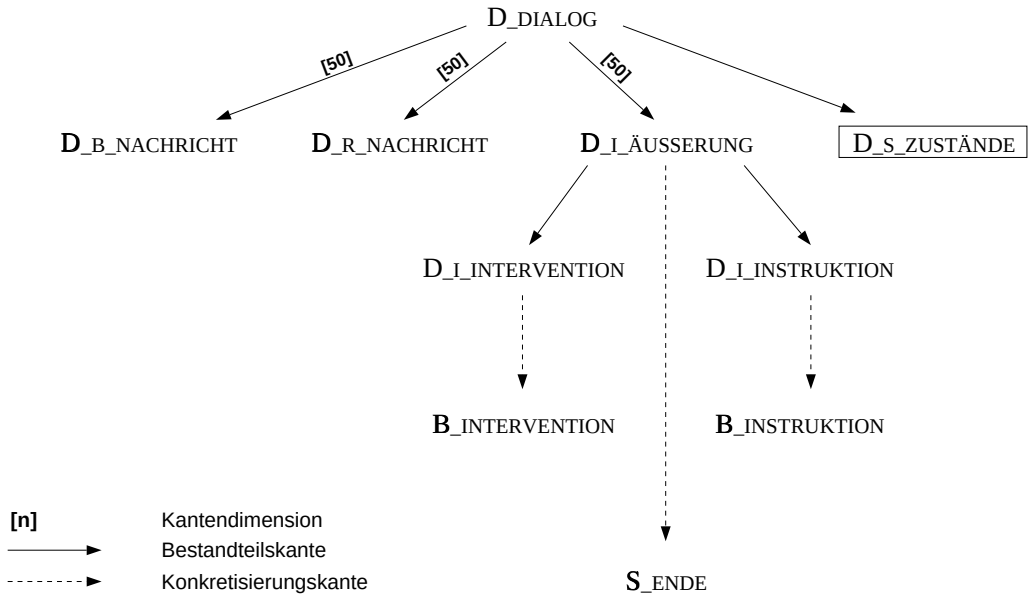

Abbildung 5.14: Die Dialogebene und ihre Anbindung an die anderen Ebenen des ERNEST–Netzes

stemzustände in den Konzeptdefinitionen D_S_ZUSTÄNDE zu modellieren, während die Übergangsfunktion $\mathcal{S}$ in der Konzeptdefinition D_DIALOG als Merkmal *szeneauswertung* realisiert ist. Die Ereignisse, die einen Zustandsübergang auslösen können, sind ebenfalls in eigenen Konzeptdefinitionen modelliert: für eine Instrukteursäußerung steht D_I_ÄUSSERUNG, eine interne Robotermeldung ist in D_R_NACHRICHT modelliert und neue Bildinformationen werden durch D_B_NACHRICHT repräsentiert. Die beiden zuletzt genannten Konzeptdefinitionen sind noch nicht ausführlich modelliert. Die Notwendigkeit dazu besteht erst, wenn ein Manipulator beziehungsweise ein gesteninterpretierendes Modul in das Gesamtsystem integriert sind. Erst dann hat man auch die Möglichkeit, die eigene Konzeption konkret auszugestalten und umzusetzen. Aus diesem Grund konzentriere ich mich im weiteren auf die Darstellung von D_DIALOG und D_I_ÄUSSERUNG sowie den Konzeptdefinitionen D_S_ZUSTÄNDE.

Die Konzeptdefinitionen D_I_INTERVENTION und D_I_INSTRUKTION nehmen die auf der Beschreibungsebene berechneten semantischen Repräsentationen entgegen. Der aktuelle Stand des Systems erlaubt nur genau eine Instruktion oder Intervention pro Äußerung. Daher entspricht ihre jeweilige Interpretation der Interpretation der ganzen Äußerung in D_I_ÄUSSERUNG, die mit Hilfe der Attributwertklasse *Instruktion_Beschreibung* (Abbildung 5.10 auf Seite 87) repräsentiert wird.

Die Konzeptdefinitionen zur Modellierung der Systemzustände sind in Abbildung 5.15 zu sehen. Bei der Instantiierung von D_S_BEGRÜSSUNG wird der Instrukteur begrüßt und um eine erste Instruktion gebeten.

D_S_RÜCKFRAGE ist für die klärenden Rückfragen an den Instrukteur zuständig. Die Rückfra-

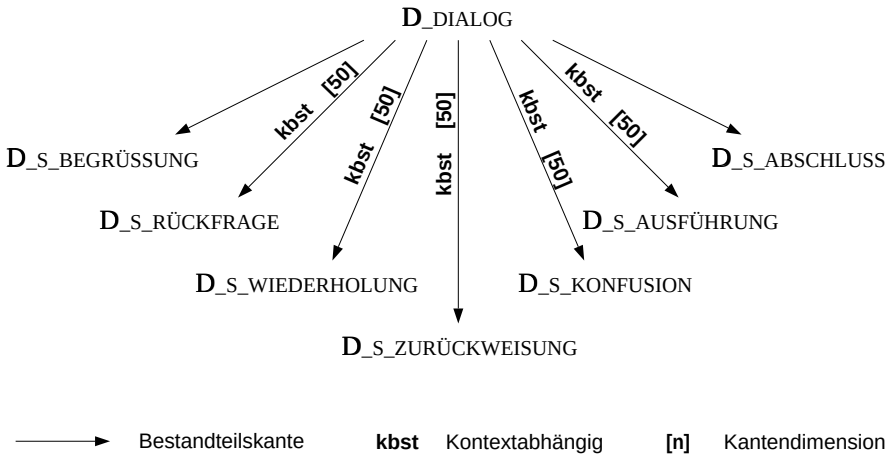

Abbildung 5.15: Die Repräsentation der Systemzustände auf der Dialogebene des ERNEST++–Netzes

gen können sich entweder auf die benannten Objekte, oder auf die mit ihnen auszuführende Aktion beziehen. In Algorithmus 5.1 ist die Generierung einer angemessenen Rückfrage im Merkmal D_S_RÜCKFRAGE.*GeneriereRückfrage* darstellt. Zunächst werden die aktuelle Aktion und das Fokusobjekt bestimmt. Das Fokusobjekt ist dabei das Objekt, über das gerade gesprochen wird. Falls die aktuelle Interpretation, in der der Dialogkontext bereits berücksichtigt ist, keine Aktionsangabe enthält, wird der Instrukteur um eine Handlungsanweisung gebeten. Andernfalls wird zunächst geprüft, ob die Argumente der Instruktion erfüllt sind, also ob die Handlungsanweisung genügend Objektbenennungen enthält. Sodann wird getestet, ob zu dem Fokusobjekt

◇ Funktion: berGeneriereRückfrage
▷ Parameter: Aktuelle Interpretation *I* mit Berücksichtigung des Dialogkontextes

```
aktuelle_aktion := I.hole_aktuelle_Aktion();
fokus_objekt := I.hole_Fokus_Objekt();
if (aktuelle_aktion == undefiniert) then
    generiere_Rückfrage_Aktion(fokus_objekt);
elseif (I.Argumente_erfüllt() == FALSE) then
    generiere_Rückfrage_zuwenig_Objekte_benannt(aktuelle_aktion);
elseif (fokus_objekt→hole_Anzahl_Referenten() == 0) then
    generiere_Rückfrage_kein_Referent(aktuelle_aktion);
elseif (fokus_objekt→hole_Anzahl_Referenten() > 1) then
    generiere_Rückfrage_welcher_Referent(fokus_objekt);
```

Algorithmus 5.1: Generierung einer angemessenen Rückfrage des Systems

kein Referent oder mehr als ein Referent im Bild gefunden wurde.[14] Für jeden dieser Fälle gibt es eine entsprechende Rückfrage des Systems, die mit Hilfe von Textschablonen erstellt wird.

Zu einer angemessenen Gestaltung der Mensch–Maschine–Kommunikation gehört auch die Fähigkeit des Systems, mit anscheinend ausweglosen Situationen umzugehen. Insbesondere die Konzeptdefinitionen D_S_WIEDERHOLUNG und D_S_KONFUSION sind diesem Anliegen gewidmet. Es kann vielfältige Gründe geben, warum eine Instruktion vom System nicht verstanden wurde. Beispielsweise könnte das Ergebnis des Spracherkenners so fehlerbehaftet sein, daß keine Informationen daraus gezogen werden können. Genausogut kann es passieren, daß der Instrukteur die Domäne verläßt[15] und folglich keine Instruktion vorliegt oder Probleme bei der Sprachaufnahme entstehen — die korrekte Bedienung eines Nahbesprechungsmikrofons ist nicht für jeden Instrukteur eine Selbstverständlichkeit. In diesen Fällen wird der Instrukteur zunächst um eine Wiederholung seiner Äußerung gebeten. Falls auch die erneute Anweisung keine neuen Informationen bringt, wird D_S_KONFUSION aktiviert. Das System bringt sich dann in einen wohldefinierten Zustand, das heißt der Roboter fährt in eine Ausgangsstellung, die Bildverarbeitung exploriert die Szene erneut und die Verstehenskomponente bittet um eine neue Anweisung. Die mit D_S_KONFUSION verbundenen Aktionen des Systems können auch in anderen Situationen sinnvoll sein, beispielsweise dann, wenn die verschiedenen Informationskanäle völlig unterschiedliche Angaben über die aktuelle Konstruktionsszene machen und eine Integration daher nicht möglich ist. Die Idee von D_S_KONFUSION besteht also darin, den einleitenden Anspruch „the system should never give up“ immer dann umzusetzen, wenn keine andere Systemreaktion mehr möglich ist.

In D_S_ZURÜCKWEISUNG wird die Instruktion zurückgewiesen — sie wurde zwar vollständig verstanden, kann aber nicht ausgeführt werden. Die Zurückweisung ist mit einer näheren Erläuterung verbunden, um dem Instrukteur eine Lösung des Problems zu erleichtern. Die Nichtausführung kann drei Ursachen haben:

1. Es liegt eine unsinnige Anweisung vor wie „Steck’ die Leiste in den Würfel.“. Die Instruktion ist im Baufix–Szenario niemals ausführbar.

2. Die Instruktion läßt sich zwar im Prinzip durchführen, die gegenwärtige Konstruktionssituation läßt aber eine Ausführung nicht zu. Wird etwa angewiesen „Schrauben Sie die orange Mutter auf die kurze gelbe Schraube.“, auf dieser befindet sich aber schon eine Mitnehmerbuchse, so kann die Instruktion nicht von statten gehen, weil das Gewinde der Schraube vollständig belegt ist.

[14] Die Klasse *Objekt_Beschreibung* besitzt außer den in Abbildung 5.9 gezeigten Einträgen noch einen weiteren, in dem die Hypothesen für die Referenten festgehalten werden.

[15] Im Wizard–of–Oz–Korpus–I finden sich beispielsweise etliche Bemerkungen über technische Details, die den Versuchsaufbau betreffen.

3. Der Roboter ist nicht in der Lage, die Aktion auszuführen, oder die Ausführung ist mißlungen: Eine Anweisung wie „Nimm den blauen Würfel.“ kann dann nicht ausgeführt werden, wenn der blaue Würfel außerhalb der Reichweite des Roboterarms liegt.

Aus welchem Grund die Anweisung nicht ausgeführt werden konnte, wird von dem Planungsmodul ermittelt. Auf der Grundlage seiner Ergebnisse kann wiederum eine detaillierte Systemausgabe dem Instrukteur weiterhelfen.

In der Konzeptdefinition D_S_AUSFÜHRUNG wird dem Instrukteur die Ausführung der Aktion mitgeteilt und der Roboter über das Planungsmodul entsprechend angestoßen. D_S_ABSCHLUSS schließlich beendet den Konstruktionsprozeß, falls der Instrukteur sich entsprechend geäußert hat.

Die zentrale Konzeptdefinition auf der Dialogebene ist D_DIALOG. Hier laufen die Informationen zusammen und die nächsten Systemaktionen werden bestimmt. Abbildung 5.16 zeigt die Konzeptdefinition. Zunächst möchte ich auf die dort eingetragenen Kanten näher eingehen. Die hochdimensionalen Kanten bringen zum Ausdruck, daß in einem Dialog beispielsweise bis zu 50 Rückfragen vorkommen können, denn bis zu 50 Netzknoten von D_S_RÜCKFRAGE können an einen Netzknoten von D_DIALOG gebunden werden.[16] Diejenigen Konzeptdefinitionen, in denen eine auf die Situation bezogene Systemausgabe produziert wird, sind jeweils als kontextabhängiges Bestandteil von D_DIALOG gekennzeichnet. Daher hat das jeweilige Merkmal, welches die Ausgabe erstellt, Zugriff auf D_DIALOG.*szeneauswertung*. Dieses wird genutzt, um die jeweils verwendeten Textschablonen mit hilfreichen Informationen aufzufüllen. So braucht bei einer Zurückweisung nicht ganz allgemein synthetisiert werden „Ich kann das angegebene Teil nicht greifen.“, sondern die Information, welches Teil gerade im Fokus ist, wird genutzt, um zum Beispiel „Ich kann den blauen Schraubwürfel nicht greifen.“ auszugeben. In der einzigen Modalität von D_DIALOG ist festgehalten, daß fast alle Kanten optional sind. Die einzige Ausnahme bildet die obligatorische Kante *begrüßung*, die außerdem mit einem Vorrang versehen ist. Das hat zur Folge, daß sie präferiert expandiert wird, wie im folgenden Abschnitt 5.4 noch näher erläutert wird.

Das Merkmal D_DIALOG.*szeneauswertung* stellt die jeweilige Instruktion in den Zusammenhang des Dialogs. Wie bereits erwähnt bleiben derzeit allerdings die Argumente *roboter.meldung* und *bild.neue_information* dabei ungenutzt. Als Ergebnis speichert das Merkmal eine während des Dialogs wachsende Liste von Instruktionsbeschreibungen, wobei jeweils das erste Listenelement die Repräsentation der aktuell intendierten Handlungsanweisung beinhaltet. Somit ist ein sehr einfaches Gedächtnis über den Verlauf des Dialogs realisiert. Das Szeneauswertung vollzieht sich in zwei Schritten. Zunächst wird die aktuelle Äußerung mit dem Kopfelement der Liste der Instruktionsbeschreibungen verbunden. Dadurch werden der Dialogzusammenhang herge-

[16] Natürlich ist die Zahl fünfzig relativ willkürlich gewählt. Theoretisch kann sie beliebig groß gewählt werden.

D_DIALOG

Bestandteil:
Rolle: **begrüßung**
Zielknoten: D_S_BEGRÜSSUNG
Bestandteil:
Rolle: **roboter**
Zielknoten: D_R_NACHRICHT
Dimension: 50
Bestandteil:
Rolle: **bild**
Zielknoten: D_B_NACHRICHT
Dimension: 50
Bestandteil:
Rolle: **äußerung**
Zielknoten: D_I_ÄUSSERUNG
Dimension: 50
kontextabhängiges Bestandteil:
Rolle: **wiederholung**
Zielknoten: D_S_WIEDERHOLUNG
Dimension: 50

kontextabhängiges Bestandteil:
Rolle: **zurückweisung**
Zielknoten: D_S_ZURÜCKWEISUNG
Dimension: 50
kontextabhängiges Bestandteil:
Rolle: **konfusion**
Zielknoten: D_S_KONFUSION
Dimension: 50
Bestandteil:
Rolle: **abschluß**
Zielknoten: D_S_ABSCHLUSS
kontextabhängiges Bestandteil:
Rolle: **ausführung**
Zielknoten: D_S_AUSFÜHRUNG
Dimension: 50
kontextabhängiges Bestandteil:
Rolle: **rückfrage**
Zielknoten: D_S_RÜCKFRAGE
Dimension: 50

Modalität:
Rolle: **modell**

Kante: **begrüßung**		Kante: **roboter**		...	Kante: **rückfrage**	
min:	1	min:	0		min:	0
max:	1	max:	50		max:	50
inhärent:	false	inhärent:	false		inhärent:	false
vorrang:	42	vorrang:	0		vorrang:	0

Merkmal:
Rolle: **szeneauswertung**
Werttyp: Instruktion_Beschreibung_Liste
Berechnung: *berSzeneauswertung* (**äußerung.interpretation, roboter.meldung, bild.neue_information**)

Merkmal:
Rolle: **ablaufsteuerung**
Werttyp: (Kantenrolle; Objekt_Beschreibung)
Berechnung: *berAblaufsteuerung* (**szeneauswertung,äußerung.interpretation**)

Abbildung 5.16: Konzeptdefinition D_DIALOG (Ausschnitt)

stellt und Informationen integriert, die über mehrere Äußerungen verteilt gegeben werden. Das Verfahren ist in Algorithmus 5.2 dargestellt. Unter Berücksichtigung der letzten Systemausgabe und des Fokusobjektes werden die benannten Objekteigenschaften und die Aktionsbenennung

◇ Funktion: Instruktion_Beschreibung_Liste::verbinde_Instruktionen
▷ Parameter: Interpretation *I* der aktuellen Äußerung
▷ Parameter: Letzte Systemausgabe *S*
▷ Parameter: aktuelles Fokusobjekt *fokus_obj*
◁ Ergebnis: Liste von Instruktionsbeschreibungen im Dialog

```
dialog_kontext := hole_Kopfelement();
if (Verbindung_sinnvoll(dialog_kontext, I, S, fokus_obj)) then
    trage_ein(dialog_kontext, I, S, fokus_obj);
else
    trage_als_neues_Kopfelement_ein(I);
```

Algorithmus 5.2: Das Verbinden einer aktuellen Äußerung mit dem Dialogkontext

in das Kopfelement der Liste eingetragen, sofern dies sinnvoll ist.[17] Andernfalls wird die aktuelle Äußerung zum Kopfelement der Liste. Nachdem sich nunmehr im Kopfelement das Wissen über die Handlungsanweisung befindet, wird im zweiten Schritt beim Modul Objektreferenz nach möglichen Referenten angefragt. Diese werden ebenfalls in das Kopfelement eingetragen.

Im Merkmal D_DIALOG.*ablaufsteuerung* wird die nächste Systemaktion auf Grundlage der Szenenauswertung bestimmt (siehe Algorithmus 5.3). Als Ergebnis wird die Rolle der Kante, die zum folgenden Zustand führt, sowie das Fokusobjekt geliefert. Durch die Markierung des Fokusobjektes kann im Folgezustand bei Bedarf konkret auf das Objekt Bezug genommen werden. Das Verfahren beginnt mit der Überprüfung, ob die aktuelle Instrukteursäußerung überhaupt Informationen enthält — bei den oben erwähnten Problemen kann es dazu kommen, daß es eine leere Instanz von D_I_ÄUSSERUNG gibt. Liegt eine solche leere Instanz vor, wird im Normalfall der Wiederholungszustand eingenommen. Allerdings wird der Konfusionszustand gewählt, falls bereits der vorangehende Zustand eine Wiederholung war. Eine Rückfrage wird dann ausgewählt, wenn dem System die aktuelle Aktion unklar ist. Ist dagegen in der Szeneauswertung die Aktion „beenden“ enthalten, wird der Systemzustand *ABSCHLUSS* als Folgezustand bestimmt. Eine Rückfrage wird auch dann gestellt, wenn die auszuführende Instruktion als nicht sinnvoll bewertet wird. Dazu wird in der Methode *aktuelle_Instruktion_sinnvoll* der Klasse *Instruktion_Beschreibung_Liste* folgendes überprüft:

1. Sind die Argumente der Handlungsanweisung erfüllt?
   Ist in der aktuellen Instruktionsbeschreibung etwa „schrauben“ als Aktion eingetragen, gleichzeitg aber nur eine Objektbeschreibung enthalten, kann die Instruktion noch nicht

[17] Sinnvoll ist ein Verbinden immer dann, wenn sich keine widersprechenden Einträge in dem Kopfelement und der aktuellen Äußerung befinden.

