Deutsches
Forschungszentrum
für Künstliche
Intelligenz GmbH

Document
D-92-16

Verzeichnis von Softwarekomponenten für natürlichsprachliche Systeme

Ergebnisse einer Umfrage im Rahmen der VERBMOBIL-Vorbereitung

Judith Engelkamp (Hrsg.)

Mai 1992

Deutsches Forschungszentrum für Künstliche Intelligenz GmbH

Postfach 20 80
D-6750 Kaiserslautern, FRG
Tel.: (+49 631) 205-3211/13
Fax: (+49 631) 205-3210

Stuhlsatzenhausweg 3
D-6600 Saarbrücken 11, FRG
Tel.: (+49 681) 302-5252
Fax: (+49 681) 302-5341

Deutsches Forschungszentrum für Künstliche Intelligenz

The German Research Center for Artificial Intelligence (Deutsches Forschungszentrum für Künstliche Intelligenz, DFKI) with sites in Kaiserslautern and Saarbrücken is a non-profit organization which was founded in 1988. The shareholder companies are Atlas Elektronik, Daimler Benz, Fraunhofer Gesellschaft, GMD, IBM, Insiders, Mannesmann-Kienzle, Philips, SEMA Group Systems, Siemens and Siemens-Nixdorf. Research projects conducted at the DFKI are funded by the German Ministry for Research and Technology, by the shareholder companies, or by other industrial contracts.

The DFKI conducts application-oriented basic research in the field of artificial intelligence and other related subfields of computer science. The overall goal is to construct *systems with technical knowledge and common sense* which - by using AI methods - implement a problem solution for a selected application area. Currently, there are the following research areas at the DFKI:

- Intelligent Engineering Systems
- Intelligent User Interfaces
- Intelligent Communication Networks
- Intelligent Cooperative Systems.

The DFKI strives at making its research results available to the scientific community. There exist many contacts to domestic and foreign research institutions, both in academy and industry. The DFKI hosts technology transfer workshops for shareholders and other interested groups in order to inform about the current state of research.

From its beginning, the DFKI has provided an attractive working environment for AI researchers from Germany and from all over the world. The goal is to have a staff of about 100 researchers at the end of the building-up phase.

Prof. Dr. Gerhard Barth
Director

Verzeichnis von Softwarekomponenten für natürlichsprachliche Systeme

Judith Engelkamp (Hrsg.)

DFKI-D-92-16

Diese Arbeit wurde finanziell unterstützt durch das Bundesministerium für Forschung und Technologie.

Verzeichnis von Softwarekomponenten für natürlichsprachliche Systeme

Ergebnisse einer Umfrage im Rahmen der VERBMOBIL-Vorbereitung

Judith Engelkamp (Hrsg.)

Deutsches Forschungszentrum für Künstliche Intelligenz
Stuhlsatzenhausweg 3
D-W-6600 Saarbrücken 11
Tel: (0681) 302 - 5252
Fax: (0681) 302 - 5341
engelkamp@coli.uni-sb.de

Mai 1992

Vorwort

Das DFKI (Deutsches Forschungszentrum für Künstliche Intelligenz) wurde vom BMFT (Bundesministerium für Forschung und Technologie) mit der Durchführung einer Umfrage zu existierenden Software-Komponenten im Bereich Verarbeitung natürlicher Sprache beauftragt (413 - 4001 - 01 IV 201). Das Ziel der Umfrage war die Erstellung einer Übersicht von in Deutschland verfügbaren Software-Komponenten, die im Bereich der natürlichsprachlichen Systeme für das Projekt VERBMOBIL relevant sein könnten. Das Ergebnis dieser Umfrage liegt nun vor.

Zur Durchführung der Umfrage wurde ein Fragebogen erstellt, der im März 1992 über die News-Gruppe mod-ki verbreitet und außerdem an ca. 400 Adressen geschickt wurde (Mitglieder der Gesellschaft für Informatik e.V. FA 1.3 „Natürliche Sprache", Mitglieder der DGfS, Sektion Computerlinguistik).

Das Verzeichnis ist auf in Deutschland entwickelte Software beschränkt und enthält akademische, kommerzielle und geschützte Software, wobei jeweils angegeben ist, unter welchen Bedingungen die Komponenten erhältlich sind.

Judith Engelkamp

Strukturierung des Verzeichnisses

Das vorliegende Verzeichnis enthält 106 ausgefüllte Fragebögen. Die Antworten wurden anhand folgender Kriterien gegliedert:

- Spracherkennung und Sprachdatensammlung
- Sprachsynthese
- Lexikon und Morphologie
- Analyse natürlicher Sprache: Grammatik, Parsing und Entwicklungsumgebung
- Semantische, Dialog- und Wissensverarbeitung
- Generierung
- Architektur
- Tools
- Schnittstellen und Systeme

Die Ordnung der Komponenten in den einzelnen Abschnitten orientiert sich am Eingangsdatum.

Der Anhang enthält zwei Register: im ersten werden die Namen der Komponenten in alphabetischer Reihenfolge aufgeführt, das zweite listet die Institute auf, an denen die Komponenten entwickelt wurden oder eingesetzt werden.

Es folgt der verschickte Fragebogen sowie eine Liste der Komponenten mit einer Kurzbeschreibung.

Deutsches
Forschungszentrum
für Künstliche
Intelligenz GmbH

Judith Engelkamp
DFKI Saarbrücken
Stuhlsatzenhausweg 3
6600 Saarbrücken
Telefon: 0681/302-4495
Telefax: 0681/302-4351
e-mail: engelkamp@coli.uni-sb.de

Umfrage zu existierenden Software-Komponenten im Bereich Verarbeitung natürlicher Sprache

Das DFKI (Deutsches Forschungszentrum für Künstliche Intelligenz) wurde vom BMFT (Bundesministerium für Forschung und Technologie) mit der Durchführung dieser Umfrage beauftragt. Das Ziel ist die Erstellung einer Übersicht von verfügbaren Software-Komponenten, die im Bereich der natürlichsprachlichen Systeme für das Projekt VERBMOBIL relevant sein könnten.