◇ Funktion: berAblaufsteuerung
▷ Parameter: Aktuelle Szeneauswertung *S*
▷ Parameter: Aktuelle Äußerung *akt_äußerung*
◁ Ergebnis: Rolle *rolle* der Kante, die zum nächsten Zustand führt
◁ Ergebnis: Objektbeschreibung *fokus_obj* des Fokusobjektes

```
fokus_obj := NULL;
letzter_zustand := hole_letzten_Systemzustand();
if (enthält_keine_Information(akt_äußerung)) then
    if (letzter_zustand == wiederholung) then
        rolle := konfusion;
    else
        rolle := wiederholung;
elseif (S.hole_aktuelle_Aktion() == undefiniert) then
    rolle := rückfrage;
elseif (S.hole_aktuelle_Aktion() == Beenden) then
    rolle := abschluß;
elseif (S.aktuelle_Instruktion_sinnvoll(fokus_obj) == FALSE) then
    rolle := rückfrage;
elseif (S.aktuelle_Instruktion_ausführbar(fokus_obj) == FALSE) then
    rolle := zurückweisung;
else
    rolle := ausführung;
```

Algorithmus 5.3: Ablaufsteuerung im Dialog

ausgeführt werden. Daher wird in einer Rückfrage ein weiteres Objekt erfragt. Allerdings wird die Funktionalität der Objekte nicht berücksichtigt. Die Anweisung „Steck' die Leiste in den Würfel." wird also an dieser Stelle akzeptiert. Die Begründung dafür liegt darin, daß alles Konstruktionswissen in dem Planungsmodul, welches diese Anweisung als nicht durchführbar erkennt, konzentriert liegen soll. Somit kann die Planungsaufgabe gut gekapselt und in sich geschlossen gelöst werden.

2. Ist die Anzahl der Referenten sinnvoll?
   Das ist zum einen nicht der Fall, wenn in einer Objektbeschreibung der auszuführenden Instruktion kein Referent eingetragen ist. Dann wird dieses Objekt als Fokusobjekt markiert und eine spezielle Rückfrage kann gestellt werden. Eine zu große Anzahl von Referenten ist ebenfalls problematisch, weil offenbar eine Ambiguität vorliegt, die ebenfalls

zunächst in einer Rückfrage geklärt werden sollte. Um den Dialog möglichst natürlich zu gestalten, wird an dieser Stelle allerdings (entgegen dem prinzipiellen Anspruch) etwas Konstruktionswissen genutzt: da die Würfel alle die gleiche Funktionalität haben, werden an dieser Stelle farbliche Mehrdeutigkeiten zugelassen, wenn der Instrukteur diese nicht explizit ausschließt. Die Entscheidung, welcher Referent schließlich in der Ausführung der Handlung benutzt wird, bleibt dem Roboter selbst überlassen. Schrauben und Leisten dagegen brauchen genau einen oder baugleiche Referenten, damit die Instruktion als sinnvoll angesehen wird, denn ob eine kurze oder lange Leiste verwendet werden soll, liegt im Ermessen des Instrukteurs, nicht des konstruierenden Systems.

Eine Instruktion wird zurückgewiesen, falls das Planungsmodul entscheidet, daß eine Ausführung nicht möglich ist. Wenn alle bisherigen Prämissen im Algorithmus nicht zutreffen, geht das System von der Ausführbarkeit der Instruktion aus und entscheidet sich daher für die Kante *ausführung*.

Abschließend möchte ich an einem Beispiel veranschaulichen, wie die beiden Merkmale zur Szeneauswerung und Ablaufsteuerung zusammenwirken und den Dialog steuern. Dazu ist in Abbildung 5.17 ein vereinfachter Dialogauschnitt grau unterlegt abgebildet. Die erste Instruk-

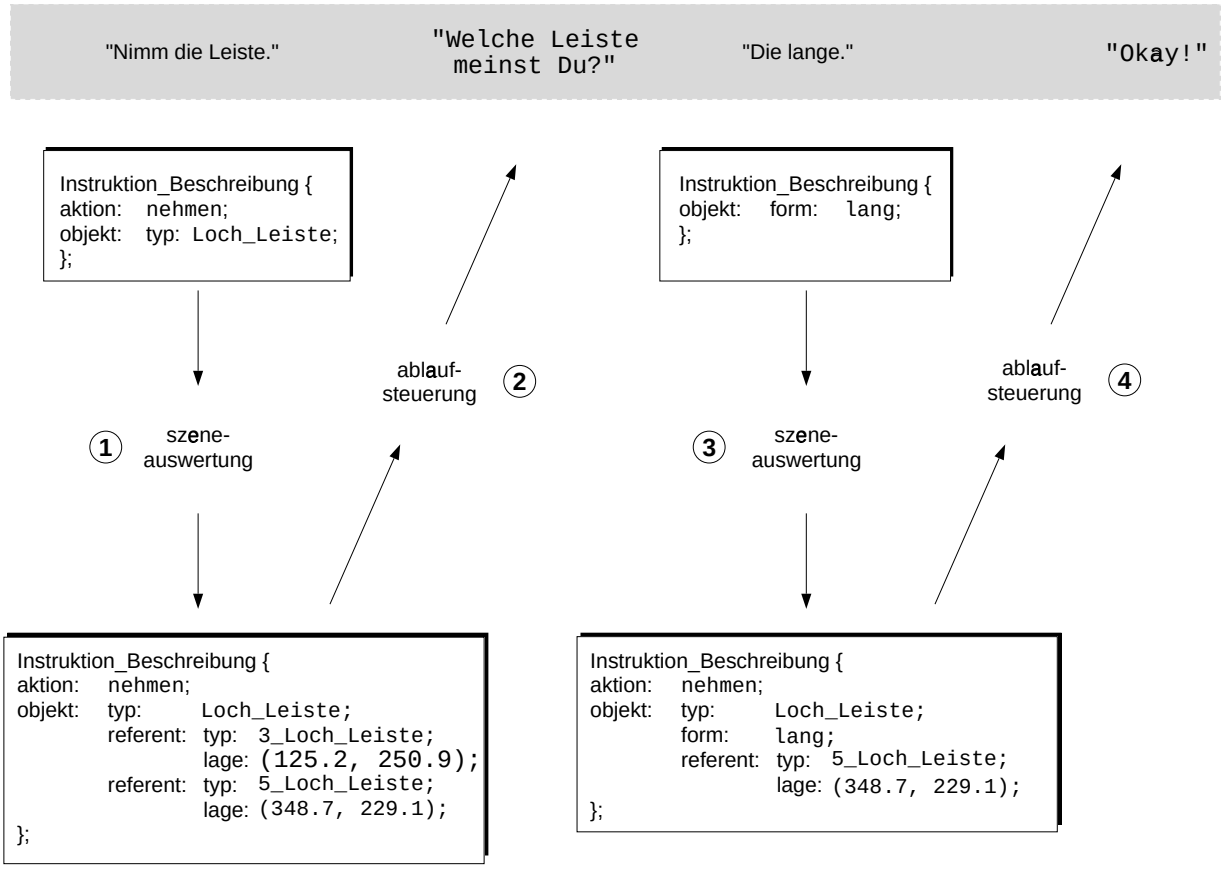

Abbildung 5.17: Dialogsteuerung

teursäußerung „Nimm die Leiste.“ führt auf der Beschreibungsebene zu der Berechnung des Merkmalswertes *Instruktion_Beschreibung*, in dem Werte *nehmen* und *Loch_Leiste* eingetragen sind. Die *szeneauswertung* ① fragt bei dem Modul Objektreferenz nach passenden Referenten und trägt das Ergebnis ein — im Beispiel seien zwei Leisten in der Szene, deren Schwerpunkte im Eintrag *lage* notiert sind. Auf dieser Grundlage entscheidet sich die Berechnung im Merkmal *ablaufsteuerung* ② für eine Rückfrage, die verkürzt „Welche Leiste meinst Du?“ lautet. Der Instrukteur antwortet mit der Ellipse „Die lange.“, was zum Merkmalswert mit dem einzigen Eintragswert *lang* führt. Diese Information wird nun im Merkmal *szeneauswertung* ③ zunächst mit der vorherigen Anweisung verbunden, so daß eine umfassendere Instruktionsbeschreibung vorliegt. Diese bildet die Grundlage der erneuten Anfrage beim Modul Objektreferenz, welches wegen der detaillierteren Information nunmehr nur noch einen Referenten liefert, der im Ergebnis der Szeneauswertung eingetragen wird. Die Ablaufsteuerung ④ kommt daher zu dem Ergebnis, daß die Handlung durchgeführt werden kann, stößt den Ausführungprozeß an und generiert eine entsprechende Nachricht für den Instrukteur, die in Abbildung 5.17 mit „Okay!“ abgekürzt ist.

Damit ist die Darstellung der ERNEST++–Wissensbasis abgeschlossen. Im folgenden Abschnitt geht es um die Frage, wie dieses Wissen durch die Anwendung der ERNEST++–Inferenzregeln und einer an das Problem angepaßten Kontrollstrategie zur Analyse der Spracheingabe und zur Dialogführung genutzt wird.

5.4 Analysestrategie

Die Analysestrategie in ERNEST++–Netzen muß im allgemeinen sicherstellen, daß das in den Konzeptdefinitionen und Berechnungsfunktionen modellierte Wissen angemessen aktiviert wird. Für die konkrete Analysestrategie der Verstehenskomponente bedeutet dies im wesentlichen die Realisierung des bereits im dargelegten Ablaufmodell skizzierten Zyklusses:

1. Analyse der aktuellen sprachlichen Eingabe unter Berücksichtigung der Analyseergebnisse der Bildverarbeitung und der Robotermeldungen

2. Anstoßen der Systemreaktion und Erläuterung der Systemreaktion für den Instrukteur

Im weiteren erläutere ich wie dieser Zyklus in ERNEST++ realisiert wurde.

Das Verfahren beginnt damit, daß die ERNEST++ Basiskontrolle mit dem Argument D_DIALOG aufgerufen wird. Durch die problemunabhängige Initialisierung entsteht der Suchbaumknoten [1] in der Abbildung 5.18. Dieser wird durch die Expansion der Kante D_DIALOG.*begrüßung* zu dem Knoten [2] erweitert, in dem MK_1(D_S_BEGRÜSSUNG) instantiiert werden kann. Infolgedessen wird auch MK_1(D_DIALOG) zur Instanz wie im Suchbaumknoten [3] zu sehen, denn in der

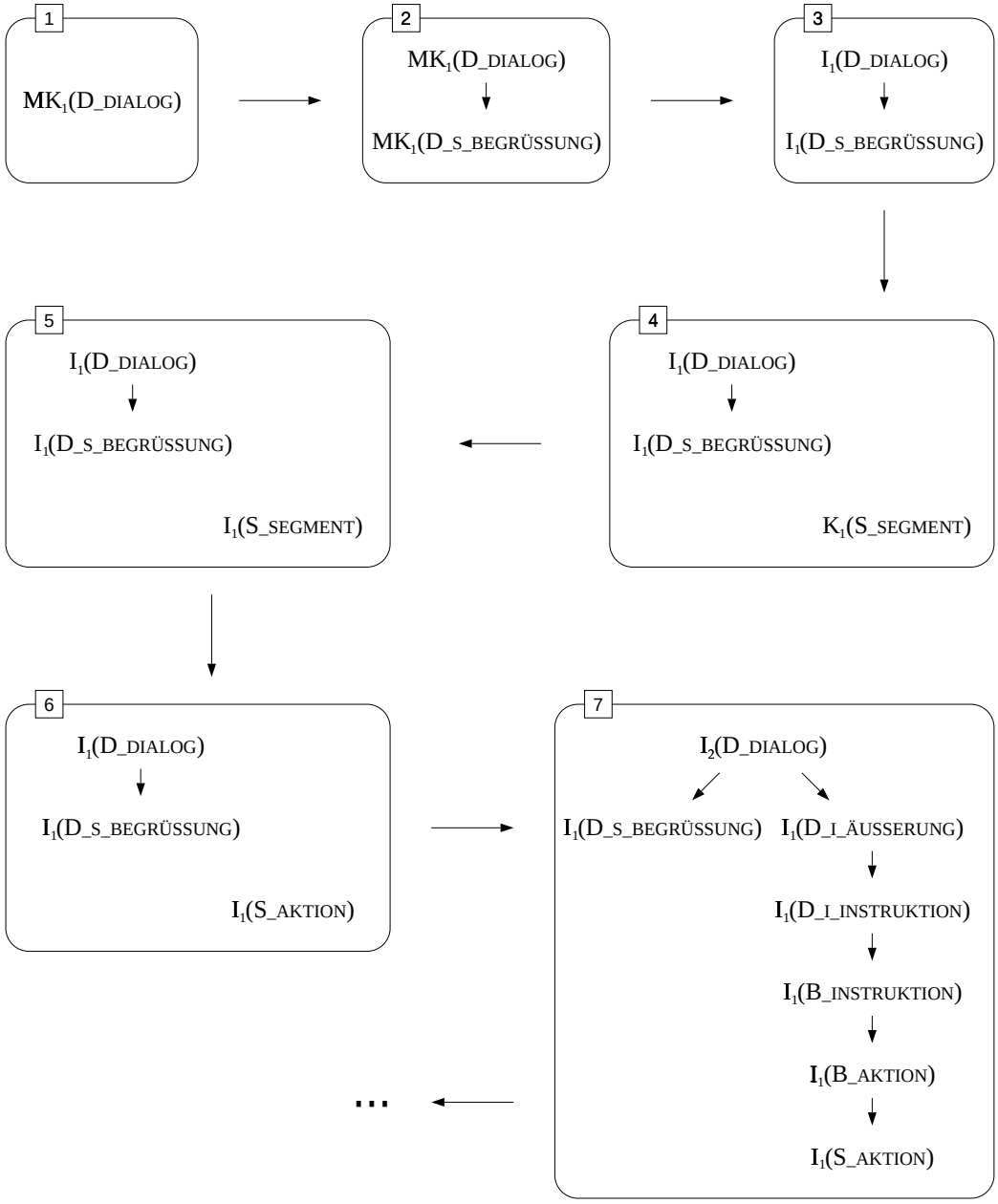

Abbildung 5.18: Die Verarbeitungsstrategie im ERNEST–Netz

Modalität von D_DIALOG ist nur die Kante *begrüßung* als obligatorisch deklariert. Nun muß das System in die Lage versetzt werden, dem Instrukeur zuzuhören. Das geschieht, indem dem Suchbaumknoten 3 ein ungebundenes Konzept S_SEGMENT hinzugefügt wird – wie in Knoten 4 gezeigt. Das weitere Verfahren möchte ich an der Beispieläußerung aus dem vorherigen Abschnitt „Nimm eine Leiste.“ erläutern. Sobald das erste Ergebnis des Spracherkenners, nämlich *(nimm: $AKTION)* vorliegt, führt dieses zu der Instanz I_1(S_SEGMENT) in Knoten 5. Aufgrund der Information des Spracherkenners, daß es sich bei diesem Stück Sprache um die Benennung einer Aktion handelt, kann I_1(S_SEGMENT) im nächsten Schritt zu I_1(S_AKTION) spezialisiert werden (Knoten 6). Diese Instanz wird modellgetrieben an I_1(D_DIALOG) gebunden, wodurch

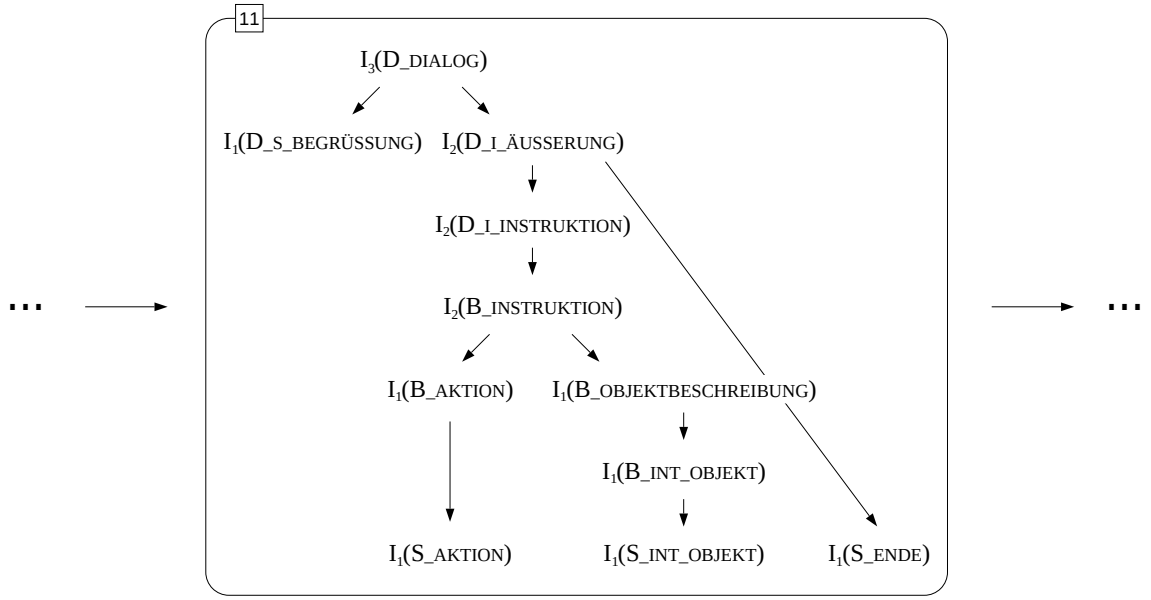

Abbildung 5.19: Die Verarbeitungsstrategie im ERNEST++–Netz (Fortsetzung)

in Knoten 7 eine neue Instanz I_2(D_DIALOG) entsteht. Schon zu diesem Zeitpunkt der Analyse liegt damit auf der obersten Ebene die Information vor, daß eine Nehmen–Aktion von statten gehen soll. Danach wird das nächste Segment vom Spracherkenner *(die Leiste: $INT_OBJEKT)* analog verarbeitet: ein Konzept von S_SEGMENT wird hinzugefügt, zu I_1(S_INT_OBJEKT) spezialisiert und modellgetrieben an I_2(D_DIALOG) gebunden. Das vom Spracherkenner sodann entgegengenommene Signal für das Ende der Äußerung führt mit derselben Vorgehensweise zu der Instanz I_1(S_ENDE), die ebenfalls an die abstrakteren Netzknoten gebunden wird. Es liegt dann im Suchbaumknoten 11 eine Situation vor, wie sie in Abbildung 5.19 dargestellt ist. In I_3(D_DIALOG) kann nun die Berechnung der nächsten Systemaktion erfolgen. Wie bereits bekannt soll das in diesem Beispiel eine Rückfrage sein. Daher wird im nächsten Knoten die optionale Kante D_DIALOG.*rückfrage* expandiert und dabei ein modifiziertes Konzept von D_S_RÜCKFRAGE eingefügt. Bei der Instantiierung dieses modifizierten Konzeptes in Knoten 13 wird die Rückfrage generiert (Abbildung 5.20). Nunmehr ist der erste Zyklus in der Mensch–Maschine–Kommunikation abgearbeitet. Da alle für die weitere Interaktion benötigten Informationen in I_3(D_DIALOG) enthalten sind, werden alle Netzknoten außer I_3(D_DIALOG) und I_1(D_S_BEGRÜSSUNG) gelöscht. Damit ist von der Struktur her derselbe Instanzenbaum wie im Suchkbaumknoten 3 gegeben und die Verarbeitung der Antwort auf die Rückfrage kann in gleicher Weise wie die Verarbeitung der ersten Instruktion angegangen werden.