VERBMOBIL steht als Arbeitstitel für ein BMFT-Projekt der mobilen Übersetzung spontan gesprochener Sprache. Es geht langfristig darum, ein System zur Übersetzung von Verhandlungsdialogen in face-to-face Situationen zu entwickeln. Dabei wird angenommen, daß die beiden Gesprächspartner, z.B. ein Deutscher und ein Japaner, Englisch als Dialogsprache benutzen und zumindest passiv beherrschen. Zu diesem Projekt, das voraussichtlich Ende des Jahres gestartet wird und wahrscheinlich eine Laufzeit von zweimal vier Jahren haben wird, ist eine öffentliche Ausschreibung für verschiedenen Teilprojekte geplant.

Folgende Schwerpunktgebiete werden in VERBMOBIL bearbeitet:

- Spracherkennung
- Lexikon
- Morphologie
- Parsing
- Syntax
- Semantikkonstruktion
- Semantische Auswertung
- Wissensverarbeitung
- Dialog
- Transfer/Übersetzung
- Generierung

- Sprachsynthese

Um bei der Übersetzung von spontan in Dialog-Situationen gesprochener Sprache entscheidende Fortschritte erzielen zu können, sollen existierende Software-Komponenten auf ihre Verwendbarkeit in VERBMOBIL geprüft werden.

Um eine möglicht vollständige Übersicht zu erhalten, sind wir daran interessiert, möglichst viele Antworten zu bekommen, und bitten Sie daher darum, den Fragebogen an alle eventuell Interessierten weiterzuleiten. Auch unvollständig ausgefüllte Fragebogen sind willkommen. Allerdings sollten zumindest die ersten sechs Fragen beantwortet werden.

Sollte der auf dem Fragebogen vorgesehene Platz für die Beantwortung einer Frage nicht ausreichen, so bitten wir Sie, die Antworten auf einem zusätzlichen Blatt zu notieren.

Wir werden Ihnen nach Abschluß der Studie die Umfrageergebnisse auf Anfrage zuschicken.

Die Antworten sollten Sie (nach Möglichkeit per e-mail) bis spätestens 4.4.92 richten an (der Fragebogen kann auch per e-mail angefordert werden):

Judith Engelkamp	e-mail: engelkamp@coli.uni-sb.de
DFKI Saarbrücken	Telefon: 0681/302-4495
Stuhlsatzenhausweg 3	Telefax: 0681/302-4351
6600 Saarbrücken	

1. **Name der Komponente**

2. **Zweck der Komponente**

3. **Entwickelt und implementiert durch**

4. **Anschrift, Kontaktperson, e-mail**

5. **Implementierungssprache**

6. **Zugrundeliegende Hardware / Betriebssystem(e)**

7. Beginn / Ende der Entwicklung

8. Projekt

9. Input / Eingabeschnittstellen

10. Output / Ausgabeschnittstellen

11. Aktueller Zustand (z.B. Produkt, noch in der Entwicklung)

12. Abhängigkeiten / Anforderungen an andere Systeme

13. Dokumentation

14. Verfügbarkeit (Copyright, Preis)

15. Wartbarkeit

16. Fremdinstallationen

17. Benutzte Techniken und zugrundeliegende Theorien

18. Allgemeine Beschreibung

Spezielle Fragen für Lexikon und Grammatik:

19. Entwicklungsumgebung zur Erstellung der Einträge

20. Anzahl und Art der Einträge

Literaturangaben

Namen der Komponenten mit Kurzbeschreibung in alphabethischer Reihenfolge:

1. 3R Software: Sprecherunabhaengiger Einzelworterkenner (ca. 50 Worte), sprecherabhaengiger Einzelworterkenner (ca. 50 Worte), 2 Player zur Wiedergabe digitalisierter Sprache, die alle gleichzeitig arbeiten koennen.

2. ADKMS-Datenbankinterface: natuerlichsprachlicher Dialog mit einem erweiterten relationalen Datenbanksystem ORACLE (R)

3. AKUSTIK-PHONETIK Tools: Signalvorverarbeitung, Kurzzeitanalyse (Spektrum, Cepstrum, AKF, LPC), Merkmalauswahl, Diskriminanzanalyse, Automatische Klassifikation / Vektorquantisierung, überwachtes / unüberwachtes Lernen

4. Akustisch-artikulatorische Sprachsignalanalyse: Programmpaket zur simultanen Erfassung und Auswertung von EMA- (elektromagnetische Artikulographie), EPG- (elektrisch- dynamische Palatographie) und akustischen Sprachsignaldaten

5. ALEX_Pc: Bildschirmorientiertes Zugriffssystem fuer die Bonner Wortdatenbank (BONNLEX)

6. AMOS (= A MOrphosyntactical expert System): morphosyntaktische Analyse althebraeischer Texte

7. Antwortgenerierungskomponente von NAUDA: Erweiterung einer NL-DB-Schnittstelle (LanguageAccess) um natuerlichsprachl. Antworten bei: - geeignetem DB-Output (Tabellen mit 1 numerischen Wert) - Zurueckweisung/Korrektur von Praesuppositionsverletzungen - Ueberbeantwortung bei Fragen nach numerischen Werten

8. ASL-Chart: Einsatz als Parser ueber Worthypothesen (und anderen Hypothesen) in einem stark interaktiven (parallelen) integrierten Speech-Language-System

9. ASL-Nord-Architektur: Kommunikationsorientierte Architektur fuer das ASL-Nord-System. Die Komponente bildet einen Rahmen fuer die Verarbeitungskomponenten eines Speech-Language-Systems. Dieser Rahmen stellt den Verarbeitungskomponenten ein Kommunikationssystem zu Verfuegung, mit dessen Hilfe der Austausch von partiellen Analyseresultaten ermoeglicht wird. Zusaetzlich koennen die Komponenten einen interaktiven Dialog fuehren mit dem Ziel der Aufloesung von lokalen Ambiguitaeten durch Ermittlung nicht-lokaler Information. Die Architektur orientiert sich an verteilten Systemen und an Ergebnissen aus der Verteilten KI (DAI).