Ich habe in diesem Beispiel den aktuellen Stand des Systems geschildert. Sollen auch die Konzeptdefinitionen D_R_NACHRICHT beziehungsweise D_S_NACHRICHT aktiviert werden, ist das Verfahren derart zu modifizieren, daß der Verarbeitungszyklus nicht nur mit dem Hinzufügen ungebundener Konzepte von S_SEGMENT, sondern auch von D_R_NACHRICHT und

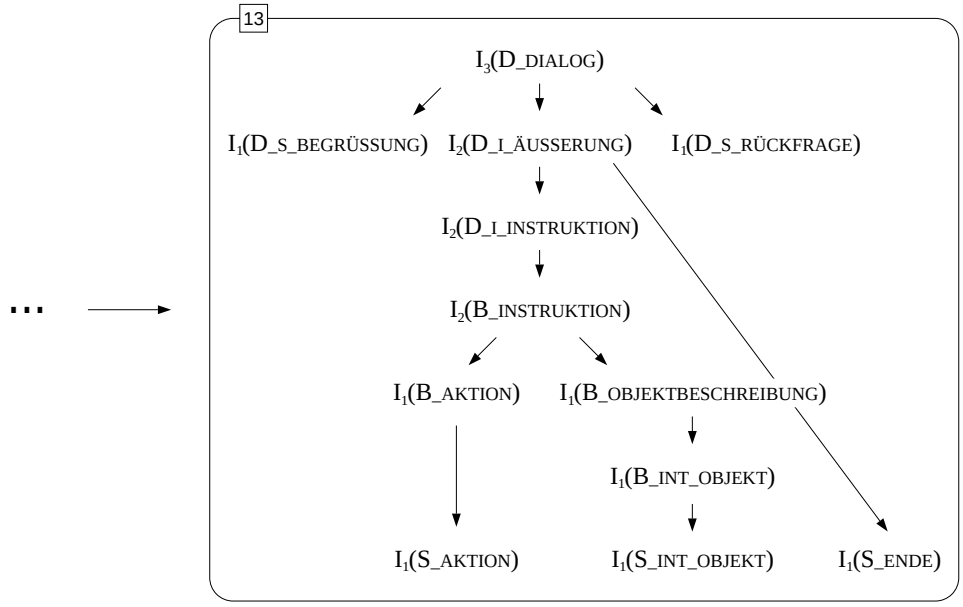

Abbildung 5.20: Die Verarbeitungsstrategie im ERNEST–Netz (Fortsetzung)

D_S_NACHRICHT beginnt. Das Konzept, welches durch ein entsprechendes Signal zuerst instantiiert wird, löst dann die nächste Systemreaktion aus. Sollten mehrere Informationen nahezu gleichzeitig eintreffen, müssten diese zunächst zwischengespeichert und dann verarbeitet werden. Damit wäre die adäquate Verarbeitung aller multimodaler Eingaben gesichert.

5.5 Einige besondere Aspekte der ERNEST–Modellierung und –Verarbeitungsstrategie

Nachdem in den letzten beiden Abschnitten das ERNEST–Netz und die Verarbeitungsstrategie vorgestellt wurde, möchte ich nun einige besondere Aspekte betrachten, welche die Leistungsfähigkeit der Verstehenskomponente insgesamt wesentlich beeinflussen.

Rein datengetriebene Analyse

Im Unterschied zu den bisher in ERNEST–Netzen verfolgten Strategien geschieht die Analyse der sprachlichen Eingaben rein datengetrieben. Bisher wurde eine gemischte Strategie verfolgt, in der während der Analyse des Signals zwischen datengetriebener und erwartungsgesteuerter — das heißt also modellgetriebener — Weiterentwicklung der Interpretation gewechselt wurde [Kum92, Seite 153]. Mit dieser Strategie verbunden war eine sehr aufwendige Modellierung der lingustischen Wissensbasis. Der Vorteil der rein datengetriebenen und nur auf der Analyse von Schlüsselphrasen basierenden flachen Analyse besteht vor allem in der größeren Effizienz. Durch die Konzentration auf die für das Verständnis der Instruktion wesentlichen Äußerungsteile

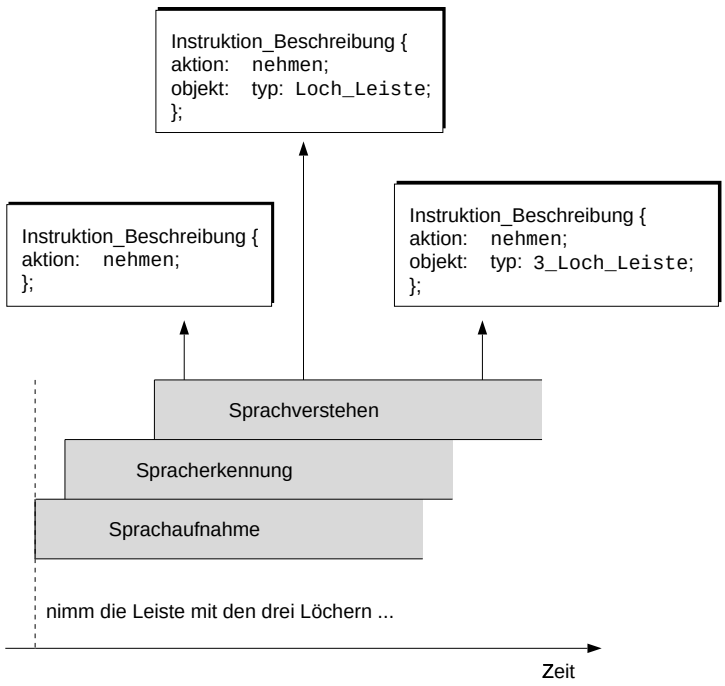

Abbildung 5.21: Inkrementelle Verarbeitung

entfällt die erwartungsgesteuerte Expansion des Suchbaums. Die Verarbeitungzeit einer Anweisung liegt daher im Millisekundenbereich (siehe Abschnitt 6.2.3). Außerdem besteht durch die wesentlich schlankere Wissenbasis und dem datengetriebenen Verarbeitungsalgorithmus eine viel einfachere Erweiterungsmöglichkeit und Wartbarkeit des Systems. Dies wird beispielhaft im Abschnitt 5.6 dargelegt.

Inkrementalität

Der verwendete Spracherkenner liefert die ersten Analyseergebnisse bereits nach etwa einer Sekunde [Fin98][18], also in der Regel vor dem Abschluß der Instruktion. Dieses inkrementelle Vorgehen wird auch vom sprachverstehenden Prozeß fortgesetzt. Wie dargelegt liegt durch die datengetriebene Bindung der instantiierten Netzknoten auf der Segmentebene das Analyseergebnis auf einer hohen Abstraktionsebene bereits vor, bevor das nächste Segment eingelesen wird. Somit besteht die Möglichkeit, auf der Dialogebene andere Prozesse, beispielsweise den Planungsprozeß, anzustoßen, bevor die Instruktion beendet ist. Abbildung 5.21 zeigt schematisch die inkrementelle Verarbeitung. Die Sprachaufnahme liefert etwa 200 Millisekunden nach Beginn der Instruktion die ersten Ergebnisse an die Spracherkennung, die ihrerseits nach etwa 750 Millisekunden die ersten Ergebnisse an die Verstehenskomponente gibt. Somit kann die Sprachverstehenskomponente etwa eine Sekunde nach Beginn der Instruktion mit ihrer Analyse

[18] Genaugenommen läßt sich der Zeitversatz, nach dem die ersten Ergebnisse ausgegeben werden sollen, per Kommandozeile angeben. Der experimentell gewonnene Wert von einer Sekunde sichert allerdings die benötigte Qualität der Ergebnisse.

beginnen. Nach dem ersten Zyklus[19] ist die Nehmen–Aktion detektiert, nach dem zweiten auch das zu nehmende Objekt, welches im dritten noch näher spezifiziert wird.

Integration von Ergebnissen zusätzlicher sprachverarbeitender Module

Bereits die Ergebnisse im Verbmobil–Projekt zeigten, daß eine Kombination verschiedener Verfahren der automatischen Sprachverarbeitung die besten Systemleistungen erbringt. Daher ist die Verstehenskomponente so angelegt, daß die Ergebnisse anderer sprachverarbeitender Module unmittelbar integriert werden können. Durch den modularen Aufbau der Verstehenskomponente wird die Kommunikation mit diesen Modulen unterstützt.

In [Hil97] wird ein auf der Kategorialgrammatik von Steedman [Ste93] basierender Parser vorgestellt, mit dem Konstruktionsanweisungen im Baufix–Szenario syntaktisch–semantisch analysiert werden. Der Anspruch in dieser Arbeit besteht nicht nur darin, die wichtigsten Informationen aus einer Anweisung zu filtern, um einen Roboter anzusteuern. Vielmehr geht es auch darum, die verschiedenen Lesarten, die in ein und derselben Anweisung verborgen liegen, zu extrahieren. Die Ergebnisse dieser tiefen linguistischen Analyse können im ERNEST$^{++}_{+}$–Netz statt der flachen Analyse, wie sie in der Segment– und Beschreibungebene durchgeführt wird, verwendet werden. Sie werden in den Instanzen von D_I_ÄUSSERUNG eingelesen und dann genauso weiterverarbeitet, wie die Ergebnisse der flachen Analyse. Die Kommunikation mit diesem Modul geschieht wie auch im bereits vorgestellten Gesamtsystem üblich mit Hilfe von DACS.

Wie geschildert werden die Objektbenennungen in den Systemausgaben mit Hilfe von Schablonen erzeugt. Ein konnektionistisches Modell für die Produktion von Objektbenennungen wird in [Sch98] dargestellt. Dort wird bei der Benennung von Objekten der Szenekontext berücksichtigt. Wenn ein auf diesem Modell basierendes Modul bei den Systemausgaben verwendet wird, sind angemessenere und natürlichere Objektbenennungen möglich. Beispielsweise wird dann eine Fünflochleiste im Kontext von Dreilochleisten als „lange Leiste“ benannt und im Kontext von Siebenlochleisten als „kurze Leiste“.

Es können noch die Resultate von zwei weiteren Ansätzen unmittelbar integriert werden. Im Unterschied zu den zuvor genannten Modulen ist diese Integration aber noch nicht realisiert. In [Bri98] wird ein Verfahren zur Gewinnung prosodischer Informationen erläutert, das allein auf dem Sprachsignal und gänzlich unabhängig von der Spracherkennung arbeitet. Beispielsweise können mit diesem Modul gestenbegleitende Betonungen erkannt werden, die im Merkmal B_INSTRUKTION.*beschreibung* einer Objektbeschreibung zugeordnet werden könnten, um die Disambiguierung des Referenten mit zusätzlichen Informationen zu unterstützen. [Kro98] ist der Verarbeitung diskontinuierlicher Konstituenten gewidmet, die häufig bei Nachtragskonstruktionen auftreten, wie zum Beispiel in „Jetzt nimmst du den Klotz und die Schraube — den

[19] Zyklus meint hier die Sequenz: Einfügen ungebundener Segmentkonzepte — Instantierung dieser ungebundenen Konzepte — modellgetriebene Bindung der Instanzen an I_j(D_DIALOG).

roten.“. Die flache Analyse, die keine Kongruenzüberprüfung bezüglich Kasus, Numerus und Genus vornimmt, würde fälschlicherweise annehmen, daß neben dem Schraubwürfel und der Schraube ein drittes Objekt benannnt wird. Bei einem entsprechenden Hinweis auf eine Nachtragskonstruktion des in [Kro98] präsentierten Verfahrens könnte in diesem Fall korrigierend eingegriffen werden.

Mit der Betrachtung dieser Aspekte ist die Vorstellung der Verstehenskomponente abgeschlossen. In Kapitel 6 wird sie unter verschiedenen Gesichtspunkten evaluiert. Zuvor möchte ich aber noch einen Ausblick auf die weitere Arbeit geben und das Kapitel zusammenfassen.

5.6 Ausblick

In diesem Ausblick möchte ich zwei Fragen behandeln, die meines Erachtens besonders vordringlich in der Weiterarbeit an der Sprachverstehenskomponente sind. Es geht zum einen um die Erweiterbarkeit des Systems um noch nicht modellierte Formulierungen und zum anderen um eine modifizierte Analysestrategie bei der Verarbeitung mehrdeutiger Anweisungen.

In den Wizard–of–Oz–Korpora findet sich eine Vielzahl von Formulierungen, die im derzeitigen Implementationsstand der Sprachverstehenskomponente noch nicht verarbeitet werden können. Der modulare Aufbau vereinfacht aber eine Erweiterung des Systems enorm. Bisher kann beispielsweise eine Benennung eines Ortes in der Szene — wie sie in „Leg’ die Leiste neben den roten Klotz.“ enthalten ist — nicht verarbeitet werden. Um solche Benennungen verarbeiten zu können, muß zunächst ein entsprechendes Segment definiert werden. Das semantische Netz ist um je eine Konzeptdefinition auf der Segmentebene und der Beschreibungsebene zu erweitern, deren Berechnungsergebnisse in der Klasse *Instruktion_Beschreibung* festzuhalten sind, die zu diesem Zweck um einen Eintrag *platz* zu ergänzen ist. Auf analoge Weise kann die Kompetenz des Systems auch bezüglich anderer Formulierungen vergrößert werden.

Ein großes Problem stellt in sprachverarbeitenden Systemen die parallele Verarbeitung unterschiedlicher Lesarten ein und derselben Anweisung dar. Anhand der Anweisung „Schraub’ die Raute mit der Schraube an den Würfel.“ möchte ich vorstellen, wie die Verarbeitung in der vorgestellten Sprachverstehenskomponente erfolgen muß. Die Anweisung enthält zwei Lesarten: entweder soll eine Raute, die bereits mit einer Schraube aggregiert ist, an einem Würfel befestigt werden oder eine Rautenmutter soll mit Hilfe einer Schraube am Würfel angebracht werden.[20] Zur Verarbeitung beider Lesarten muß im ERNEST^{++}–Netz das Segment, welches den Äußerungsteil „mit der Schraube“ enthält, gemäß den Lesarten aufgespalten werden, so daß bei der Spezialisierung der Instanzen von S_SEGMENT zwei konkurrierende Nachfolger des aktuellen Suchbaumknotens entstehen. Es muß im weiteren Verlauf der Äußerung sichergestellt wer-

[20] Beim derzeitigen Implementationsstand des Systems wird die zweite Lesart als Interpretationsergebnis ermittelt.

den, daß neu hinzukommende Segmente auch in die jeweils konkurrierenden Suchbaumknoten beziehungsweise deren Nachfolger eingetragen werden. Es erfolgt also keine Entwicklung des Suchbaums entsprechend dem A*–Algorithmus, sondern der Suchbaum wird voll expandiert. Zu einem späteren Zeitpunkt kann entweder eine Lesart gänzlich verworfen werden (etwa weil die Bildverarbeitungskomponente feststellt, daß sich kein Aggregat bestehend aus einer Rautenmutter und einer Schraube in der aktuellen Szene befindet) oder es kann aufgrund der Bewertung des Suchbaumknotens eine präferierte Lesart ausgewählt werden. Wie bereits erwähnt ist die derzeitige Verarbeitungszeit so klein, daß diese aufwendigere Analysestrategie benutzt werden kann, ohne daß es zu unzumutbaren Wartezeiten auf die Systemreaktion kommen wird.

5.7 Resümee

Mit der in diesem Kapitel vorgestellten Sprachverstehenskomponente konnte erstmals für diese Domäne ein komplettes Dialogsystem fertiggestellt werden, das in einem komplexen Gesamtsystem die spontansprachliche Interaktion mit einem virtuellen oder realen Roboter ermöglicht. Das Gesamtsystem besteht aus Modulgruppen zur Bildverarbeitung, zur Robotersteuerung und zur Sprachverarbeitung. Die Aufgaben der Sprachverstehenskomponente in dem Konstruktionsszenario bestehen vor allem in der Disambiguierung von Objektbenennungen, dem Verstehen von Handlungsaufforderungen, der Verarbeitung von Interventionen und der Bestimmung einer angemessenen Systemreaktion. Die Organisation des Wissens in der Verstehenskomponente ist vertikal angelegt, das heißt auf den verschiedenen Ebenen der Wissensbasis wird auch das Wissen aus anderen Ebenen benutzt.

Ein ursprünglich zur Verbesserung der Spracherkennungsergebnisse entwickelter Parser ermöglicht die Definition von Segmenten, die vom Spracherkenner während der Analyse des Sprachsignals als Ganzes erkannt und an die nachfolgenden Module weitergegeben werden. In der Segmentdefinition für das Konstruktionsszenario sind im wesentlichen die syntaktischen und semantischen Strukturen von Objektbenennungen und Handlungsverben modelliert. Daher sind traditionelle Aufgaben von Verstehenssystemen, nämlich die syntaktische Analyse des Gesprochenen und ihre Zuordnung zu semantisch–pragmatischen Einheiten, zu großen Teilen schon in den Spracherkennungsprozeß verlagert, der seinerseits durch die Segmentdefinition bei der Erkennungsaufgabe zusätzlich unterstützt wird. Dieses Verfahren bedeutet eine neuartige Verschränkung von Spracherkennung und Sprachverstehen.

Das ERNEST^{++}-Netz zum Sprachverstehen folgt dem Paradigma der flachen Analyse und gliedert sich in drei Ebenen. Die Segmentebene realisiert die Schnittstelle zum Spracherkenner. Auf der Beschreibungsebene werden die einzelnen Segmente zu sinnvollen Einheiten zusammengefaßt, und eine semantische Repräsentation dieser Konstituenten wird gewonnen. Die Interpretation

einer Äußerung umfaßt dann die Gesamtheit der Repräsentationen der einzelnen Konstituenten. Auf der Dialogebene werden die Instruktionen mit dem Dialogkontext verbunden, und es wird unter Berücksichtigung der aktuellen Szene eine geeignete Systemreaktion ausgewählt. Der Auswahl liegt ein Ablaufmodell zugrunde, welches sieben Systemzustände festlegt, die jeweils in Konzeptdefinitionen modelliert sind. Aufgabe der auf dem ERNEST^{++}-Netz basierenden Analysestrategie ist es, den Zyklus von sprachlicher Eingabe und angemessener Systemreaktion unter Berücksichtigung der Ergebnisse der anderen Module des Gesamtsystems sicherzustellen. Sie ist rein datengetrieben und arbeitet inkrementell. Der modulare Aufbau der Wissensbasis ermöglicht die einfache Integration von Ergebnissen anderer sprachverarbeitender Module zur Steigerung der Leistungsfähigkeit der Komponente. In der Sprachverstehenskomponente werden also die Vorteile der Modellierung mit semantischen Netzen, wie Modularität, Wohlstrukturiertheit und Kompaktheit der Wissensrepräsentation genutzt. Eine neue Verarbeitungsphilosophie für die ERNEST^{++}–Netze zum Sprachverstehen begegnet dem Vorwurf der ineffizienten Analyse, welcher der Modellierung mit semantischen Netzen mitunter gemacht wird, so daß insgesamt ein System zur sehr schnellen und robusten Interpretation gesprochener Sprache realisiert werden konnte.