10. ATNP: Parsing

11. AUTFIT: Automatische Extraktion der Steuerparameter des quantitativenIntonationsmodells von Fujisaki, fuer das Deutsche modifiziert undweiterentwickelt durch B. Moebius und M. Paetzold, zur Nachbildungnatuerlichsprachlicher F0-Verlaeufe.

12. BACK: Berlin Advanced Computational Knowledge Representation System Version 4.4 - - 4. October 1991: Repräsentation terminologischen und assertionalen Wissens (Sprach- und Weltwissen) in der Tradition terminologischer Logiken (KL-ONE).

13. BONNLEX: Lexikalische Datenbank als Quelle fuer natuerlichsprachliche Systeme

14. CAT2: Multilinguale MUe

15. Charon-System: System zur transferbasierten maschinellen Uebersetzung mittels LFG- Grammatiken. Durch den modularen Aufbau eignet sich das System auch zum einzelsprachlichen Testen von Grammatiken und laesst sich als Testumgebung fuer den Einsatz von Parsern, Generatoren und andere NL-Komponenten, wie z.B. Semantikkomponenten und Morphologiekomponenten verwenden. Die Minimalkonfiguration der Entwicklungsumgebung besteht aus einem Kommandointerpreter, einem Parser und einem Generator.

16. CHART-DISPLAY: Darstellung der Chart waehrend des Parsing

17. Conceptual Modelling Language CML: Wissensrepraesentationskomponente fuer logikbasierte Modellierung von Domaenen- und Weltwissen, Benutzermodellierung, Dialogmodellierung. Schlussfolgerungskomponenten fuer deduktives Schliessen, Konsistenzpruefung, Defaults, Abduktion. Erstellung von wissensbasierten Anwendungen

18. Context Feature Structure System (CFS-System): Graphunifikationsformalismus

19. COSIMA: Experimentalsystem zur Erkennung kontinuierlich gesprochener deutscher Sprache.

20. CUF-System-Kern: Entwicklung und Verarbeitung von CUF-Programmen (z.B. HPSG-artigen Grammatiken, die je nach Form des Goals Sätze parsen bzw. generieren)

21. DCG-Workbench: Entwicklung von DCG-basierten Grammatiken fuer verschiedene Sprachen

22. DDL Tool - Dialogue Description Language and Dialogue Design Tool: Der Benutzerdialog wird auf einer Workstation mit Hilfe des Dialogue Design Tools entworfen. Das DDL Tool ist ein maechtiges grafisches Werkzeug, das drei Schichten umfasst. Es er- moeglicht die Spezifizierung, Implementierung und Verifikation des Dialogs, sowie einige Systemtests.

23. DECtalk: Vollsynthetisches Sprachausgabesystem fuer deutsche Sprache.

24. Dialogkomponente von NAUDA: Erweiterung einer NL-DB-Schnittstelle (LanguageAccess) um kooperative Aspekte: - Erkennung / Korrektur von Praesuppositionsverletzungen bezueglich Anzahl- und Rollenrestriktionen des begrifflichen Wissens - Ueberbeantwortung bei E-Fragen nach numerischen Werten - Information ueber Begriffe, wenn eine zurueckgewiesene Frage nur leicht modifiziert erneut gestellt wird.

25. DiTo (Diagnostic Tool for German Syntax): Fehlerdiagnose innerhalb der Syntaxkomponente von Systemen, die natuerliche Sprache verarbeiten; Kontrolle der Systemperformanz und Unterstuetzung der Konsistenzerhaltung bei der Verarbeitung syntaktischer Phaenomene; Entwicklung von Syntaxfragmenten

26. EGG - Editor fuer Generalisierte Phrasenstruktur-Grammatiken (GPSG): Verfassen von GPS-Grammatiken zur Verwendung in MUe-Systemen

27. ELF/ELR Repraesentation: Erzeugung einer formal-semantischen Repraesentation

28. Erkennung kontinuierlicher Sprache mit grossem Wortschatz

29. ERNEST (ERlanger NEtzwerk SysTem): Systemschale zur Repraesentation von deklarativem und prozeduralem Wissen und zur effizienten Verarbeitung (problemunabhaengig, Basis A*-Algoritmus, Bewertungsvektor) des dargestellten Wissens auf der Basis eines semantischen Netzes

30. ERNEST: Eine Software-Umgebung zur wissensbasierten Musteranalyse

31. Evaluationsverfahren zur Anaphern-Resolution: Finden des besten Antezedenten fuer ein Pronomen durch Anwendung verschiedener Bewertungskriterien und Gewichte. Aktualisierung der Funktor-Argument-Strukturen (FAS).

32. EVAR-Lexikon: Lexikon fuer ein sprachverstehendes System mit sauberer Trennung von Grundwortschatz und Anwendungswortschatz

33. EVAR: Ein sprachverstehendes System zum Führen eines informationsabfragenden Dialogs; Pilotanwendung InterCity-Zugauskunft

34. Feature Editor fuer Macintosh: Vollstaendig interaktive (GUI) Anzeigen und Veraendern von Featurestrukturen von (fast) beliebigen Merkmalsformalismen

35. Feature Editor fuer X-Windows: Vollstaendig interaktive (GUI) Anzeigen und Veraendern (noch in Entwicklung) von Featurestrukturen von (fast) beliebigen Merkmalsformalismen

36. Finite-State-Morphologie: Morphologische Analyse und Generierung

37. FSS-WASTL: Interaktives Wissensakquisitionskomponente zur Erweiterung des semantischen Lexikons im natuerlichsprachlichen Zugangssystem XTRA

38. Generierung von Worthypothesen: Generierung von Worthypothesenketten oder -graphen zu einem Sprachsignal. Ein statistisches Sprachmodell (Bigramm Grammatik) wird verwendet.

39. Grammatik/Lexikon fuer DISCO: Grammatik/Lexikon fuer Dialog-System

40. GuLP (General unification-based Linguistic Processor): Parsing

41. G_LOG: Erweitertes Prologsystem: Featureterme + Featureunifikation, einfache 'constraint resolution', Modulkonzept: Wissenspakete, Datenbankanschluß in Vorbereitung.

42. HADIFIX: Sprachsynthese

43. Hypertext-System fuer die Lexikographie: on-line Unterstuetzung bei der lexikographischen Datenerfassung sowie semantische Analyse von Lexemen

44. IBM SAA LanguageAccess: Datenabfrage in natürlicher Sprache (Englisch, Deutsch)

45. ICM - Interpretation and Control Module: Der ICM handhabt den Dialog des Systems mit dem Benutzer gemaess einer zugrundeliegenden Dialogbeschreibung. Der ICM ist dabei Applikationsunabhaengig. Mehrere ICMs koennen im System gleichzeitig aktiv, d.h. mehrere (unterschiedliche) Applikationen und Dialoge koennen vom System gleichzeitig gehand- habt werden. Aufgaben des ICM: Parsing der Dialogbeschreibung, Interpretation und Kontrolle des Dialogs zwischen Benutzer und Applikation, Kontrolle von Ein- /Ausgabegeraeten und Applikationen, Fehlerbehandlung.

46. ILSP: Detektion phonologisch relevanter akustischer Ereignisse im Signal, z.B. Okklusionslösung, Stimmhaftigkeit, ...

47. Interaktiver Grapher.: Visualisierungs- und Inspektionstool für Typhierarchien (a la HPSG), die mit TDL ExtraLight erstellt wurden

48. Intercity- Zugauskunft: Fuehren eines Auskunftsdialogs ueber ein begrenztes Anwendungsgebiet. Die akustische Eingabe ist kontinuierlich gesprochene Sprache.

49. Interpreter und Entwicklungsumgebung fuer nicht konfluente Termersetzungssysteme in der maschinellen Sprachuebersetzung: Parsing, semantische Analyse, konzeptuelle Analyse, Transfer und Generierung

50. ISADORA (Integrated System for Automatic Decoding of Observation sequences of Real- values Arrays): Musteranalyse komplexer eindimensionaler Muster, insbesondere 'Automatische Spracherkennung'

51. KLEIST: (Text-) Generierung von Wegbeschreibungen in deutscher und japanischer Sprache

52. KOMET: Multilinguale Textgenerierung und Textplanung.

53. konnektionistische Implementation eines Parsers zur Simulationen der syntaktischen Verarbeitung von Aphasiepatienten: Simulation der Verarbeitung von sprachlichen Äußerungen bei Aphasiepatienten.

54. Konnektionistischer Parser fuer kontextfreie Grammatiken Demonstration der Funktion und Arbeitsweise von konnektionistischen Parsern.

55. Konnektionistisches Modell der Sprachproduktion: Modellierung von Teilprozessen des kognitiven Prozesses der Sprachproduktion

56. LEU/2 Lexikonmanager: Der Lexikonmanager bidet die lexikalische Komponente der "LILOG Experimentier Umgebung"(LEU/2). DIese Komponente stellt die lexikalische Information für die Analyse und Generierung natürlicher Sprache zur Verfügung. Das System ist dabei in der Lage, eine Menge interaktiver Strategien zur Behandlung unbekannter Wörter anzuwenden.

57. LEU/2, LILOG-KR Ergaenzungen zur Integration von Wissenspaketen: Aufbau strukturierter Hintergrundwissensbasen (Ontologie und Axiomatik)

58. LILOG-Inferenzmaschine: Inferenzsystem zur Verarbeitung der hybriden Wissensrepraesentationssprache L-LILOG

59. Linguistik-Komponente: Semantisch-pragmatische Verarbeitungskomponente für ein sprachverstehendes System

60. Linguistischer Kernprozessor (LKP): 1) Analyse einer deutschen Eingabe 2) Generierung einer deutschen Ausgabe

61. MAIDAI: Begruendungsverwaltende ABox fuer Termdefinitions-sprachen (bevorzugt solche, die Default Information beschreiben koennen)

62. MAUS (Meaning Acquisition And Understanding System): Empirische Ermittlung von semantischen, dispositionellen Dependenzstrukturen (DDS) aus Corpora natürlich-sprachlicher Texte

63. MediTAS 1.0: Syntax-Analyse von Sätzen (in deutscher Sprache) aus dem klinischen Routinebetrieb

64. METEXA: Wissensbasierte Analyse radiologischer Befundungstexte und Generierung von Erwartungen zur Unterstuetzung von Systemen zur Erkennung kontinuierlich gesprochener Sprache

65. MODALYS: Mittels einer semantisch-pragmatischen Analyse von Modalverben werden die Systemreaktionen auf modalisierte Eingaben in natürlichsprachlichen Dialogsystemen gesteuert. Eine Teilaufgabe dabei ist die Disambiguierung zwischen den moeglichen Lesarten von Modalverben.