Kapitel 6

Evaluierung

Grau, teurer Freund, ist alle Theorie und
grün des Lebens goldner Baum.
Johann Wolfgang Goethe

Die Evaluierung von Dialogsystemen, insbesondere solcher mit multimodaler Eingabe, hat in den letzten Jahren immer mehr Interesse gefunden [Sim93, Ara97, Moo98, Han98]. Das bisher ungelöste Problem besteht darin, einheitliche und somit vergleichbare Maßstäbe für Systeme zu finden, die in ganz unterschiedlichen Domänen arbeiten und mit verschiedenen Aufgabenstellungen konfrontiert sind. Hinzu kommt die Schwierigkeit, daß verschiedene Benutzer solcher Systeme ein und dieselbe Systemleistung möglicherweise ganz unterschiedlich bewerten.[1] Um die Bewertungsproblematik etwas besser in den Griff zu bekommen, wird häufig die Idee aus [Sim93] aufgegriffen und sowohl das Verhalten des Systems als Ganzes gegenüber dem menschlichen Benutzer bewertet (*black box evaluation*) als auch eine auf die Korrektheit der internen Repräsentation gerichtete Evaluierung durchgeführt (*glass box evaluation*). Auf diese Weise können zumindest für Teile solcher Systeme ansatzweise vergleichbare Zahlen, beispielsweise die Wortfehlerrate des spracherkennenden Moduls, gewonnen werden.

Die Evaluierung der im vorigen Kapitel vorgestellten Verstehenskomponente geschieht unter drei Fragestellungen:

1. Welche Fähigkeiten hat die Verstehenskomponente im Gesamtsystem?
   Im folgenden Abschnitt 6.1 wird ein Eindruck vom Verlauf eines Dialogs bei dem derzeitigen Stand der Sprachverstehenskomponente und des gesamten Konstruktionssystems gegeben.

[1] Araki schlägt daher sogar vor, ein System durch ein anderes automatische System, welches den Benutzer simuliert, zu evaluieren [Ara97].

2. Wie bewährt sich das System bei Anweisungen von verschiedenen Sprechern?
   Im Abschnitt 6.2 werden die Interpretationsleistung (glass box evaluation) sowie das Dialogverhalten (black box evaluation) betrachtet, wenn dem System Sprachdaten aus verschiedenen Stichproben eingespeist werden.

3. Wie allgemein ist das vorgestellte Verfahren?
   Schließlich wird untersucht, ob sich das vorgestellte Verfahren zur Verarbeitung gesprochener Sprache auch auf andere Domänen übertragen läßt.

Die Untersuchung der ersten beiden Fragestellungen dient der Verifikation des formulierten Anspruchs an die Verstehenskomponente, den Anforderungen in einem kooperativ gestalteten Dialog zwischen Mensch und Maschine gerecht zu werden. Die dritte Fragestellung beleuchtet, ob der vorgeschlagene Ansatz zur automatischen Sprachverarbeitung nicht zu viele Restriktionen des Konstruktionsszenarios berücksichtigt und er somit nur als eine Speziallösung für ein sehr eingeschränktes Problem gelten kann.

6.1 Kompetenz der Verstehenskomponente

Bei der Darstellung der Verstehenskomponente habe ich bereits eine Reihe von Beispielen gegeben, welche Arten von Äußerungen und Konstituenten die Verstehenskomponente verarbeiten kann. Im Anhang A.1 finden sich die definierten Segmente, die ja die Grundlage der möglichen Konstituentenstrukturen festlegen. Darum verzichte ich an dieser Stelle auf eine Auflistung von Konstruktionen, die verarbeitet werden können. Vielmehr möchte ich in diesem Abschnitt die Kompetenz der Verstehenskomponente als Hauptschnittstelle in der Mensch–Maschine–Kommunikation anhand eines repräsentativen Dialogs, der in Abbildung 6.2 wiedergegeben ist, veranschaulichen. Der Dialog gibt auch einen Eindruck von der Leistungsfähigkeit des Gesamtsystems, denn die Anweisungen wurden in einer realen Szene gegeben. Die Ausführung erfolgte in einer äquivalent simulierten virtuellen Szene. Abbildung 6.1 zeigt, wie sich die Szene

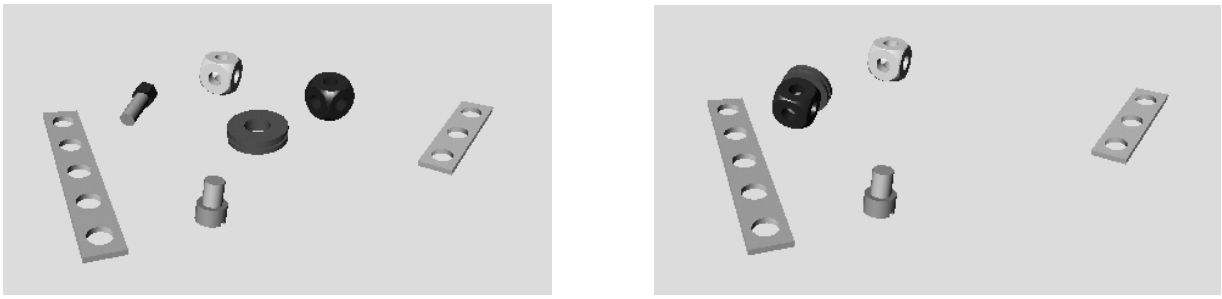

Abbildung 6.1: Visualisierung der Konstruktionsszene während der Mensch–Maschine–Kommunikation

verändert: im linken Bild befindet sich die Situation zu Beginn des Dialogs, während das rechte Bild das Ergebnis der ausgeführten Anweisungen am Ende des Dialogausschnittes zeigt. Die einzelnen Zwischenschritte in der Konstruktion, die ebenfalls visualisiert werden, sind hier nicht dargestellt. Anhang B gewährt im übrigen einen genauen Einblick in die internen Ergebnisse, die während der Analyse dieses Dialogs erarbeitet wurden.

Die Konstruktion beginnt mit der Begrüßung des Instrukteurs. Die initiale Bitte um eine Anweisung (System 1) ist sehr allgemein gehalten, um dem Instrukteur in keiner Weise zu restringieren. Als Reaktion auf die erste Anweisung erfolgt eine Rückfrage (System 2), denn die in der Instruktion enthaltene Objektbenennung („eine Schraube“) war unterspezifiziert. Bei der Rückfrage benennt die Verstehenskomponente die Referenten, auf welche die Benennung in der ersten Instruktion paßt und die daher noch zur Disambiguierung anstehen. Somit ist der

System 1: „Guten Tag! Ich kann mit Baufix spielen! Es geht sofort los! Ich warte auf Deine Anweisungen!“

Instrukteur 1: „Nimm eine Schraube!“

System 2: „Moment! Meinst Du die blaue Schraube oder die orange Schraube?“

Instrukteur 2: „Die blaue.“

System 3: „Okay, ich nehme die blaue Schraube. Was soll ich als nächstes tun?“

Instrukteur 3: „Steck’ dieses rote runde Teil auf die Schraube.“

System 4: „Okay, ich stecke die Felge auf die blaue Schraube. Was soll ich als nächstes tun?“

Instrukteur 4: „Ähm, jetzt nimm . . .“

System 5: „Moment! Da fehlt doch was! Um etwas nehmen zu können, brauche ich doch auch irgendwelche Objekte. Ich weiß aber nicht, welche Du meinst.“

Instrukteur 5: „Einen Klotz.“

System 6: „Entschuldigung, was hast Du gesagt? Ich weiß immer noch nicht, welches Teil Du meinst.“

Instrukteur 6: „Ich meinte den blauen Klotz.“

System 7: „Okay, ich nehme den blauen Würfel. Was soll ich als nächstes tun?“

Instrukteur 7: „Schraube den Würfel auf die blaue Schraube.“

System 8: „Okay, ich verschraube den blauen Würfel und die blaue Schraube. Was soll ich als nächstes tun?“

Abbildung 6.2: Dialogausschnitt zur Veranschaulichung der Kompetenz des Systems

Instrukteur über den Wissensstand des Systems informiert. Allerdings lädt die in der Rückfrage gewählte Formulierung zu einer elliptischen Objektbenennung in der Antwort ein, die auch prompt erfolgt (Instrukteur 2). Nach ihrer Verarbeitung ist das zu nehmende Objekt ausreichend spezifiziert, und das System informiert den Instrukteur darüber, was nun passiert (System 3). Die in der anschließenden Instruktion (Instrukteur 3) enthaltene Objektbenennung („dieses rote runde Teil“) reicht aus, um einen Referenten in der Szene zu bestimmen. Dies macht die Verstehenskomponente durch die konkretere Benennung „die Felge“ in der vierten Systemausgabe deutlich. Der Instrukteur erhält also eine implizite Verifikation seiner Anweisung. Die folgende Anweisung (Instrukteur 4) wird abgebrochen. Die bereits geäußerte Information wird allerdings von dem Verstehensmodul aufgenommen und dem Instrukteur auch mitgeteilt. Durch die Überprüfung der Argumente des geäußerten Verbs kann das System genau spezifizierten, warum keine sinnvolle Aktion möglich ist und was das System zur Behebung dieses Problems von dem Instrukteur erwartet (System 5). Seine Äußerung (Instrukteur 5) wird allerdings vom Spracherkenner überhaupt nicht erkannt und folglich auch nicht verstanden. Dieses Nichtverstehen wird dem Instrukteur explizit mitgeteilt. Gleichzeitig wird der Gesprächsfaden der vorhergehenden Instruktion wieder aufgegriffen und nach einem konkreten Objekt gefragt (System 6). Die anschließende Erläuterung des Instrukteurs enthält das Verb „meinen“, das keine direkte Aktion anweist. Das Verstehensmodul kann allerdings den Bezug zum bisherigen Dialog herstellen und bestätigt daher die Ausführung der Nehmen–Aktion (System 7). Die letzte Anweisung im Dialog verlangt eine etwas kompliziertere Aktion. Der Zusammenhang der Handlungsanweisungen wird erkannt: nach dem Nehmen folgt das Schrauben — das Nehmen ist also gewissermaßen Bestandteil der Schrauben–Handlung. Darum kann die Verstehenskomponente auch schlußfolgern, daß mit „den Würfel“ in der Instruktion 7 der zuvor genommene gemeint ist.

Die Verstehenskomponente wird in dem ausgewählten Dialog also den Anforderungen gerecht. Sie versteht die Anweisungen, kann mit spontansprachlichen Phänomenen wie Satzabbrüchen sowie Fehlern des Spracherkenners umgehen und führt im Zusammenspiel mit den anderen Modulen des Gesamtsystems eine angemessene Konstruktion durch. Im folgenden Abschnitt muß sich zeigen, ob diese Leistungsfähigkeit auch den rauhen Sprachdaten von verschiedenen Instrukteuren, die weder geübte Sprecher noch mit dem System vertraut sind, Stand hält.

6.2 Evaluierung anhand spontansprachlicher Daten

Die Auswertung der Leistungsfähigkeit von sprachverstehenden Systemen mit Daten von verschiedenen Sprechern dient vor allem der Überprüfung der Robustheit: kann das System mit den besonderen Phänomenen der gesprochenen Sprache umgehen? Wird das System der Vielfalt der Formulierungen gerecht, wie sie verschiedene Instrukteure wählen? Diese und andere Fragen lassen sich nur sehr schwer anhand von Einzelbeispielen umfassend beantworten. Ich werde sie

daher in diesem Abschnitt anhand verschiedener Teststichproben erörtern. Teilweise wurden die Ergebnisse bereits in [BP99] publiziert.

6.2.1 Evaluierungsdaten

Zur Auswertung unter verschiedenen Aspekten habe ich Ergebnisse auf sechs Stichproben erzeugt, deren charakteristische Daten in Tabelle 6.1 dargestellt sind.[2]

	WOZ–I	WOZ–II	SFB–I	SFB–II	Instrukt–I	Instrukt–II
Art	MCI	MMI	MMI	MMI	MCI	MCI
Anzahl Wörter	38.927	11.187	16.107	1.647	2.220	1.393
Anzahl Äußerungen	3.236	492	1.195	110	453	174
Anzahl Sprecher	40	10	14	8	10	6

Tabelle 6.1: Charakteristika der verwendeten Teststichproben

Es handelt sich bei allen Stichproben um spontansprachliches Material, das heißt die Instrukteure haben frei formuliert. Die Wizard–of–Oz–Korpora (WOZ–I und WOZ–II) sind bereits aus Abschnitt 3.3 bekannt.[3] Die Stichproben SFB–I und SFB–II sind dem SFB–Korpus [Bri95b], also Mensch–Mensch–Dialogen, entnommen. Sie unterscheiden sich allerdings in den Aufnahmebedingungen. Während die SFB–I–Stichprobe solche Anweisungen umfaßt, in denen die Beteiligten Sichtkontakt hatten, besteht die SFB–II–Stichprobe aus solchen Instruktionen, bei denen zwischen Konstrukteur und Instrukteur eine Sichtblende aufgebaut war. Die Stichproben Instrukt–I und Instrukt–II wurden gänzlich unabhängig von den Konstruktionsdialogen im Rahmen von Arbeiten zur Objektreferenz [Soc97] gewonnen. Den Instrukteuren wurde die Pseudo–Systemausgabe „Welches Objekt soll ich nehmen?“ gezeigt. Daraufhin hatten sie ein markiertes Objekt im Bild zu benennen. Bei der Aufnahme der Stichprobe Instrukt–II wurden sie explizit gebeten, die Objekte mit Hilfe von Referenzobjekten zu benennen. Die Stichproben bestehen daher aus Anweisungen wie „Ähm — nimm die Dreilochleiste.“ oder „Gib mir die Leiste neben dem Reifen.“.

6.2.2 Validität der Segmentdefinition

Ein Kriterium für die Validität der Segmentdefinition ist ihre Abdeckungsrate. Sie zeigt, ob die richtigen Modelle definiert worden sind und ob deren Spezifikation beziehungsweise Umsetzung angemessen ist. Wegen der Menge der dabei zu verarbeitenden Daten, kam eine manuelle

[2] In Tabelle 6.1 steht MMI für Mensch–Mensch–Interaktion und MCI für Mensch–Computer–Interaktion.

[3] Analog zur Benennung des Wizard–of–Oz–Korpus–I wird die Summe der Instruktionen des Szenarios II in der Wizard–of–Oz–Studie als Wizard–of–Oz–Korpus–II bezeichnet.

Auszählung der Äußerungen nicht in Frage. Daher wurde der Spracherkenner in einer Konfiguration gestartet, die es erlaubt, geschriebenen Text als Eingabe zu verwenden. Der Spracherkenner liest diesen Text ein, wendet die Segmentdefinition auf ihn an und gibt das Ergebnis aus. Bei der Verarbeitung erstellt er eine Statistik über die Abdeckung, also über die Rate der eingelesenen Wörter, die in einem Segment oder Subsegment enthalten sind. Die Ergebnisse wurden anhand sehr eng am Signal orientierter Transkripte erzeugt, was sich im Endeffekt leicht negativ auswirkte. Beispielsweise kann in der Äußerung 03i046 aus der verwendeten Stichprobe SFB–II

„Jetzt mußt du die roten Scheiben erstmal in die Gummi — reifen stecken."

die durch eine kleine Pause unterbrochene Konstituente „die Gummireifen" nicht einem Segment zugewiesen werden, denn der Referenztext sieht wie folgt aus

jetzt mußt du die roten Scheiben erstmal in die Gummi_reifen stecken ;

und das Wort „Gummi_reifen" befindet sich natürlich nicht im Vokabular des Systems. Für diese Äußerung liefert der Spracherkenner

jetzt (mußt: $AKTION) (du: $AGENT) (die roten Scheiben: $INT_OBJEKT)
erstmal in die Gummi Reifen (stecken: $AKTION) ;

Damit erreicht die Segmentdefinition in dieser Äußerung eine Abdeckungsrate von genau fünfzig Prozent: von den zwölf gesprochenen Wörtern werden sechs von Segmenten erfaßt.

Die Tabelle 6.2 gibt die erzielten Raten wieder. Von besonderem Interesse sind die Ergebnis-

WOZ–I	WOZ–II	SFB–I	SFB–II	Instrukt–I	Instrukt–II
57 %	49 %	39 %	48 %	67 %	93 %

Tabelle 6.2: Abdeckungsraten der Segmentdefinition

se auf den relativ großen Stichproben WOZ–I, WOZ–II und SFB–I, weil sie die größte Vielfalt in den Formulierungen aufweisen. Auf der WOZ–I–Stichprobe konnte erwartungsgemäß das beste Ergebnis bei diesen großen Stichproben erreicht werden, denn sie diente ja auch als Korpus bei den Untersuchungen zur Modellbildung: 57 Prozent des gesamten Textes besteht aus den Schlüsselphrasen, die in der Segmentdefinition festgelegt sind. Auf den simulierten Mensch–Mensch–Dialogen im Wizard–of–Oz–Korpus–II beträgt die Rate immerhin noch 49 Prozent. Es zeigt sich also an dieser Zahl erneut die Andersartigkeit von Mensch–Mensch– und Mensch–Maschine–Kommunikation. Die Aufnahmebedingung der SFB–I–Stichprobe, nämlich der Blickkontakt von Konstrukteur und Instrukteur, die zudem teilweise einen gemeinsamen Blick auf die Konstruktionsszene hatten, verändert den Inhalt und die Struktur der Äußerungen sehr stark. Insbesondere wird in den Dialogen so kooperativ gearbeitet, daß die Instrukteursäußerungen häufig extrem deiktisch und elliptisch sind wie zum Beispiel die Äußerung „senkrecht dazu — genau" (04I030). Das ist der Grund dafür, daß die Segmentdefinition nur eine

Abdeckungsrate von 39 Prozent erreicht. In der SFB–II–Stichprobe ist die Abdeckung wiederum etwas besser; die Answeisungen ähneln aufgrund der Aufnahmebedingungen der WOZ–II–Stichprobe, so daß die Rate 48 Prozent beträgt. Sehr hoch sind die Raten, die auf den Stichproben Instrukt–I und Instrukt–II erzielt wurden, denn die dort gewählten Formulierungen passen exakt zu dem Szenario, für das die Segmentdefinition als Teil der Sprachverstehenskomponente entwickelt wurde. Die Stichprobe Instrukt–I enthält allerdings linguistische Hecken wie „Die Fünferleiste ganz oben." und Angaben, die sich auf das Bild beziehen wie „Ich möchte die Siebenerleiste rechts im Bild.". Beide Phänomene sind der Segmentdefinition nicht modelliert. Das ist der wesentliche Grund für die geringere Abdeckung auf dieser Stichprobe.