66. MORPHIX-3: Flexionsanalyse und -synthese

67. NLI-AIDOS: Natürlichsprachliches Interface fuer das Informationssystem AIDOS

68. NLL: SOWOHL semantische Repraesentation ALS AUCH semantische Verarbeitung (semantische Inferenzen)

69. NUGGET (eingetr.Warenzeichen der SNI AG): Generierung natürlichsprachlicher, dynamischer Erkldrungen und Fakten, sowie Regeln im Rahmen von Expertensystemanwendungen der Shell TWAICE (R)

70. P-center-Analyse: Bestimmung der Rhythmuspunkte im akustischen Sprachsignal

71. P-TRA: Regelbasierte phonetische Transkription deutschsprachiger Texte.

72. Parser fuer Grammatiken im Format von TDL/Udine: Analyse von Saetzen in natuerlicher Sprache auf der Grundlage einer in TDL/Udine erstellten Grammatik.

73. Parser fuer GPS-Grammatiken (konstruktive Version): syntaktische Analyse von natürlichsprachlichen Sätzen

74. Parsing (im TOPIC-Projekt)

75. PAULA (Parsing Algorithm for UG-based Language Analysis): Parsing und Syntaxanalyse natuerlicher Sprache

76. PHONDAT-Aufnahmesystem: Zur schnellen digitalen Erfassung akustischer Sprachsignale

77. PHONWORK: Phonetische Werkbank zur interaktiven Segmentation und Ettikettierung grosser Mengen akustischer Sprachsignaldaten unter auditiver und visueller Kontrolle (incl. digitales Oszillogramm)

78. Pitchsynchrone Sprachsignalmanipulation: Programmsystem zur PSOLA-aehnlichen Sprachschallsybthese

79. PLAIN-D Grammatik (Templates): Grundlage fuer den PLAIN+ Parser

80. PLAIN-D Lexikon: Maschinenlesbares Lexikon des Deutschen fuer NLP-Systeme, enthaelt alle Information, die ein Parser benoetigt (Morphologische Angaben, Valenzangaben)

81. Playmobild: Testumgebung zur Entwicklung reversibler Grammatiken und Verfahren; umfasst Parser, Generator, Patr aehlichen Merkmalsformalismus

82. POPEL: Generierungssystem fuer multimodale Dialogbeitraege

83. Prosodie-Modul: Erstellung einer Betonungsbeschreibung oder einer Frage/Nicht-Frage- Klassifikation oder einer prosodischen Verifikation von Worthypothesen

84. SBLITTERS: erweiterte ABox fuer SBONE (Termdefinitionssprache)

85. SCAN: konnektionistische und hybride (symbolisch/konnektionistische) Verarbeitung von natuerlicher Sprache, insbesondere Verarbeitung von syntaktischen und semantischen konnektionistischen Repraesentationen fuer strukturelle Disambiguierung und semantische Klassifizierung von Nominalphrasen

86. SCM - SUNSTAR Communication Manager: Zentraler Bestandteil der offenen Systemarchitektur. Verbindet die Software Module des Systems, die auf unterschiedlichen Rechnern (Workstations, PCs) unter unterschiedlichen UNIX- Implementationen laufen koennen. Aufgaben: Management der Kommunikation zwischen Prozessen und Rechnern, Handhabung der Systemkonfiguration (Belegung von Geraeten

und Applikationen, Initialisierung und Kontrolle der Systemkomponenten, Systemueberwachung und Fehlerbehandlung.

87. Semantikbasiertes maschinelles Uebersetzungssystem, Grammatiktyp LFG, Semantiktyp DRT.

88. Semantische Constraint-Modellierung: Prueft gleichzeitig mit der syntaktischen Analyse, ob bestimmte allgemeine, und domainspezifische Constraints zwischen den Phrasen oder Phrasenteilen erfuellt sind, um so falsche akustische Hypothesen zu eliminieren und falsches Attachment in der Analyse zu verhindern. Die semantischen Beziehungen werden auch waehrend der Anaphernresolution genutzt: erstens um Pronomen eine temporaere, vom Satzkontext abhaengige semantische Kategorie zuzuweisen; zwietens: um die semantische Vertraeglichkeit von Antezedenz-Beziehungen zu gewaehrleiten.

89. SpeechMaster Entwicklungssystem 1.0: Spracherkennung und -interpretation

90. SPREADIAC - a SPREADIng ACtivator: SPREADIAC ist ein Spreading Activation System, das auf KL-ONEartigen Wissensnetzen arbeitet. Die Auflösung der beschriebenen Referenzen ist die Hauptanwendung von SPREADIAC.

91. Sprecherinterpolation: Programmsystem zur automatischen Interpolation zwischen vorgegebenen akustischen Sprachsignalpaaren (insbesondere textgleiche Äußerungen zweier Sprecher)

92. SUNSTAR - Systemarchitektur fuer Sprachverarbeitende Systeme: Eine Systemarchitektur zur einfachen und schnellen Generierung sprachgesteuerter Anwendungen

93. Syntaktische Analyse: Identifizierung der korrekten akustischen Satzhypothese und Erzeugung eines syntaktischen Strukturbaumes

94. T D L ExtraLight: Erstellung und Test von unifikationsbasierten Lexika und Grammatiken (d.h. von Merkmalstypen a la HPSG) für NL Systeme

95. T D L: Erstellung und Test von unifikationsbasierten Lexika und Grammatiken (d.h. von Merkmalstypen a la HPSG) für NL Systeme

96. TACTILUS: Zeigegestenanalyse fuer NL-Dialogsysteme

97. TAGDevEnv - Tree Adjoining Grammar Development Environment: TAGDevEnv ist eine Werkbank zur Unterstuetzung der Arbeit eines Grammatikentwicklers fuer den Grammatikformalismus der Tree Adjoining Grammars. Es stehen unter einer benutzerfreundlichen Menueoberflaeche verschiedene Dienste zur Verfuegung.

98. TDL2LATEX: Uebersetzung von interner Darstellung von TDL in LATEX Code

99. Testtool: Integritaetstest des Unifikators

100. TFS-System: Repräsentation und constraint-basierte Verarbeitung von einsprachigem und kontrastivem lexikalischem und grammatischem Wissen für sprachverarbeitende Systeme

101. TYSIS: Sammlung von ca. 200 Sprachsignalverarbeituns-Routinen fuer PDP-11 (RT-11) mit LPS-11 System und GT 40

102. UDINE: Unifikator fuer Merkmalslogik

103. VERBALIZE: Ueberfuehrung einer logischen Form in F-Strukturen; Besonderheit dabei ist die Moeglichkeit von erheblichen Umstrukturierungen durch inverse Lexeminterpretation

104. WESPA (Waveform Editor & SPectral Analyzer): Darstellen, Messen, Manipulieren und Segmentieren ('Labeln') von Sprachsignalen im Zeitbereich. Analyse und Darstellung des Kurzeit- Leistungsspektrums in zwei- und dreidimensionaler Form (Sonagraphie).

105. WIP-TAG-GEN: Syntaxkomponente zur inkrementellen Generierung natuerlicher Sprache

106. x2morF (eXtended 2-level morphology with Filters): Analyse und Generierung von Wortformen

Inhaltsverzeichnis

1 Spracherkennung und Sprachdatensammlung

1.1 ILSP

1. *Name der Komponente:* ILSP
2. *Zweck der Komponente:* Detektion phonologisch relevanter akustischer Ereignisse im Signal, z.B. Okklusionslösung, Stimmhaftigkeit, ...
3. *Entwickelt und implementiert durch:* Andreas Hauenstein, Kai Hübener
4. *Anschrift, Kontaktperson, e-mail:* kai@nats5.informatik.uni- hamburg.de, andreas @nats4.informatik.uni-hamburg.de, Andreas Hauenstein, Kai Hübener, Universität Hamburg, Fachbereich Informatik, Arbeitsbereich Naürlichsprachliche Systeme, Bodenstedtstr. 16, 2000 Hamburg 50
5. *Implementierungssprache:* C++
6. *Zugrundeliegende Hardware / Betriebssystem(e):* SPARC, SunOS 4.1
7. *Beginn / Ende der Entwicklung:* Beginn: 1.10.91 Ende: 12.94 ? Prototyp Ende 92
8. *Projekt:* ASL
9. *Input / Eingabeschnittstellen:* Eingabe aus einem Gehörmodell. Außerdem Erwartungen der phonologischen Ebene zur Einschränkung des Suchraums.
10. *Output / Ausgabeschnittstellen:* Mit Konfidenzen und Zeitangaben versehene Ereignisse zur phonologischen Ebene.
11. *Aktueller Zustand:* in der Entwicklung
12. *Abhängigkeiten / Anforderungen an andere Systeme:* Interaktionen mit dem Gehörmodell und der phonologischen Ebene essentiell.
17. *Benutzte Techniken und zugrundeliegende Theorien:* Scale Space Approach, phonological knowledge

Literaturangaben:

James Robert Glass: Finding acoustical regularities in speech: applications to phonetic recognition. Tech. Rep. 536, MIT, December 1988.

A. Hauenstein, K. Hübener: Phonological knowledge for speech recognition. ASL-TR-12-91/UHH, 1991

Andrew P. Witkin: Scale Space Filtering: A new approach to multi-scale description. In: Proc of the IEEE, pp 1-4, 1984

1.2 3R Software

1. *Name der Komponente:* 3R Software

2. *Zweck der Komponente:* Sprecherunabhaengiger Einzelworterkenner (ca. 50 Worte), sprecherabhaengiger Einzelworterkenner (ca. 50 Worte), 2 Player zur Wiedergabe digitalisierter Sprache, die alle gleichzeitig arbeiten koennen.

3. *Entwickelt und implementiert durch:* Projekt SUNSTAR

4. *Anschrift, Kontaktperson, e-mail:* Fraunhofer-Institut fuer Arbeitswirtschaft und Organisation Nobelstrasse 12 7000 Stuttgart 80 Dipl.-Ing. T. Renner, Dipl.-Ing. J. Brettschneider renner@iao.fhg.de, brettsch@iao.fhg.de

5. *Implementierungssprache:* C, Assembler

6. *Zugrundeliegende Hardware / Betriebssystem(e):* DSP32C Signalprozessor Einsteckkarte fuer PC

11. *Aktueller Zustand:* Der Spracherkenner wird augenblicklich weiterentwickelt. Zum Projektende wird sprecherunabhaengige Erkennung kontinuierlich gesprochener Sprache moeglich sein.

1.3 COSIMA

1. *Name der Komponente:* COSIMA
2. *Zweck der Komponente:* Experimentalsystem zur Erkennung kontinuierlich gesprochener deutscher Sprache.
3. *Entwickelt und implementiert durch:* Fraunhofer-Institut IAO
4. *Anschrift, Kontaktperson, e-mail:* Fraunhofer-Institut fuer Arbeitswirtschaft und Organisation Nobelstrasse 12 7000 Stuttgart 80 Dipl.-Ing. T. Renner, Dipl.-Ing. J. Brettschneider renner@iao.fhg.de, brettsch@iao.fhg.de
5. *Implementierungssprache:* FORTRAN, C
6. *Zugrundeliegende Hardware / Betriebssystem(e):* MicroVax, VMS SUN Workstations, SUN OS3 oder hoeher

1.4 WESPA

1. *Name der Komponente:* WESPA (Waveform Editor & SPectral Analyzer)

2. *Zweck der Komponente:* Darstellen, Messen, Manipulieren und Segmentieren ('Labeln') von Sprachsignalen im Zeitbereich. Analyse und Darstellung des Kurzeit-Leistungsspektrums in zwei- und dreidimensionaler Form (Sonagraphie).