Die Qualität der Segmentdefinition soll nun an ihrer Wirksamkeit bei der Spracherkennung bewertet werden. Dazu konnten nur die Stichproben SFB–II, Instrukt–I und Instrukt–II verwendet werden, weil die anderen Stichproben Teil des Trainingsmaterials des Spracherkenners waren. Tabelle 6.3 zeigt die erzielten Wortfehlerraten. Dieses Maß gibt nicht nur die Rate der korrekt

	Instrukt–I	Instrukt–II	SFB–II
HMM–Modellierung	54,5 %	39,8 %	73,4 %
HMM mit Bigramm	33,6 %	27,3 %	49,7 %
HMM mit Segmentdefinition	39,9 %	**20,5** %	71,8 %
HMM mit Bigramm und Segmentdefinition	**31,8** %	23,5 %	**49,2** %

Tabelle 6.3: Erzielte Wortfehlerraten

erkannten Wörter an, sondern berücksichtigt auch fehlerhafte Einfügungen des spracherkennenden Moduls [Lee89]. Somit kann die Leistung des Spracherkenners exakt bewertet werden. Das jeweils beste Ergebnis auf einer Stichprobe ist fett gedruckt.

In der ersten Zeile zeigt die Tabelle die Ergebnisse, die auschließlich mit der akustischen Modellierung mit Hilfe von HMMs erreicht wurden: die Wortfehlerrate liegt zwischen 73,4 und 39,8 Prozent. Diese Zahlen geben einen ersten Eindruck von dem Schwierigkeitsgrad, den die einzelnen Stichproben für die Spracherkennung bedeuten. Insbesondere mit der Stichprobe SFB–II hat der Spracherkenner sehr große Probleme: fast drei Viertel der ausgegebenen Wörter wurden falsch erkannt. Es bestätigt sich an den hohen Wortfehlerraten, was bereits bei der Abdeckungsrate sichtbar wurde: die Modelle in der Segementdefinition greifen zu wenig, um die Erkennungsergebnisse zu verbessern. Insbesondere ist keine signifikante Verbesserung zu erzielen, wenn neben der akustischen Modellierung die Segmentdefinition allein eingesetzt wird (71,8 gegenüber 73,4 Prozent). Das zentrale Ergebnis der Tabelle ist aber, daß auf den beiden Teststichproben, die aus einer Mensch–Maschine–Kommunikation stammen, die Kombination der Segmentdefinition mit dem Bigramm eine signifikante Verbesserung der Wortfehlerrate gegenüber der ausschließlichen Verwendung des Bigramms erbrachte. Damit ist das Ziel des Einsatzes der

Segmentdefinition aus der Sicht der Spracherkennung erreicht und ihre Angemessenheit für den Anwendungsbereich belegt. Auffallend gut ist das Ergebnis der Segmentdefinition auf der Stichprobe Instrukt–II: ihre alleinige Verwendung erzielt mit 20,5 Prozent Wortfehlerrate das beste Ergebnis. Hier greift die Segmentdefinition also sehr gut, was an der bereits dargelegten hohen Abdeckungsrate auf dieser Stichprobe (93 Prozent) liegt. Bei der Kombination bewirkt der Einfluß des Bigramms sogar eine Verschlechterung dieses Ergebnisses auf 23,5 Prozent.

Diese Ergebnisse bilden die Grundlage für die Interpretation der Anweisungen, deren Resultate im folgenden Abschnitt dargestellt werden.

6.2.3 Semantische Interpretation und Dialogverhalten der Verstehenskomponente

Die Evaluation der Interpretationsleistung gliedert sich in drei Punkte. Zunächst untersuche ich, wie gut die konstruktionsrelevanten Segmente interpretiert werden. Danach betrachte ich die Analyse ganzer Äußerungen und schließlich beschreibe ich das Dialogverhalten der Verstehenskomponente.

Tabelle 6.4 zeigt die Interpretation der konstruktionsrelevanten Segmente in den Stichproben

	Stichprobe	Konfiguration des Spracherkenners	vollständig interpretiert	partiell interpretiert
1	Instrukt–I	HMM	30,2 %	8,9 %
2	Instrukt–I	HMM mit Bigramm	45,6 %	15,6 %
3	Instrukt–I	HMM mit Segmentdefinition	**54,2** %	**16,0** %
4	Instrukt–I	HMM mit Bigramm und Segmentdefinition	52,7 %	13,3 %
5	Instrukt–II	HMM	37,3 %	1,2 %
6	Instrukt–II	HMM mit Bigramm	61,6 %	1,7 %
7	Instrukt–II	HMM mit Segmentdefinition	**69,8** %	**4,6** %
8	Instrukt–II	HMM mit Bigramm und Segmentdefinition	67,1 %	4,0 %

Tabelle 6.4: Erzielte korrekte Interpretationen von konstruktionsrelevanten Segmenten

Instrukt–I und Instrukt–II, das heißt aller Segmente außer $S_AGENT und $S_SATZWORT[4]. Ein Segment ist „vollständig interpretiert“, falls alle genannten Attribute (wie Typ, Form oder Farbe) auch in der semantischen Repräsentation enthalten sind. Kann dagegen nicht alles korrekt

[4] Wie schon erwähnt ist ja die Benennung eines Agenten („du“, „Sie“, „wir“) für die Konstruktionsaufgabe vergleichsweise irrelevant, denn der Adressat der Instruktion ist stets der Roboter. Auch Floskeln und Ein–Wort–Sätze, die durch $S_SATZWORT abgedeckt sind, werden derzeit nicht für die Interpretation einer Anweisung genutzt.

erfaßt werden, wird dieses Segment mit „partiell interpretiert“ bewertet. Die Einführung dieses Bewertungsmaßes ist deshalb wichtig, weil oftmals schon eine partielle Interpretation ausreicht, den Referenten zu bestimmen, wie in [Wac99] ausführlich dargelegt wird. In den Ergebnissen der Zeilen 1, 2, 5 und 6 hatte die Segmentdefinition überhaupt keinen Einfluß auf die Spracherkennung — ähnlich wie bei Ermittelung ihrer Abdeckung wurde die Segmentdefinition einfach auf das Ergebnis des Spracherkenners angewendet. Bei den Ergebnissen in den Zeilen 3 und 7 wurde ausschließlich die Segmentdefinition zusätzlich verwendet, während die Zeilen 4 und 8 die Ergebnisse beinhalten, die durch die Kombination von Segmentdefinition und Bigramm erreicht wurden.

Die Ergebnisse auf der Stichprobe Instrukt–II (zwischen 37,3 und 69,8 Prozent vollständig interpretiert) sind durchweg besser als die auf der Instrukt–I–Stichprobe (zwischen 30,2 und 54,2 Prozent vollständig interpretiert). Darin spiegeln sich die erzielten Wortfehlerraten wider. Erwartungsgemäß war die Interpretationsleistung am schlechtesten, wenn nur die akustische Modellierung bei der Spracherkennung verwendet wurde: die Ergebnisse des Spracherkenners waren dann so schlecht (vergleiche Tabelle 6.3), daß nur 30,2 beziehungsweise 37,3 Prozent vollständig interpretiert werden konnten.[5] Die besten Ergebnisse erbrachten 74,4 beziehungsweise 70,2 Prozent vollständig oder partiell korrekt interpretierte konstruktionsrelevante Segmente. Sie wurden dann erzielt, wenn die Segmentdefinition allein den Spracherkennungsprozeß unterstützt. Dies gilt interessanterweise auch für die Instrukt–I–Stichprobe, in der ja die Kombination von Segmentdefinition und Bigramm die niedrigste Wortfehlerrate erbrachte. Im Hinblick auf die Interpretation kann also das beste Ergebnis dann erreicht werden, wenn die Segmentdefinition schon sehr früh den Verarbeitungsprozeß stark beeinflußt, während unter Umständen eine Kombination von Bigramm und Segmentdefiniton die reine Wortfehlerrate optimiert.

Analog zur Bewertung der Interpretation konstruktionsrelevanter Segmente erfolgt die Beurteilung der Interpretation ganzer Äußerungen (siehe Tabelle 6.5). Zusätzlich zu „vollständig

Stichprobe	vollständig interpret.	partiell interpret.	falsch interpret.	nicht interpret.
Instrukt–I	41,7 %	34,6 %	9,1 %	14,6 %
Instrukt–II	45,4 %	51,7 %	2,9 %	0 %

Tabelle 6.5: Interpretation ganzer Äußerungen

interpretiert“ und „partiell interpretiert“ kann eine Interpretationsleistung mit „falsch interpretiert“ bewertet werden, wenn die semantische Repräsentation der Anweisung in keiner Weise entspricht. „nicht interpretiert“ sind solche Anweisungen, aus denen überhaupt keine Informati-

[5] Die Spracherkennungsergebnisse waren auch der Grund dafür, daß auf eine Auszählung der Interpretationsleistung auf der SFB–II–Stichprobe gänzlich verzichtet wurde.

on gezogen werden konnte. Über 97 Prozent der Äußerungen der Stichprobe Instrukt–II werden ganz oder teilweise korrekt interpretiert — bei der Instrukt–I–Stichprobe sind es über 76 Prozent. Auffallend ist der jeweils hohe Prozentsatz der nur partiell interpretierten Äußerungen. Dies hat seine Ursache in den detaillierten Objektbenennungen: schon wenn nur ein Adjetiv nicht erkannt wurde — beispielsweise werde „der blaue Würfel“ als „der Schraubwürfel“ weiterverarbeitet — kann die Analyse der gesamten Äußerung nicht vollständig korrekt bewertet werden, selbst wenn nur ein Schraubwürfel im Bild und der Referent damit völlig klar ist.

Ein formulierter Anspruch an die Verstehenskomponente ist eine effiziente Verarbeitung der Anweisungen. In Tabelle 6.6 sind einige diesbezügliche Daten zusammengestellt. Die ange-

	Stichprobe Instrukt–I	Stichprobe Instrukt–II	Stichprobe SFB–II
Anzahl Wörter pro Äußerung	4,9	8,0	15,0
Verarbeitungszeit insgesamt	5.083 ms	1.950 ms	12.199 ms
Verarbeitungszeit pro Äußerung	11,2 ms	11,2 ms	110,9 ms

Tabelle 6.6: Verarbeitungsgeschwindigkeit der Verstehenskomponente

gebenen Zeiten umfassen sowohl die in Systemfunktionen verbrauchte Zeit (*system time*) als auch den Zeitbedarf der Programmfunktionen (*user time*). Die Zeiten wurden auf einer DEC AlphaStation 500/500[6] erzielt. Für die Verarbeitung einer Anweisung aus den beiden Stichproben Instrukt–I und Instrukt–II wurden im Durchschnitt nur 11,2 Millisekunden benötigt. Die Analyse braucht also weniger Zeit, als die Artikulation der Anweisung. Als Vergleich dienten die Mensch–Mensch–Dialoge der SFB–II–Stichprobe. Es zeigt sich, daß die Verarbeitungszeit nicht linear mit der Länge der Anweisung ansteigt. Das liegt vor allem an der Verwaltung der ERNEST++–Suchbaumknoten, die bei langen Äußerungen sehr groß werden können. Nichtsdestotrotz ist auch die auf dieser Stichprobe erzielte Verarbeitungszeit von 110,9 Millisekunden pro Anweisung sehr zufriedenstellend. Die Verstehenskomponente wird daher in keinem Fall die Verarbeitungszeit des gesamten Konstruktionssystems beeinträchtigen.

Abschließend soll an den spontansprachlichen Daten auch das Dialogverhalten untersucht werden. Das Problem dabei besteht darin, daß die Äußerungen in den Teststichproben nicht einfach hintereinandergehängt werden und dann als Sequenz von Anweisungen eines Instrukteurs angesehen werden können. Denn die Äußerungen sind ja völlig zusammenhanglos. Daher habe ich die Äußerungen, zu denen auch das Bildmaterial vorliegt, als initiale Anweisungen betrachtet und einzeln in das gesamte Konstruktionssystem eingespeist. Jede Äußerung wurde also interpretiert und auf der Grundlage der erzielten semantischen Repräsentation und der gegebenenfalls gewonnenen Referenten die Systemreaktion festgelegt. Alle Anweisungen entstammen der

[6] 500 Mhz CPU–Takt, 1.024 MB Hauptspeicher, etwa 15 SPECint95 und 20 SPECfp95

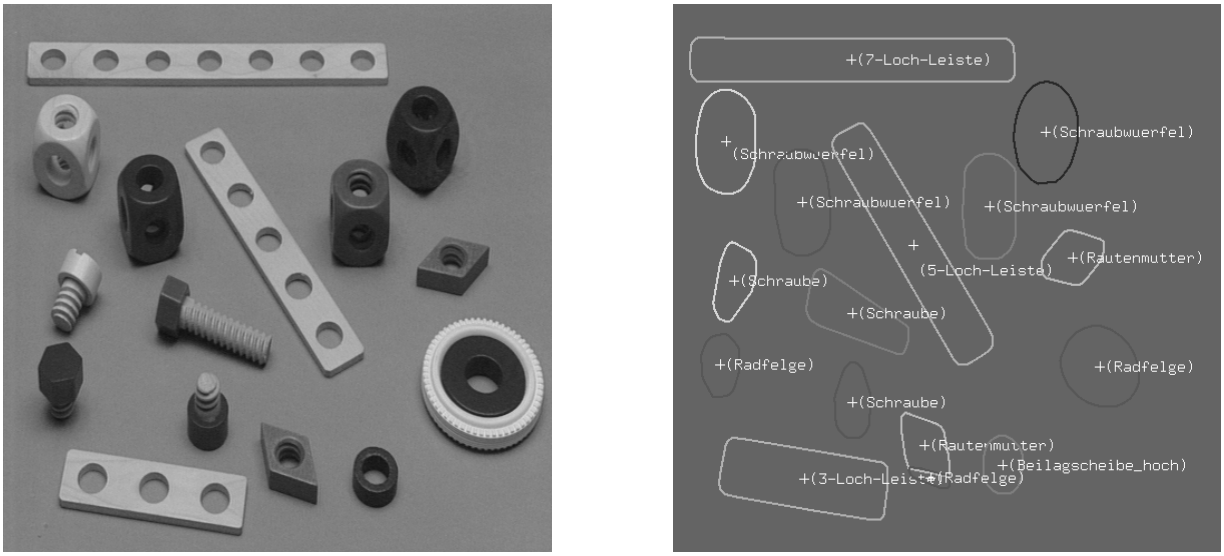

Abbildung 6.3: Typische Szene bei der Bewertung des Dialogverhaltens

Instrukt–II–Stichprobe. Abbildung 6.3 zeigt eine typische verwendete Szene und das Ergebnis der Bildanalyse dazu, das bezüglich zweier Objekte fehlerhaft ist. Eine Schraube wird fälschlicherweise als Felge identifiziert und von dem Rad wird nur die innere Felge erkannt.

Das System konnte alle 144 Äußerungen verarbeiten. Keinmal gab es die Bitte nach einer Wiederholung der Anweisung, weil überhaupt keine Information gewonnen werden konnte oder gar einen Systemabsturz. Tabelle 6.7 zeigt die gewählten Systemreaktionen. Für 38,8 Prozent der Anweisungen wurde die Ausführung bestätigt wie etwa die folgende Anweisung[7]:

> „Gib mir die Dreilochleiste vor der roten Schraube!“ (201/1012)
> `Okay, ich nehme die Leiste. Was soll ich als nächstes tun?`

Allerdings war die Bestätigung in sieben Fällen nicht korrekt, denn dreimal lag eine fehlerhafte Interpretation vor und viermal wurde von der Bildanalyse ein falsches Objekt ausgewählt. Die meisten Anweisungen konnten nicht direkt ausgeführt werden, sondern ihnen folgte eine

Äußerungen	Ausführung bestätigt (38,8 %)			Rückfrage (61,2 %)			
	korrekt	falsche Interpret.	falscher Referent	zuwenig Objekte	kein Referent	welcher Referent	welche Aktion
144	49	3	4	1	4	40	43
100 %	**34,0** %	2,1 %	2,8 %	0,7 %	2,8 %	27,8 %	29,9 %

Tabelle 6.7: Dialogverhalten des Systems

[7] Wie die anderen Zitate in diesem Abschnitt bezieht sich die Anweisung auf die Szene in Abbildung 6.3. Die jeweils synthetisierte Systemreaktion ist gesperrt gedruckt.

Rückfrage, die gemäß Algorithmus 5.1 auf Seite 92 in vier verschiedene Arten unterteilt werden kann. Eine Anweisung begann mit „Gib mir eine …“ und wurde dann abgebrochen, so daß eine Rückfrage nach dem zu nehmenden Objekt gestellt wurde, wie die Ausgabe System 5 im Dialog auf Seite 111. Viermal konnte kein Referent bestimmt werden, je zur Hälfte von der Bild– und Sprachverarbeitung verursacht:

„Ich möchte die Leiste rechts neben der Schraube.“ (bm21/1014)
`Ich finde die Leiste nicht. Was soll ich jetzt machen?`

Dem Instrukteur wird also durch die Objektbenennung in der Rückfrage signalisiert, daß seine Anweisung zwar verstanden wurde, aber nicht ausgeführt werden konnte. In 27,8 Prozent der Fälle war dem System der Referent unklar, weil zu der Objektbenennung mehrere Referenten unterschiedlichen Typs bestimmt wurden. Die Ursachen liegen etwa zu gleichen Teilen in der Bild– und in der Sprachverarbeitung. Dann erfolgte eine Rückfrage der folgenden Art:

„Ich möchte die Schraube vor der gelben Schraube.“ (bm21/1012)
`Moment! Meinst Du die rote Schraube oder die grüne Schraube?`

Diese Anweisung ist auch ein Beispiel dafür, daß trotz einer fehlerhaften Leistung des Systems — die spezifizierte Schraube wurde von der Bildverarbeitung nicht erkannt — der Dialog angemessen fortgesetzt werden kann. Auf 29,9 Prozent der Anweisungen wurde mit einer Rückfrage zu der gewünschten Aktion reagiert:

„Gib mir die Raute links neben der Scheibe!“ (101/1010)
`Das habe ich noch nicht ganz verstanden. Was soll ich mit`
`der Raute machen?`

Die Ursache dafür, daß die Handlungsabsicht häufig zunächst nicht verstanden werden konnte, liegt vor allem an den sehr kurzen Konstituenten wie „nimm“ oder „gib“, die für die Spracherkennung eine große Herausforderung darstellen und häufig nicht erkannt wurden.

Betrachtet man das Dialogverhalten insgesamt, hat das System als Ganzes auf dieser Stichprobe also nur siebenmal falsch gehandelt, indem für nicht intendierte Objekte die Handlungsausführung bestätigt wurde. In allen anderen Fällen war die Systemreaktion für einen Instrukteur nachvollziehbar und trug zur Klärung von Unklarheiten bei. Damit ergibt sich in Anlehnung an Wahlster [Wah97] eine *approximativ korrekte Systemleistung* für über 95 Prozent der Äußerungen.