3. *Entwickelt und implementiert durch:* Dr. Dieter Stock, IKP Bonn

4. *Anschrift, Kontaktperson, e-mail:* siehe 3.; ueber wgh@wgh.ikp.uni- bonn.de

5. *Implementierungssprache:* FORTRAN 77 + 386-Assembler

6. *Zugrundeliegende Hardware / Betriebssystem(e):* AT 486 unter DOS; AD/DA-Wandler von DATA TRANSLATION, S-VGA - Graphikkarte mit 0,5 MB Speicher und Graphics Controller Chip ET 4000, HP Laserjet III.

9. *Input / Einagbeschnittstellen:* Analogsignale direkt von Mikrofon

11. *Aktueller Zustand:* Lauffaehig mit noch einigen Verbesserungen in der Bedienung

14. *Verfügbarkeit:* Verkauf nach Angebot. Kein Produkt im Rahmen von ASL.

1.5 P-TRA

1. *Name der Komponente:* P-TRA
2. *Zweck der Komponente:* Regelbasierte phonetische Transkription deutschsprachiger Texte.
3. *Entwickelt und implementiert durch:* Dr. Dieter Stock, IKP Bonn
4. *Anschrift, Kontaktperson, e-mail:* siehe 3.; ueber wgh@wgh.ikp.uni- bonn.de
5. *Implementierungssprache:* FORTRAN 77
6. *Zugrundeliegende Hardware / Betriebssystem(e):* PC AT, DOS
9. *Input / Einagbeschnittstellen:* Text in ASCII
11. *Aktueller Zustand:* Lauffaehig
14. *Verfügbarkeit:* Verkauf nach Angebot. Kein Produkt im Rahmen von ASL.

1.6 AUTFIT

1. *Name der Komponente:* AUTFIT

2. *Zweck der Komponente:* Automatische Extraktion der Steuerparameter des quantitativen Intonationsmodells von Fujisaki, fuer das Deutsche modifiziert und weiterentwickelt durch B. Moebius und M. Paetzold, zur Nachbildung natuerlichsprachlicher F0-Verlaeufe.

3. *Entwickelt und implementiert durch:* Matthias Paetzold, Bernd Moebius

4. *Anschrift, Kontaktperson, e-mail:* Prof. Dr. Wolfgang Hess, IKP; wgh@wgh.ikp.uni-bonn.de

5. *Implementierungssprache:* IBM PC RT VS FORTRAN

6. *Zugrundeliegende Hardware / Betriebssystem(e):* IBM PC RT 6150, AIX 2.2

7. *Beginn / Ende der Entwicklung:* 1989 - April 1991; Weiterentwicklung laeuft (siehe 11.)

8. *Projekt:* FG-Projekt He 1019/5 Intonationsverlaeufe

9. *Input / Einagbeschnittstellen:* F0-Messwerte mit manuell markierten Akzentgruppen- und Phrasengrenzen

10. *Output / Ausagbeschnittstellen:* Steuerparameter des Intonationsmodells und die daraus generierte Intonationskontur

11. *Aktueller Zustand:* Lauffaehig. Portierung auf IBM- kompatible PC AT und SUN Workstations wird vorgenommen, Implementierung in C / C++ ist angestrebt. Der Algorithmus wird derzeit fuer Zwecke der automatischen Spracherkennung in ASL-Nord umgestaltet. Die regelbasierte F0-Steuerung fuer Zwecke der Sprachsynthese ist bereits moeglich.

14. *Verfügbarkeit:* Kein Verkauf. Befristete Weitergabe an ASL-Partner im Rahmen des ASL-Kooperationsvertrages; dies wird auch fuer Verbmobil gelten, sofern das IKP einer der Verbmobil-Partner sein wird.

16. *Fremdinstallationen:* Grafik-Routinen (W. Hess, IKP)

17. *Benutzte Techniken und zugrundeliegende Theorien:* on linguistischen Restriktionen abhaengige Approximation, zum Teil nach der Methode der kleinsten Fehlerquadrate, per Suchverfahren. Intonationsmodell von Fujisaki (1983; 1988), Anpassung fuer das Deutsche mit linguistischer Interpretation (Moebius et al., 1991; Moebius, 1992).

18. *Allgemeine Beschreibung:* Beschreibung des Algorithmus in (Paetzold, 1991).

Literaturangaben: Fujisaki Hiroya (1983): "Dynamic characteristics of voice fundamental frequency in speech and singing". In P.F. MacNeilage (Hg.), The production of speech (Springer, New York), S. 39-55

Fujisaki Hiroya (1988): "A note on the physiological and physical basis for the phrase and accent components in the voice fundamental frequency contour". In O. Fujimura (Hg.), Vocal physiology: voice production, mechanisms and functions (Raven, New York), S. 347-355

Moebius Bernd, Demenko Grazyna, Paetzold Matthias (1991): "Parametrische Beschreibung von Intonationskonturen". In W. Hess, W.F. Sendlmeier (Hg.), Beitraege zur angewandten und experimentellen Phonetik (Steiner, Stuttgart), S. 109-124

Moebius Bernd (1992): Ein quantitatives Modell der deutschen Intonation - Analyse und Synthese von Grundfrequenzverlaeufen (Diss., Univ. Bonn)

Paetzold Matthias (1991): Nachbildung von Intonationskonturen mit dem Modell von Fujisaki - Implementierung des Algorithmus und erste Experimente mit ein- und zweiphrasigen Aussagesaetzen (Mag.-Arb., Univ. Bonn)

1.7 Erkennung kontinuierlicher Sprache mit grossem Wortschatz

1. *Name der Komponente:*
2. *Zweck der Komponente:* Erkennung kontinuierlicher Sprache mit grossem Wortschatz
3. *Entwickelt und implementiert durch:* Philips GmbH, Forschungslaboratorien
4. *Anschrift, Kontaktperson, e-mail:* Dr. Martin Oerder Philips GmbH, Forschungslaboratorien Weisshausstrasse 2, 5100 Aachen oerder@pfa.philips.de
5. *Implementierungssprache:* C
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Hardware/OS UNIX
7. *Beginn / Ende der Entwicklung:* laufende Entwicklung