6.3 Übertragbarkeit des vorgestellten Verfahrens

Nachdem ich in den letzten beiden Abschnitten gezeigt habe, daß die Verstehenskomponente die ursprünglich formulierten Ansprüche in dem Konstruktionszenario erfüllt, möchte ich in

diesem Abschnitt das vorgestellte Verfahren als solches bewerten. Dazu werde ich das Szenario verlassen und eine Sprachverstehenskomponente vorstellen, in der mit genau demselben Verfahren die Sprachverstehensaufgabe in einer anderen Domäne, nämlich der Zugauskunft, gelöst wird [Von99]. Das sprachverarbeitende System verwendet zum einen die in dieser Arbeit vorgestellten Methodiken, Strukturen und Algorithmen und orientiert sich zum anderen an [Mas94], einem auf ERNEST basierenden Dialogsystem zur Zugauskunft.

An das System können frei formulierte Anfragen zu einer Zugverbindung gestellt werden. Abbildung 6.4 zeigt einen kurzen Beispieldialog, der mit dem realisierten System geführt werden kann. Nach einer Begrüßung durch das System bringt der Benutzer seinen Reisewunsch vor.

System 1: „Guten Tag! Hier ist die automatische Bahnauskunft! Womit kann ich Ihnen dienen?“

Benutzer 1: „Ja, guten Tag! Ich möchte gern nach Hamburg fahren.“

System 2: „Wann wollen Sie fahren ?“

Benutzer 2: „Morgen am Nachmittag — so um drei.“

System 3: „Sie wollen von Bielefeld nach Hamburg fahren. Sie möchten am 13.4. um 15.00 Uhr reisen. “

Benutzer 3: „Ja genau.“

System 4: „Okay, Sie können fahren . . . “

Abbildung 6.4: Beispieldialog zur Zugauskunft

Das System stellt fest, daß noch der Zeitpunkt der Reise fehlt und fragt daher explizit nach (System 2). Bei der Analyse der nachfolgenden zweiten Benutzeräußerung wird die Zeitangabe „so um drei“ zu fünfzehn Uhr aufgelöst, weil die Reise ja nachmittags stattfinden soll. Nun sind alle benötigten Informationen beisammen, und das System faßt die Anfrage zusammen (System 3). Dabei wird im Beispieldialog Bielefeld als Abfahrtsort angenommen, da der Benutzer keine anderen Angaben gemacht hat. Nach einer Bestätigung des Benutzers (Benutzer 3) erfolgt die Datenbankanfrage und das System teilt die Reisezeit mit (System 4).

Ein Überblick über die Architektur des gesamten Systems ist in Abbildung 6.5 zu sehen. Die sprachliche Eingabe wird von dem Spracherkenner entgegengenommen, der wie der Erkenner im Konstruktionsszenario neben den akustischen Modellen und den statistischen Sprachmodellen eine domänenspezifische Segmentdefinition benutzt. Seine strukturierten Resultate werden in der Sprachverstehenskomponente interpretiert, die dazu auf ein Lexikon zugreift. Dieses Modul kommuniziert auch mit der Datenbank. Die sprachlichen Rückmeldungen an den Benutzer werden ebenfalls von der Verstehenskomponente formuliert und vom Synthese–Modul synthetisiert.

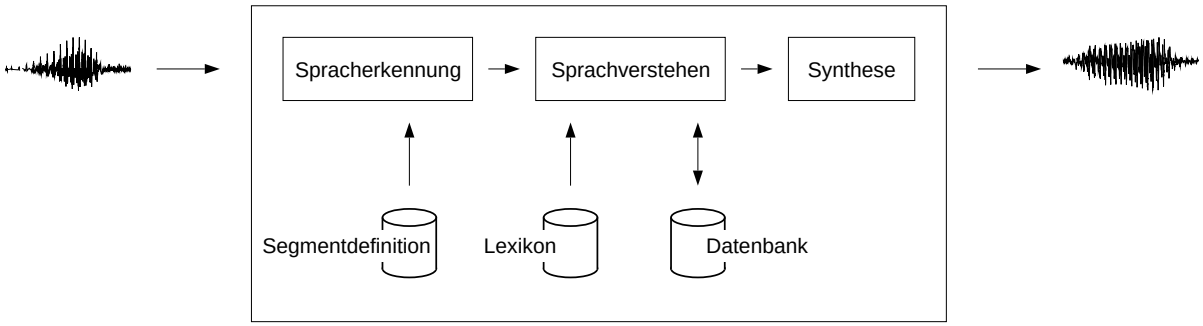

Abbildung 6.5: Architektur des Systems zur Zugauskunft

Die Segmentdefintion enthält Segmente, die sich thematisch in die folgenden Bereiche gliedern lassen:

Ortsangaben: Dabei wird im wesentlichen unterschieden zwischen Abfahrtsorten, Ankunftsorten und solchen Ortsangaben, die einen Umsteigebahnhof benennen.

Zeitangaben: Es gibt Segmente zur Abdeckung von Formulierungen, die einen Tag spezifizieren (wie „am ersten Mai“, „montags“, „Heiligabend“, „am kommenden Wochenende“, „heute“) sowie zur Uhrzeit („acht Uhr dreißig“, „um viertel nach zwei“, „zehn vor zwölf“), zu Tageszeiten („morgens“) und zu relativen Zeitangaben („in zwei Stunden“).

Verben: Für das System ist es wichtig zu wissen, ob gegebenenfalls eine Ankunfts– oder Abfahrtszeit benannt wurde. Dies läßt sich über die Verbsemantik schließen, wie folgende Beispieläußerungen zeigen:

„Ich möchte gegen zehn Uhr in Hamburg ankommen.“
„Ich möchte gegen zehn Uhr in Hamburg losfahren.“

Im ersten Beispiel denotiert „gegen zehn Uhr“ eine Ankunftszeit, im zweiten die Abfahrtszeit. Daher wird in der Segmentdefinition zwischen Verben, die als Argument eine Ankunftszeit und solchen, die eine Abfahrtszeit erwarten, unterschieden.

Angaben zum Zugwunsch und zur Art und Weise der Reise: Diese Segmente decken Äußerungsteile ab, die besondere Wünsche des Benutzer enthalten, wie beispielsweise „mit einem Intercity“ oder „ohne umsteigen“.

Damit sind die wesentlichen Formulierungen, deren propositionaler Gehalt in eine Datenbankanfrage eingetragen werden können, modelliert. Die Modellspezifikationen wurden aus [Mas94] abgeleitet, die sich ihrerseits auf das FACID–Korpus [Hit88] bezieht.

Die ERNEST++–Wissenbasis der Sprachverstehenskomponente besteht aus drei Ebenen. Die *Segmentebene* enthält den Segmenten entsprechende Konzeptdefinitionen, die auf der *Interpretationsebene* zu Konstituenten zusammengefaßt und interpretiert werden. Auf der *Dialogebene* erfolgt schließlich die Auswertung der einzelnen Äußerungen im Kontext des Dialogs, die Auswahl der nächsten Systemausgabe sowie die Datenbankanfrage.

Die Verarbeitungsstrategie ist analog zur Sprachverstehenskomponente im Konstruktionsszenario organisiert: ein Segment wird instantiiert und zu einem speziellen Segment spezialisiert. Dieses wird sodann an die Dialoginstanz datengetrieben gebunden, bevor das nachfolgende Segment eingelesen wird. Dieser Zyklus wiederholt sich bis zum Ende einer Äußerung. Dananch erfolgt die Instantiierung der Systemausgabe, und die Verarbeitung der nächsten Äußerung geschieht in der gleichen Weise, bis schließlich der Informationswunsch des Benutzers erfüllt ist und der Dialog beendet werden kann.

Eine umfassende Evaluierung des Systems steht noch aus. Die ersten Ergebnisse wurden auf einer Stichprobe erzielt, die aus 115 spontansprachlichen Erstanfragen von 82 verschiedenen Sprechern besteht. Dabei wurde das beste Spracherkennungsergebnis mit 25,4 Prozent Wortfehlerrate bei der Kombination eines Bigramms mit der Segmentdefinition erreicht. Auf der Interpretationsebene konnten 81,3 Prozent der Segmente korrekt verarbeitet werden. Die Verarbeitungszeit liegt im Millisekundenbereich. Die bisherigen Resultate haben somit eine ähnliche Qualität wie die im Konstruktionsszenario erreichten Ergebnisse. Das in dieser Arbeit vorgestellte Verfahren zur Verarbeitung gesprochener Sprache läßt sich also auch in anderen Anwendungsbereichen erfolgreich einsetzen.

6.4 Resümee

In diesem Kapitel erfolgte die Evaluierung der realisierten Sprachverstehenskomponente unter drei Aspekten.

Zunächst wurde das System unter dem Blickwinkel Hauptschnittstelle in der Mensch–Maschine–Kommunikation zu sein betrachtet. In einem repräsentativen Dialog zeigte sich, daß die Sprachverstehenskomponente in Kooperation mit den anderen Modulen fähig ist, den Dialog mit einem Instrukteur angemessen zu gestalten. Dabei kann es mit spontansprachlichen Phänomenen ebenso umgehen wie mit fehlerhaften Ergebnissen des Spracherkenners. Die Systemantworten auf die Instruktionen sind stets so gewählt, daß der Instrukteur implizit erfahren kann, was der jeweilige Wissensstand des Systems ist und was das System von ihm erwartet.

Die Evaluierung an spontansprachlichen Daten von vielen Sprechern diente der Überprüfung der Robustheit des Systems. Die Abdeckungsrate von bis zu 93 Prozent belegt die Angemessenheit der Modellierung der Segmentdefinition. Sie unterstüzt die Spracherkennung im Zusammenspiel

mit einem Bigramm. Die Wortfehlerrate konnte daher auf bis zu 20,5 Prozent gedrückt werden. Die semantisch–pragmatische Interpretation erwies sich als sehr robust, denn auf den benutzten Teststichproben wurden über 70 Prozent der konstruktionsrelevanten Segmente vollständig oder partiell korrekt analysiert und auf dieser Grundlage in gleicher Weise bis zu 97 Prozent der Anweisungen. Die Verarbeitungszeit für eine Äußerung lag im Millisekundenbereich. Schließlich wurde an dem Material auch das Dialogverhalten des gesamten Konstruktionssystems untersucht. Nur in weniger als fünf Prozent der Anweisungen hat das System nicht angemessen reagiert.

Der letzte Abschnitt betrachtete die Realisierung der Verstehenskomponente unter dem Aspekt der Allgemeingültigkeit des gewählten Verfahrens. Mit der Vorstellung einer Sprachverstehenskomponente für die Zugauskunftsdomäne, die nach dem selben Verfahren arbeitet, wurde gezeigt, daß der Ansatz keine Speziellösung für ein Spezialproblem ist sondern allgemein in der Verarbeitung gesprochener Sprache eingesetzt werden kann.

Insgesamt zeigte sich also, daß die Sprachverstehenskomponente den formulierten Ansprüchen gerecht wird. Die Evaluierungsergebnisse belegen die Realisierung eines kompetenten, robusten und effizienten Systems zur Interpretation gesprochener Sprache im Konstruktionsszenario. Das dabei entwickelte Verfahren ist auch in anderen Domänen einsetzbar.

Kapitel 7

Zusammenfassung

Hübsch als es währte —
und nun ist's vorüber.
Bertold Brecht

Thema der vorliegenden Arbeit ist die Konzeption und Realisierung einer Sprachverstehenskomponente in einem Konstruktionsszenario. Die realisierte Sprachverstehenskomponente ist eingebettet in ein komplexes System, das zusätzlich aus einer Bildverarbeitungs– und einer Robotikkomponente besteht. Mit dem System ist ein menschlicher Instrukteur in der Lage, einen Roboter in einem Konstruktionsszenario spontansprachlich anzuweisen.

Die Aufgaben der Sprachverstehenskomponente als Hauptschnittstelle in der Mensch–Maschine–Kommunikation bestehen in der Interpretation der Benennungen der Konstruktionsgegenstände in der jeweiligen Szene sowie der Ableitung von Handlungsaufforderungen. Dabei ist jeweils der Dialogkontext zu berücksichtigen. Außerdem sind Interventionen des Instrukteurs, die auf eine sofortige Unterbrechung des Konstruktionsprozesses zielen, zu verarbeiten. Schließlich muß in Kooperation mit den anderen Komponenten des Gesamtsystems eine geeignete Systemreaktion bestimmt, deren Ausführung angestoßen und dem menschlichen Gegenüber erläutert werden.

Die Grundlagen für die wichtigsten Modelle, das heißt für die Objektbenennungen und die Verarbeitung von Verben, werden mit Hilfe eines Korpus aus simulierter Mensch–Maschine–Kommunikation ermittelt. Darüber hinaus lassen sich aus dem Korpus allerdings keine engen Restriktionen für die Satzkonstruktion und kein aussagekräftiges Dialogmodell gewinnen.

Die Architektur der Sprachverarbeitung im gesamten Konstruktionssystem ist gekennzeichnet von einer flachen Hierarchie und einer vertikalen Organisation des Wissens — in die signalnahen Ebenen fließt auch Wissen der abstrakteren Ebenen ein und die einzelnen Ebenen sind sehr gut aufeinander abgestimmt.

Der in dem Gesamtsystem verwendete Spracherkenner erlaubt neben statistischen Sprachmodellen die Verwendung einer deklarativen Grammatik, der sogenannten Segmentdefinition, zur Unterstützung der Analyse des Sprachsignals. Die ursprüngliche Motivation für dieses Verfahren besteht darin, Mängel von statistischen Sprachmodellen gezielt auszugleichen und das Spracherkennungsergebnis auf diese Weise zu verbessern. Aus der Sicht des sprachverstehenden Prozesses ergeben sich dadurch neue Möglichkeiten: zum einen kann ein Großteil der typischerweise im Verstehensmodul erledigten syntaktischen Analyse bereits vom Spracherkenner bewältigt werden. Zum anderen kann durch die Festlegung von geeigneten akzeptierten Nichtterminalsymbolen, den Segmenten, domänenspezifisches Wissen in den spracherkennenden Prozeß einfließen. In der für das Konstruktionssystem festgelegten Segmentdefinition sind folglich die Erkenntnisse aus der Korpusuntersuchung zu den Formulierungen der Instrukteure berücksichtigt. Im Ergebnis dessen liefert der Spracherkenner nicht eine Folge von Wörtern, sondern ein strukturiertes Resultat ab, das die Grundlage für die Interpretation bildet. Dieses Verfahren bedeutet eine neuartige Verschränkung von Spracherkennung und Sprachverstehen, die üblicherweise strikt voneinander getrennt sind.

Die Interpretation der einzelnen Anweisungen und die Herstellung des Dialogkontextes geschieht in einem ERNEST^{++}–Netz. Es gliedert sich in drei Ebenen. Die Segmentebene bildet die Schnittstelle zum Spracherkenner, in der jedes Segment der Segmentdefinition in einer Konzeptdefinition repräsentiert ist. Ihnen zugeordnet sind die Konzeptdefinitionen der Beschreibungsebene, in der die konstruktionsrelevanten Segmente zu sinnvollen Konstituenten kombiniert und interpretiert werden. Diese werden in einer Konzeptdefinition zur Repräsentation einer Äußerung zusammengefaßt. Auf der Dialogebene des semantischen Netzes werden die einzelnen Äußerungen im Kontext des Dialogs ausgewertet. Es werden Objektbenennungen disambiguiert sowie Handlungsstränge erkannt und auf Vollständigkeit überprüft. Die Auswahl der Systemreaktion gründet sich auf einem wenig restriktiven Ablaufmodell und hängt in starkem Maße auch von den Ergebnissen der Bildverarbeitungs– und Robotikmodule ab. Jede Systemreaktion wird dem Instrukteur angemessen erläutert. Dabei wird ihm jeweils implizit mitgeteilt, in welcher Weise seine Anweisung verstanden wurde. Der modulare Aufbau der Sprachverstehenskomponente ermöglicht eine einfache Erweiterung der Kompetenz des Systems.

Die Analysestrategie im semantischen Netz ist rein datengetrieben organisiert. Die vom Spracherkenner gelieferten Segmente werden inkrementell eingelesen und weiterverarbeitet. Daher liegen schon vor dem Abschluß der Äußerung auf der obersten Ebene der Sprachverstehenskomponente Interpretationsergebnisse vor. Auf der Grundlage der Anweisungsinterpretation werden entsprechende Referenten in der Szene gesucht, eine Systemreaktion bestimmt und erläutert. Sodann beginnt der Zyklus von neuem mit dem Entgegennehmen der nächsten Anweisung. Sowohl die Wissensbasis als auch die Analysestrategie sind so realisiert, daß Ergebnisse von anderen sprachverarbeitenden Modulen unmittelbar integriert werden können. Mit der rein da-

tengetriebene Interpretation wird eine neue Verarbeitungsstrategie in auf ERNEST beziehungsweise ERNE$^{++}_{S}$T beruhenden sprachverstehenden Systemen realisiert, die sich vor allem durch eine große Effizienz und Robustheit auszeichnet. Durch eine Modifikation der Strategie können auch verschiedene Lesarten einer Anweisung effizient verarbeitet werden.

Die Evaluation des Systems erfolgt unter mehreren Gesichtspunkten. Zunächst wird die Kompetenz der Sprachverstehenskomponente anhand eines repräsentativen Dialogs dargelegt, der auch demonstriert, wie mit unvollständigen Anweisungen und Fehlern der Spracherkennung umgegangen wird.

Die Evaluierung anhand spontansprachlichen Datenmaterials dient vor allem der Überprüfung der Robustheit, denn es enthält Sprachdaten von vielen verschiedenen Sprechern. Die Validität der Segmentdefinition zeigt sich vor allem an den Abdeckungsraten. Sie sind dann besonders hoch, wenn die Aufnahmebedingungen der Teststichprobe dem Szenario entsprechen, für das die Sprachverstehenskomponente entwickelt wurde. Die Wirksamkeit der Segmentdefinition bei der Spracherkennung wird daran sichtbar, daß die Wortfehlerraten sinken, wenn sie zusätzlich zu einem Bigramm eingesetzt wird. Die Interpretationsleistung wird auf verschiedenen Stufen bewertet. Über 70 Prozent der Segmente in den Teststichproben werden vollständig oder partiell korrekt interpretiert. Bei der Interpretation ganzer Äußerungen erhöht sich diese Zahl auf bis zu 97 Prozent, wobei die Verarbeitungszeit für eine Äußerung im Millisekundenbereich liegt. Zur Evaluierung des Dialogverhaltens werden die Anweisungen, zu denen auch das Bildmaterial vorliegt, dem gesamten Konstruktionssystem eingespeist und als initiale Anweisungen aufgefaßt. In über 95 Prozent der Anweisungen wird eine approximativ korrekte Systemleistung erbracht. Die Sprachverstehenskomponente bewährt sich also auch bei nicht geübten Instrukteuren.

Daß das realisierte Verfahren zur Verarbeitung gesprochener Sprache nicht eine spezielle Lösung für ein spezielles Problem ist, zeigt die Übertragbarkeit des Ansatzes auf eine andere Domäne. Ein System zur Zugauskunft wird vorgestellt, welches nach der gleichen Methode entwickelt wurde und strukturell wie algorithmisch der Sprachverstehenskomponente für die Konstruktionsdomäne entspricht.

Insgesamt wird in der vorliegenden Arbeit also eine Sprachverstehenskomponente dargelegt, die mit einem neuartigen Verfahren in Kooperation mit anderen Modulen spontane Konstruktionsanweisungen robust und effizient verarbeitet.

Literaturverzeichnis

[Abe97] A. Abella, M. K. Brown, B. Buntschuh: *Development Principles for Dialog-Based Interfaces*, in E. Maier, M. Mast, S. LuperFoy (Hrsg.): *Dialogue Processing in Spoken Language Systems*, Bd. 1236 von *Lecture Notes in Artifical Intelligence*, Springer, Berlin, 1997, S. 141 – 155.

[Aho86] A. V. Aho, R. Sethi, J. D. Ullman: *Compilers: Principles, Techniques, and Tools*, Addison-Wesley, Reading, Massachusetts, 1986.

[All95a] J. F. Allen, G. Fergusson, B. W. Miller, E. K. Ringger: *Spoken Dialogue and Interactive Planning*, in *Proceedings ARPA Spoken Language Systems Technology Workshop (SLST)*, Austin, Texas, 1995.

[All95b] J. F. Allen, L. K. Schubert, G. Ferguson, P. Heeman, C. H. Hwang, T. Kato, M. Light, N. G. Martin, B. W. Miller, M. Poesio, D. R. Traum: *The TRAINS Project: A case study in building a conversational planning agent*, *Journal of Experimental and Theoretical AI*, Bd. 7, Nr. 6, 1995, S. 7 – 48.

[All96] J. F. Allen, B. W. Miller, E. K. Ringger, T. Sikorski: *A Robust System for Natural Spoken Dialogue*, in *Proceedings of the 1996 Annual Meeting of the Association for Computational Linguistics (ACL'96)*, 1996, S. 62 – 70.

[Amt96] J. Amtrup, J. Benra: *Communication in large distributed AI Systems for Natural Language Processing*, in *Proceedings International Conference on Computational Linguistics (COLING'96)*, Copenhagen, 1996, S. 35 – 40.

[Ara97] M. Araki, S. Doshita: *Automatic Evaluation Environment for Spoken Dialogue Systems*, in E. Maier, M. Mast, S. LuperFoy (Hrsg.): *Dialogue Processing in Spoken Language Systems*, Bd. 1236 von *Lecture Notes in Artifical Intelligence*, Springer, Berlin, 1997, S. 183 – 194.

[Aus62] J. L. Austin (Hrsg.): *How to Do Things with Words*, Oxford University Press, Oxford, 1962.

[Bah83] L. Bahl, F. Jelinek, R. Mercer: *A Maximum Likelihood Approach to Continuous Speech Recognition*, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Bd. 5, Nr. 2, 1983, S. 179 – 190.

[Bar97] J. Barnett, M. Singh: *Designing a Portabel Spoken Dialogue System*, in E. Maier, M. Mast, S. LuperFoy (Hrsg.): *Dialogue Processing in Spoken Language Systems*, Bd. 1236 von *Lecture Notes in Artifical Intelligence*, Springer, Berlin, 1997, S. 156 – 170.

[Bau00] L. F. Baum: *The Wonderful Wizard of Oz*, G.M.Hill , Chicago, 1900.

[Bau98] C. Bauckhage, F. Kummert, G. Sagerer: *Modeling and Recognition of Assembled Objects*, in *Proceedings 24th Annual Conference of the IEEE Electronics Society (IECON'98)*, Aachen, 1998, S. 2051 – 2056.

[Bos96] J. Bos, M. Egg, M. Schiehlen: *Definition of the Abstract Semantic Classes for the Verbmobil Forschungsprototyp 1.0*, Verbmobil – Report 165, 1996.

[BP95] H. Brandt-Pook, J. Bückner: *Objektorientierte Realisierung der Kontrollstrategie einer Wissensrepräsentationssprache*, Diplomarbeit, Universität Bielefeld, Technische Fakultät, 1995.

[BP96] H. Brandt-Pook, G. A. Fink, B. Hildebrandt, F. Kummert, G. Sagerer: *A Robust Dialogue System for Making an Appointment*, in *Proceedings International Conference on Spoken Language Processing (ICSLP'96)*, Bd. 2, Philadelphia, 1996, S. 693 – 696.

[BP99] H. Brandt-Pook, G. A. Fink, S. Wachsmuth, G. Sagerer: *Integrated Recognition and Interpretaion of Speech for a Construction Task Domain*, in *Proceedings 8th International Conference on Human-Computer Interaction (HCI'99)*, München, 1999, erscheint.

[Bri95a] C. Brindöpke, J. Häger, M. Johanntokrax, A. Pahde, M. Schwalbe, B. Wrede: *„Darf ich dich Marvin nennen?" — Instruktionsdialoge in einem Wizard-of-Oz-Szenario: Szenario-Design und Auswertung*, Report 16/95, Sonderforschungsbereich 360 „Situierte Künstliche Kommunikatoren", Universität Bielefeld, 1995.

[Bri95b] C. Brindöpke, M. Johanntokrax, A. Pahde, B. Wrede: *„Darf ich dich Marvin nennen?" — Instruktionsdialoge in einem Wizard-of-Oz-Szenario: Materialband*, Report 7/95, Sonderforschungsbereich 360 „Situierte Künstliche Kommunikatoren", Universität Bielefeld, 1995.

[Bri98] C. Brindöpke, G. Fink, F. Kummert, G. Sagerer: *An HMM-based recognition system for perceptive relevant pitch movements of spontaneous German speech*, in *Proceedings International Conference on Spoken Language Processing (ICSLP'98)*, Bd. 7, Sydney, 1998, S. 2895 – 2898.

[Bub96] T. Bub, J. Schwinn: *VERBMOBIL: The Evolution of a Complex Large Speech – to – Speech Translation System*, in *Proceedings International Conference on Spoken Language Processing (ICSLP'96)*, Bd. 4, Philadelphia, 1996, S. 2371 – 2374.

[Bub97] T. Bub, W. Wahlster, A. Waibel: *VERBMOBIL: The Combination of Deep and Shallow Processing for Spontaneous Speech Translation*, in *Proceedings International Conference on Acoustics, Speech and Signal Processing (ICASSP'97)*, Bd. 1, München, 1997, S. 71 – 74.

[Cha85] E. Charniak, D. McDermott: *Introduction to Artificial Intelligence*, Addison-Wesley, Reading, Massachusetts, 1985.

[Dav52] K. Davis, R. Biddulph, S. Balashek: *Automatic Recognition of Spoken Digits*, *JASA*, Bd. 24, 1952, S. 637 – 642.

[Dro84] G. Drosdowksi (Hrsg.): *Duden: Grammatik der deutschen Gegenwartssprache*, Bd. 4 von *Der Duden in zehn Bänden*, Bibliographisches Institut, Mannheim, Wien, Zürich, 1984.

[Eik98] H.-J. Eikmeyer: *Persönliche Kommunikation*, 1998.

[Fer96] G. Ferguson, J. F. Allen, B. W. Miller, E. K. Ringger: *The Design and Implementation of the TRAINS-96 System: A Prototype Mixed – Initiative Planning Assistant*, TRAINS Technical Note 96-5, University of Rochester, CS Department, 1996.

[Fil68] C. Fillmore: *A Case for Case*, in E. Bach, R. T. Harms (Hrsg.): *Universals in Linguistic Theory*, Holt, Rinehart and Winston, New York, 1968, S. 1 – 88.

[Fin94] T. Finin, R. Fritzson, D. McKay, R. McEntire: *KQML as an Agent Communication Language*, in *Proceedings International Conference on Information and Knowledge Management (CIKM'94)*, Gaithersburg, Maryland, 1994.

[Fin95a] G. A. Fink: *Integration von Spracherkennung und Sprachverstehen*, Bd. 103 von *Dissertationen zur Künstlichen Intelligenz*, infix, Sankt Augustin, 1995.

[Fin95b] G. A. Fink, N. Jungclaus, H. Ritter, G. Sagerer: *A Communication Framework for Heterogeneous Distributed Pattern Analysis*, in *International Conference on Algorithms and Architectures for Parallel Processing*, Brisbane, 1995, S. 881 – 890.

[Fin98] G. A. Fink, C. Schillo, F. Kummert, G. Sagerer: *Incremental Speech Recognition for Multimodal Interfaces*, in *IECON*, Aachen, 1998, S. 2012 – 2017.

[Fis95] K. Fischer, M. Johanntokrax: *Ein linguistisches Merkmalsmodell für die Lexikalisierung von diskurssteuernden Partikeln*, Report 18/95, Sonderforschungsbereich 360 „Situierte Künstliche Kommunikatoren", Universität Bielefeld, 1995.

[Fis98] K. Fischer, H. Brandt-Pook: *Automatic Disambiguation of Discourse Particles*, in M. Stede, L. Wanner, E. Hovy (Hrsg.): *Workshop on Discourse Relations and Discourse Markers at COLING-ACL98*, Montral, Quebec, Canada, 1998, S. 107 – 113.

[Fri99] J. Fritsch, H. Brandt-Pook, G. Sagerer: *Combining planning and dialog for cooperative assembly construction*, in *Workshop "Planning and Scheduling meet Real-Time Monitoring in a Dynamic and Uncertain World" at IJCAI-99*, Stockholm, 1999, erscheint.

[Fuj86] H. Fujisaki, K. Hirose, H. Udagawa, N. Kanedera: *A New Approach to Continuous Speech Recognition Based on Considerations on Human Process of Speech Perception*, in *Proceedings International Conference on Acoustics, Speech and Signal Processing (ICASSP'86)*, Tokyo, 1986.

[Han89] S. Handel: *Listening: An Introduction to the Perception of Auditory Events*, MIT Press, Cambridge, 1989.

[Han98] G. Hanrieder, P. Heisterkamp, T. Brey: *Fly with the EAGLES: Evaluation of the "ACCeSS" Spoken Language Dialogue System*, in *Proceedings International Conference on Spoken Language Processing (ICSLP'98)*, Bd. 2, Sydney, 1998, S. 503 – 506.

[Hei96] G. Heidemann, F. Kummert, H. Ritter, G. Sagerer: *A Hybrid Object Recognition Architecture*, in C. von der Malsburg, W. von Seelen, J. Vorbrüggen, B. Sendhoff (Hrsg.): *Artificial Neural Networks — ICANN 96*, Springer-Verlag, Berlin, 1996, S. 305 – 310.

[Her76] T. Herrmann, W. Deutsch: *Psychologie der Objektbenennung*, Huber, Bern, 1976.

[Her94] T. Herrmann, J. Grabowski: *Sprechen — Psychologie der Sprachproduktion*, Spektrum Akademischer Verlag, Heidelberg, Berlin, Oxford, 1994.

[Hey88] G. Heyer, J. Krems, G. Görz (Hrsg.): *Wissensarten und ihre Darstellung*, Bd. 169 von *Informatik-Fachberichte*, Springer Verlag, Berlin, Heidelberg, New York, Tokyo, 1988.

[Hil95] B. Hildebrandt: *Struktur und Bedeutung temporaler Konstituenten in einem sprachverstehenden Dialogsystem*, Bd. 104 von *Dissertationen zur Künstlichen Intelligenz*, infix, Sankt Augustin, 1995.

[Hil97] B. Hildebrandt, G. Rickheit: *Verarbeitung von Präpositionalphrasen in der Combinatory Categorial Grammar*, Report 6/97, Sonderforschungsbereich 360 „Situierte Künstliche Kommunikatoren“, Universität Bielefeld, 1997.

[Hit88] L. Hitzenberger, R. Ulbrand, H. Kritzenberger, P. Wenzel: *FACID: Fachsprachlicher Corpus informationsabfragender Dialoge*, Technischer Bericht, Linguistische Informationswissenschaft, Universität Regensburg, 1988.

[Hua90] X. Huang, Y. Ariki, M. Jack: *Hidden Markov Models for Speech Recognition*, Edinburgh University Press, Edinburgh, 1990.

[Hua93] X. Huang, F. Alleva, H.-W. Hon, M.-Y. Hwang, K.-F. Lee, R. Rosenfeld: *The SPHINX-II Speech Recognition System: An Overview*, *Computer Speech & Language*, Bd. 2, 1993, S. 127 – 148.

[Jac83] R. Jackendorff: *Semantics and Cognition*, Bd. 8 von *Current studies in linguistic series*, MIT Press, Cambridge, Mass., 1983.

[Jac89a] R. Jackendorff: *Speaking: from intention to articulation* , MIT Press, Cambridge, Mass., 1989.

[Jac89b] R. Jackendorff: *Speaking: from intention to articulation* , Bd. 18 von *Current studies in linguistic series*, MIT Press, Cambridge, Mass., 1989.

[Jac91] E. Jackson, D. Appelt, J. Bear, , R. Moore, A. Podlozny: *A Template Matcher for Robust NL Interpretation*, in *Proceedings DARPA Speech and Natural Language Workshop*, Pacific Grove, California, 1991, S. 190–194.

[Jun97] B. Jung: *Wissensverarbeitung für Montageaufgaben in virtuellen und realen Umgebungen*, Bd. 157 von *Dissertationen zur Künstlichen Intelligenz*, infix, Sankt Augustin, 1997.

[Jun98] N. Jungclaus: *Integration verteilter Systeme zur Mensch – Maschine – Kommunikation*, Dissertation, Universität Bielefeld, Technische Fakultät, 1998.

[Kat87] S. Katz: *Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer*, *IEEE Trans. on Acoustics, Speech, and Signal Processing*, Bd. 35, Nr. 3, 1987, S. 400 – 401.

[Kaw96] T. Kawahara, N. Kitaoka, S. Doshita: *Concept – Based Phrase Spotting Approach for Spontaneous Speech Understanding*, in *Proceedings International Conference on Acoustics, Speech and Signal Processing (ICASSP'96)*, Bd. 1, Atlanta, 1996, S. 291 – 294.

[Kaw98] T. Kawahara, C.-H. Lee, B.-H. Juang: *Flexible Speech Understanding Based on Combined Key-Phrase Detection and Verification*, *IEEE Transactions on Speech and Audio Processing*, Bd. 6, Nr. 6, 1998, S. 558 – 568.

[Kes96] K. Kessler, I. Duwe, H. Strohner: *Sprachliche Objektidentifikation in ambigen Situationen: Empirische Befunde*, Report 1/96, Sonderforschungsbereich 360 „Situierte Künstliche Kommunikatoren", Universität Bielefeld, 1996.

[Koh94] K. Kohler, G. Lex, M. Pätzhold, M. Scheffers, A. Simpson, W. Thon: *Handbuch zur Datenaufnahme und Transliterationin in TP14 von VERBMOBIL – 3.0*, Verbmobil – Technisches Dokument 11, 1994.

[Kra92] J. Krause: *Natürlichsprachliche Mensch-Computer-Interaktion als technisierte Kommunikation: Die computer talk-Hypothese*, in J. Krause, L. Hitzenberger (Hrsg.): *Computer Talk*, Georg Olms Verlag , Tübingen, Basel, 1992.

[Kro98] S. Kronenberg, F. Kummert: *Syntax Coordination: Interaction of Discourse and Extrapositions*, in *Proceedings International Conference on Spoken Language Processing (ICSLP'98)*, Bd. 5, Sydney, 1998, S. 2071 – 2074.

[Kuh99] U. Kuhlmann: *Wie bitte? Vier Diktiersysteme im Vergleich , ct — Magazin für Computertechnik*, 3/1999, S. 124 – 132.

[Kum92] F. Kummert: *Flexible Steuerung eines sprachverstehenden Systems mit homogener Wissensbasis*, Bd. 12 von *Dissertationen zur Künstlichen Intelligenz*, infix, Sankt Augustin, 1992.

[Kum93] F. Kummert, H. Niemann, R. Prechtel, G. Sagerer: *Control and Explanation in a Signal Understanding Environment*, *Signal Processing, special issue on 'Intelligent Systems for Signal and Image Understanding'*, Bd. 32, 1993, S. 111 – 145.

[Kum98a] F. Kummert, G. Fink, G. Sagerer, E. Braun: *Hybrid Object Recognition in Image Sequences*, in *Proceedings 14th International Conference on Pattern Recognitioni (ICPR'98)*, Bd. II, Brisbane, 1998, S. 1165 – 1170.

[Kum98b] F. Kummert: *Interpretation von Bild- und Sprachsignalen — Ein hybrider Ansatz*, Shaker, Aachen, 1998.

[Lag95] T. Lager: *A Logical Approach to Computational Corpus Linguistics*, Ph.D. Thesis, Göteborg University, Department of Linguistics, 1995.

[Lee89] K.-F. Lee, H.-W. Hon, M.-Y. Hwang, S. Mahajan, R. Reddy: *The SPHINX Speech Recognition System*, in *Proceedings International Conference on Acoustics, Speech and Signal Processing (ICASSP'89)*, 1989, S. 445 – 448.

[Lev83] W. J. Levelt: *Monitoring and Self-Repair in Speech*, *Cognition*, Bd. 14, 1983, S. 41 – 104.

[Lev89] W. J. Levelt: *Speaking: from intention to articulation* , Bd. XIV von *Series in natural-language processing*, MIT Press, Cambridge, Mass., 1989.

[Lob98] H. Lobin: *Handlungsanweisungen — Sprachliche Spezifikation teilautonomer Aktivität*, Deutscher Universitätsverlag, Wiesbaden, 1998.

[Löm98] F. Lömker: *Realisierung und Evaluation einer verteilten Kontrollstrategie für die semantische Netzwerksprache ERNEST*, Diplomarbeit, Universität Bielefeld, Technische Fakultät, 1998.

[Löm99] F. Lömker, H. Brandt-Pook, F. Kummert, G. Sagerer: *A Distributed Semantic Network Language for Signal Understanding*, in M. Mohammadian (Hrsg.): *Proceedings International Conference on Computational Intelligence for Modelling, Control & Automation: Neural Networks & Advanced Control Strategies (CIMCA' 99)*, IOS Press, Amsterdam, Wien, 1999, S. 311 – 318.

[MA92] R. Mangold-Allwinn, C. von Stutterheim, S. Baratelli, U. Kohlmann, G. Koebling: *Objektbenennung im Diskurs: Eine interdisziplinäre Untersuchung*, *Kognitionswissenschaft*, Bd. 3, 1992, S. 1 – 11.

[Mai97] E. Maier, M. Mast, S. LuperFoy: *Overview*, in E. Maier, M. Mast, S. LuperFoy (Hrsg.): *Dialogue Processing in Spoken Language Systems*, Bd. 1236 von *Lecture Notes in Artifical Intelligence*, Springer, Berlin, 1997, S. 1 – 13.

[Mal98] U. Malaske: *Sprechen statt Schreiben* , *ct — Magazin für Computertechnik*, 5/1998, S. 110 – 119.

[Mas93] M. Mast: *Ein Dialogmodul für ein Spracherkennungs – und Dialogsystem*, Bd. 50 von *Dissertationen zur Künstlichen Intelligenz*, infix, Sankt Augustin, 1993.

[Mas94] M. Mast, F. Kummert, U. Ehrlich, G. A. Fink, T. Kuhn, H. Niemann, G. Sagerer: *A Speech Understanding and Dialog System with a Homogeneous Linguistic Knowledge Base*, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Bd. 16, 1994, S. 179 – 193.

[Moo98] R. K. Moore: *Understanding Speech Understanding*, in *Proceedings International Conference on Acoustics, Speech and Signal Processing (ICASSP'98)*, Bd. 2, Seattle, Washington, 1998, S. 1049 – 1052.

[Nil71] N. J. Nilsson (Hrsg.): *Problem Solving Methods in Artificial Intelligence*, McGraw-Hill, New York, 1971.

[Nöt89] E. Nöth: *Prosodische Information in der automatischen Spracherkennung — Berechnung und Anwendung*, Dissertation, Lehrstuhl für Informatik 5 (Mustererkennung), Universität Erlangen-Nürnberg, 1989.

[Pea88] J. Pearl: *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann Publishers, San Mateo, CA, 1988.

[Pet98] K. Peters: *Natürlichsprachliche Kommunikation mit handelnden Systemen*, Dissertation, Universität Bielefeld, Technische Fakultät, 1998.

[Qui68] M. R. Quilian: *Semantic Memory*, in M. Minsky (Hrsg.): *Semantic Information Processing*, MIT Press, Cambridge, Mass., 1968, S. 227 – 270.

[Ric93] G. Rickheit, H. Strohner: *Grundlagen der kognitiven Sprachverarbeitung*, Francke, Tübingen, Basel, 1993.

[Ric96] G. Rickheit, I. Wachsmuth: *Collaborative Research Centre "Situated Artificial Communicators" at the University of Bielefeld, Germany*, *Artificial Intelligence Review*, Bd. 10 (3-4), 1996, S. 165 – 170.

[Rin96] E. K. Ringer, J. F. Allen: *A Fertility Channel Model for Post-Correction of Continuous Speech Recognition*, in *Proceedings International Conference on Spoken Language Processing (ICSLP'96)*, Philadelphia, 1996, S. 897 – 900.

[Sag85] G. Sagerer: *Darstellung und Nutzung von Expertenwissen für ein Bildanalysesystem*, Bd. 104 von *Informatik-Fachberichte*, Springer Verlag, Berlin, Heidelberg, New York, Tokyo, 1985.

[Sag90] G. Sagerer: *Automatisches Verstehen gesprochener Sprache*, Bd. 74 von *Reihe Informatik*, B.I. Wissenschaftsverlag, Mannheim, 1990.

[Sag97] G. Sagerer, H. Niemann: *Semantic Networks for Understanding Scenes*, Plenum Press, New York, 1997.

[Sch87] D. Schiffrin: *Discourse markers*, Nr. 5 in Studies in Interactional Sociolinguistics, Cambridge University Press, 1987.

[Sch94] B. Schmitz, S. Jekat-Rommel: *Eine zyklische Approximation an Sprechhandlungstypen — zur Annotierung von Äußerungen in Dialogen*, Verbmobil – Report 28, 1994.

[Sch95a] B. Schmitz, K. Fischer: *Pragmatisches Bescheibungsinventar für Diskurspartikeln und Routineformeln anhand der Demonstratorwortliste*, Memo 75, Verbmobil, 1995.

[Sch95b] B. Schmitz, J. J. Quantz: *Dialogue Acts in Automatic Dialogue Processing*, in *Proceedings Sixth Conference on Theoretical and Methodological Issues in Machine Translation (TMI'95)*, Leuven, 1995, S. 33 – 47.

[Sch98] U. Schade, H.-J. Eikmeyer: *Modeling the Production of Object Specifications*, in J. Grainger, A. Jacobs (Hrsg.): *Localist Connectionist Approaches to Human Cognition*, Kap. 8, Erlbaum, New Jersey, 1998, S. 257 – 282.

[Sea69] J. R. Searle (Hrsg.): *Speech Acts*, Chambridge University Press, Chambridge (UK), 1969.

[Sei97] F. Seide, A. Kellner: *Towards an Automated Directory Information System*, in *Proceedings European Conference on Speech Communication and Technology (EUROSPEECH'97)*, Rhodos, 1997, S. 1327 – 1330.

[SFB94] *„Wir bauen jetzt ein Flugzeug“: Konstruieren im Dialog*, Arbeitsmaterialien, Sonderforschungsbereich 360 „Situierte Künstliche Kommunikatoren“, Universität Bielefeld, 1994, unveröffentlichtes Manuskript.

[Sim93] A. Simpson, N. M. Fraser: *Black box and glass box evaluation of the SUNDIAL System*, in *Proceedings European Conference on Speech Communication and Technology (EUROSPEECH'93)*, Bd. 2, Berlin, 1993, S. 1423 – 1426.

[Soc97] G. Socher: *Qualitative Scene Descriptions from Images for Integrated Speech and Image Understanding*, Bd. 170 von *Dissertationen zur Künstlichen Intelligenz*, infix, Sankt Augustin, 1997.

[Soc98] G. Socher, G. Sagerer, P. Perona: *Bayesian Reasoning on Qualitative Descriptions from Images and Speech*, in H. Buxton, A. Mukerjee (Hrsg.): *Workshop on Conceptual Description of Images at ICCV-98*, Bombay, India, 1998.

[Sow84] J. F. Sowa: *Conceptual Structures: Information Processing in Mind and Machine*, The Systems Programming Series, Addison-Wesley, Reading, Massachusetts, 1984.

[ST91] E. Schukat-Talamazzini, H. Niemann: *Das ISADORA-System — ein akustisch-phonetisches Netzwerk zur automatischen Spracherkennung*, in *DAGM91: Proceedings 13. DAGM-Symposium*, Springer, Berlin, 1991, S. 251 – 258.

[ST95] E. G. Schukat-Talamazzini: *Automatische Spracherkennung*, Vieweg, Wiesbaden, 1995.

[Ste93] M. J. Steedman: *Categorial Grammar*, *Lingua*, Bd. 90, 1993, S. 221 – 258.

[Ste97] A. J. Stent, J. F. Allen: *TRAINS-96 System Evaluation*, TRAINS Technical Note 97-1, University of Rochester, CS Department, März 1997.

[Von99] U. Von der Nahmer: *Eine hybride Sprachverstehenskomponente*, Diplomarbeit, Universität Bielefeld, Technische Fakultät, 1999, erscheint.

[Wac97] S. Wachsmuth: *Kombination von Grammatiken und statistischen Sprachmodellen zur automatischen Spracherkennung*, Diplomarbeit, Universität Bielefeld, Technische Fakultät, 1997.

[Wac98] S. Wachsmuth, G. A. Fink, G. Sagerer: *Integration of Parsing and Incremental Speech Recognition*, in *Proceedings European Signal Processing Conference (EUSIPCO'98)*, Bd. 1, Rhodes, 1998, S. 371 – 374.

[Wac99] S. Wachsmuth, H. Brandt-Pook, G. Socher, F. Kummert, G. Sagerer: *Multilevel Integration of Vision and Speech Understanding using Bayesian Networks*, in H. I. Christensen (Hrsg.): *Proceedings International Conference on Computer Vision Systemsi (ICCV'99)*, Gran Canaria, Spain, 1999, S. 231 – 254.

[Wah97] W. Wahlster: *Verbmobil: Erkennung, Analyse, Transfer, Generierung und Synthese von Spontansprache*, *Spektrum der Wissenschaft — Dossier: Kopf oder Computer*, Oktober 1997, S. 52 – 56.

[Wan98] H. C. Wang, J. F. Wang: *A Telephone Number Inquiry System with Dialog Structure*, in *Proceedings International Conference on Acoustics, Speech and Signal Processing (ICASSP'98)*, Bd. 1, Seattle, Washington, 1998, S. 193 – 196.

[Win83] T. Winograd: *Language as a Cognitive Process*, Bd. 1: Syntax, Addison-Wesley, Reading, Massachusetts, 1983.

[Zha98] J. Zhang, Y. von Collani, A. Knoll: *Development of a Robot Interface Agent for Interactive Assembly*, in *Proceedings 4th International Symposium on Distributed Autonomous Systems*, Karlsruhe, 1998.

Anhang A

Wissensrepräsentation in der Verstehenskomponente

A.1 Definierte Segmente

Die folgende Tabelle gibt die derzeit in der Segmentdefinition festgelegten akzeptierten Segmente wieder. Sie sind neben einer Beschreibung jeweils durch ein Beispiel veranschaulicht.

Name	Beschreibung	Beispiel
$S_INT_OBJEKT	einfache Objektbenennung intendierter Objekte	„die lange grüne Schraube“
$S_REF_OBJEKT	Referenzobjekt zu einer einfachen Objektbenennung	„neben dem roten Klotz“
$S_OBJEKT_SPEZ.	Spezifikation eines Objektes oder Referenzobjektes	„mit den drei Löchern“
$S_AKTION	Aktionsbenennung	„stecke“
$S_INTERVENTION	Intervention	„Moment mal“
$S_ART_WEISE	Beschreibung einer Art und Weise	„von unten“
$S_OBJEKT_LOKAL	objektinterne Lokalisation	„das dritte Loch von rechts“
$S_HILFSOBJEKT	Hilfsobjekt zur Ausführung einer Handlung	„mit der langen Schraube“
$S_AGENT	benannter Agent	„du“
$S_SATZWORT	Floskeln und Ein–Wort–Sätze	„nein“

A.2 Konzeptdefinitionen

Die folgenden Tabellen geben die Konzeptdefinitionen in der ERNEST++–Wissensbasis für das Sprachverstehen im Konstruktionsszenario wieder.

Segmentebene

Konzeptdefinition	Beschreibung
SEGMENTEBENE	Allgemeine Konzeptdefinition der Segmentebene
SEGMENT	Allgemeine Konzeptdefinition der Segmente
S_AGENT	Repräsentation des Segmentes S_AGENT
S_AKTION	Repräsentation des Segmentes S_AKTION
S_ART_WEISE	Repräsentation des Segmentes S_ART_WEISE
S_ENDE	Konzeptdefinition zum Äußerungsende
S_HILFSOBJEKT	Repräsentation des Segmentes S_HILFSOBJEKT
S_INT_OBJEKT	Repräsentation des Segmentes S_INT_OBJEKT
S_INTERVENTION	Repräsentation des Segmentes S_INTERVENTION
S_KEIN_SEGMENT	Konzeptdefinition zur Aufnahme solcher Wörter, die keinem Segment zugeordnet werden.
S_OBJEKT_LOKAL	Repräsentation des Segmentes S_OBJEKT_LOKAL
S_OBJEKT_SPEZIFIKATION	Repräsentation des Segmentes S_OBJEKT_SPEZIFIKATION
S_REF_OBJEKT	Repräsentation des Segmentes S_REF_OBJEKT
S_SATZWORT	Repräsentation des Segmentes S_SATZWORT

Beschreibungsebene

Konzeptdefinition	Beschreibung
BESCHREIBUNGSEBENE	Allgemeine Konzeptdefinition der Beschreibungsebene
B_AKTION	Beschreibung einer Aktion
B_INSTRUKTION	Beschreibung einer Instruktion
B_INT_OBJEKT	Beschreibung eines intendierten Objektes
B_INTERVENTION	Beschreibung einer Intervention
B_KEIN_OBJEKT	Beschreibung solcher Wörter, die keinem Segment zugeordnet werden.
B_OBJEKT_BESCHREIBUNG	Umfassende Beschreibung eines Objektes
B_OBJEKT_LOKAL	Beschreibung einer objektinternen Lokalisation
B_REF_OBJEKT	Beschreibung eines Referenzbjektes

Dialogebene

Konzeptdefinition	Beschreibung
DIALOGEBENE	Allgemeine Konzeptdefinition der Dialogebene
D_DIALOG	Repräsentation des Mensch–Maschine–Dialogs
D_B_NACHRICHT	Repräsentation neuer Bildinformationen
D_I_ÄUSSERUNG	Repräsentation einer Instrukteursäußerung
D_I_INSTRUKTION	Repräsentation einer Instrukteursinstruktion
D_I_INTERVENTION	Repräsentation einer Instrukteursintervention
D_R_NACHRICHT	Repräsentation einer Nachricht vom Roboter
D_S_BEGRÜSSUNG	Repräsentation des Systemzustandes BEGRÜSSUNG
D_S_RÜCKFRAGE	Repräsentation des Systemzustandes RÜCKFRAGE
D_S_WIEDERHOLUNG	Repräsentation des Systemzustandes WIEDERHOLUNG
D_S_ZURÜCKWEISUNG	Repräsentation des Systemzustandes ZURÜCKWEISUNG
D_S_KONFUSION	Repräsentation des Systemzustandes KONFUSION
D_S_AUSFÜHRUNG	Repräsentation des Systemzustandes AUSFÜHRUNG
D_S_ABSCHLUSS	Repräsentation des Systemzustandes ABSCHLUSS

Anhang B

Interne Ergebnisse bei der Analyse eines repräsentativen Dialogs

Im folgenden werden die internen Ergebnisse für den repräsentativen Dialog auf Seite 111 dargestellt. Die Instruktionen sind ***fett kursiv***, die synthetisierten Systemausgaben `gesperrt` gedruckt.
Zunächst sind die Ergebnisse des Spracherkenners wiedergegeben. Die Zahlen in Klammern geben die Zeitpunkte für den Beginn und das Ende des Segmentes an. „UNK“ steht für solche Teile im Signal, denen kein Segment zugeordnet werden konnte. Die Interpretation jeder einzelnen Äußerung ohne einen Dialogkontext wird danach angegeben. Die Ausgabe des aktuellen Standes des Dialogs (mit anderen Worten: des Inhaltes des Merkmals *szeneauswertung* in der Instanz von D_DIALOG nach der Verarbeitung der aktuellen Äußerung) schließt sich daran an. Im Eintrag „Denotat“ sind die Referenten mit ihren Schwerpunkten im Bild festgehalten.

```
Guten Tag! Ich kann mit Baufix spielen!
Es geht sofort los! Ich warte auf Deine Anweisungen!
```

„Nimm eine Schraube.“

```
———— Ergebnis des Spracherkenners ————
    S_AKTION (5 - 23): nimm
    S_INT_OBJEKT (24 - 102): eine Schraube
———— Ende Ergebnis des Spracherkenners ————

———— Interpretation der aktuellen Äußerung ————
    Aktion: nehmen
    Objekt: Typ: Schraube
———— Ende Interpretation der aktuellen Äußerung ————

———— Aktueller Stand des Dialogs ————
    Aktion: nehmen
    Objekt: Typ: Schraube
            Denotat: Schraube [197.916 152.197]
                     Schraube [262.091 346.902]
———— Ende Aktueller Stand des Dialogs ————
```

```
Moment!
Meinst Du die blaue Schraube oder die orange Schraube?
```

„Die blaue.“

Ergebnis des Spracherkenners
S_INT_OBJEKT (3 - 59): die blaue
Ende Ergebnis des Spracherkenners

Interpretation der aktuellen Äußerung
Aktion: undef
Objekt: Typ: undef
Farbe: blau
Ende Interpretation der aktuellen Äußerung

Aktueller Stand des Dialogs
Aktion: nehmen
Objekt: Typ: Schraube
Farbe: blau
Denotat: Schraube [197.916 152.197]
Ende Aktueller Stand des Dialogs

```
Okay, ich nehme die blaue Schraube.
Was soll ich als nächstes tun?
```

„Steck’ dieses rote runde Teil auf die Schraube.“

Ergebnis des Spracherkenners
S_AKTION (0 - 26): stecke
S_INT_OBJEKT (27 - 185): dieses rote runde Teil
UNK (186 - 203): auf
S_INT_OBJEKT (204 - 269): die Schraube
Ende Ergebnis des Spracherkenners

Interpretation der aktuellen Äußerung
Aktion: stecken
Objekt: Typ: Objekt
Farbe: rot
Form: rund
Objekt: Typ: Schraube
Ende Interpretation der aktuellen Äußerung

Aktueller Stand des Dialogs
Aktion: stecken
Objekt: Typ: Objekt
Farbe: rot
Form: rund
Denotat: Radfelge [349.808 267.189]
Objekt: Typ: Schraube
Farbe: blau
Denotat: Schraube [197.916 152.197]
Ende Aktueller Stand des Dialogs

```
Okay, ich stecke die Felge auf die blaue Schraube.
Was soll ich als nächstes tun?
```

„Ähm, jetzt nimm . . . “

——— Ergebnis des Spracherkenners ———
UNK (0 - 27): ähm
UNK (28 - 71): jetzt
S_AKTION (72 - 108): nehmen
——— Ende Ergebnis des Spracherkenners ———

——— Interpretation der aktuellen Äußerung ———
Aktion: nehmen
——— Ende Interpretation der aktuellen Äußerung ———

——— Aktueller Stand des Dialogs ———
Aktion: nehmen
——— Ende Aktueller Stand des Dialogs ———

```
Moment! Da fehlt doch was!
Um etwas nehmen zu können, brauche ich doch auch irgendwelche
Objekte. Ich weiß aber nicht, welche Du meinst.
```

„Einen Klotz.“

——— Ergebnis des Spracherkenners ———
UNK (5 - 12): den
UNK (13 - 51): Kopf
——— Ende Ergebnis des Spracherkenners ———

——— Interpretation der aktuellen Äußerung ———
——— Ende Interpretation der aktuellen Äußerung ———

——— Aktueller Stand des Dialogs ———
Aktion: nehmen
——— Ende Aktueller Stand des Dialogs ———

```
Entschuldigung, was hast Du gesagt?
Ich weiß immer noch nicht, welches Teil du meinst.
```

„Ich meinte den blauen Klotz.“

——— Ergebnis des Spracherkenners ———
S_AGENT (4 - 17): ich
S_AKTION (18 - 48): meinte
S_INT_OBJEKT (49 - 145): den blauen Klotz
——— Ende Ergebnis des Spracherkenners ———

——— Interpretation der aktuellen Äußerung ———
Aktion: denken
Objekt: Typ: Schraubwürfel
Farbe: blau
——— Ende Interpretation der aktuellen Äußerung ———

Aktueller Stand des Dialogs
Aktion: nehmen
Objekt: Typ: Schraubwürfel
Farbe: blau
Denotat: Schraubwürfel [521.292 175.822]
Ende Aktueller Stand des Dialogs

```
Okay, ich nehme den blauen Würfel.
Was soll ich als nächstes tun?
```

„Schraube den Würfel auf die blaue Schraube.“

Ergebnis des Spracherkenners
S_AKTION (3 - 39): schraube
S_INT_OBJEKT (40 - 86): den Würfel
UNK (87 - 101): auf
S_INT_OBJEKT (102 - 194): die blaue Schraube
Ende Ergebnis des Spracherkenners

Interpretation der aktuellen Äußerung
Aktion: schrauben
Objekt: Typ: Schraubwürfel
Objekt: Typ: Schraube
Farbe: blau
Ende Interpretation der aktuellen Äußerung

Aktueller Stand des Dialogs
Aktion: schrauben
Objekt: Typ: Schraubwürfel
Farbe: blau
Denotat: Schraubwürfel [521.292 175.822]
Objekt: Typ: Schraube
Farbe: blau
Denotat: Schraube [197.916 152.197]
Ende Aktueller Stand des Dialogs

```
Okay, ich verschraube den blauen Würfel und die blaue Schraube.
Was soll ich als nächstes tun?
```

Stichwortverzeichnis