1.8 SpeechMaster Entwicklungssystem 1.0

1. *Name der Komponente:* SpeechMaster Entwicklungssystem 1.0
2. *Zweck der Komponente:* Spracherkennung und - interpretation
3. *Entwickelt und implementiert durch:* aspect Gesellschaft für Mensch-Maschine Kommunikation mbH
4. *Anschrift, Kontaktperson, e-mail:* aspect Gesellschaft für Mensch- Maschine Kommunikation mbH, Henning Bergmann, Gutenbergring 38, D- 2000 Norderstedt, Tel: 040/528 2050, Fax: 040/ 523 9749
5. *Implementierungssprache:* C, Assembler
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Lüft auf IBM PCs unter DOS oder WINDOWS mit SpeechMaster-Zusatzkarte (basiert auf Motorola Signalprozessor DSP56001). Unix-Einbindung über TCP/IP (in Vorbereitung).
7. *Beginn / Ende der Entwicklung:* Juni 90 bis Oktober 91 (Version 1.0)
8. *Projekt:* öffentliche Förderung durch Ministerium für Wirtschaft, Technik und Verkehr, Schleswig-Holstein.
9. *Input / Eingabeschnittstellen:* Analoger Spracheingan (Mikrofon), Programmschnittstelle zur Einbindung in Anwendungsprogramm
10. *Output / Ausgabeschnittstellen:* Analoger Sprachausgang (Lautsprecher), Programmschnittstelle zur Einbindung in Anwendungsprogramm
11. *Aktueller Zustand:* Produkt
12. *Abhängigkeiten / Anforderungen an andere Systeme:*
    - DOS / Windows: PC mit mindestens 386er Prozessor (25MHz), DOS oder Windows
    - UNIX: Systemvorraussetzung für Servermaschine: PC mit mindestens 386er Prozessor (25MHz), DOS oder Windows, TCP/IP Runtime-System, LAN
      Systemvorraussetzung für Clientmaschine: UNIX-Rechner, TCP/IP, LAN
13. *Dokumentation:* BenutzerHandbuch (ca. 100 S.) Deutsch oder Englisch
14. *Verfügbarkeit:* Entwicklungssystem ist sofort verfügbar. Preis 2.900 DM + MWSt. Laufzeitsysteme (PC-Zusatzboard ohne Entwicklungssoftware) auf Anfrage.
15. *Wartbarkeit:* Hersteller unterstützt bei technischen Fragen. Update- Service.
16. *Fremdinstallationen:* ca. 10

17. *Benutzte Techniken und zugrundeliegende Theorien:* Signalverarbeitung: FFT; Spracherkennung: Markov-Modelle, endliche Netze; Sprachinterpretation: kompositionale Semantik, syntaxgesteuerte Übersetzung

18. *Allgemeine Beschreibung:* Entwicklungssystem zur Einbindung von Spracherkennung in Anwendungsprogramme Leistungsmerkmale: Kontinuierliche Spracherkennung; Wortschatz: mehrere hundert frei definierbare Wörter pro Anwendung (abhängig vom verfügbaren Hauptspeicher); davon können maximal 50 gleichzeitig aktiv sein; Sprecherabhängig.

    Systemkomponenten: Zusatzkarte für PCs, einschließlich Signalverarbeitungssoftware; Software: Netzwerkcompiler zur Erstellung / Übersetzung von Lexikon und Grammatik, Trainningsprogramm, Laufzeitsystem (Erkennung, Interpretation)

19. *Entwicklungsumgebung zur Erstellung der Einträge:* Text- Editor, NETCOMP-Netzwerk-Compiler (setzt Lexikon und Grammatik in endliche Netze um).

20. *Anzahl und Art der Einträge:* Lexikon: Wort + Menge von Attribut-Wert-Paaren. Grammatik: kontextfreie Regeln mit semantischen Gleichungen (ohne Rekursion).

Literaturangaben:

H. Bergmann, H.-H. Hamer, A. Noll, A. Paeseler, H. Tomaschewski: An adaptable Man-Machine Interface Using Connected-Word Recognition on Speech. In: Proc. of the EUROSPEECH 91 - 2nd European Conference on Speech Communication and Technology, Genua, Italien, 1991.

1.9 Sprecherinterpolation

1. *Name der Komponente:* Sprecherinterpolation

2. *Zweck der Komponente:* Programmsystem zur automatischen Interpolation zwischen vorgegebenen akustischen Sprachsignalpaaren (insbesondere textgleiche Äußerungen zweier Sprecher)

3. *Entwickelt und implementiert durch:* Mitarbeiter des Instituts

4. *Anschrift, Kontaktperson, e-mail:* PD Dr. Bernd Pompino-Marschall Institut fuer Phonetik und Sprachliche Kommunikation der Universitaet Muenchen Schellingstr. 3/II/VG Tel.: (+89) 21 80 - 27 58 Fax: (+89) 28 00 362 e-mail: fipkm@phonetik.uni-muenchen.dbp.de

5. *Implementierungssprache:* Assembler, FORTRAN und C

6. *Zugrundeliegende Hardware / Betriebssystem(e):* PDP-11 (RT-11), Micro-Vax (VMS) mit DSP board (TMS 320C25)

1.10 P-center-Analyse

1. *Name der Komponente:* P-center-Analyse
2. *Zweck der Komponente:* Bestimmung der Rhythmuspunkte im akustischen Sprachsignal
3. *Entwickelt und implementiert durch:* Mitarbeiter des Instituts
4. *Anschrift, Kontaktperson, e-mail:* PD Dr. Bernd Pompino-Marschall Institut fuer Phonetik und Sprachliche Kommunikation der Universitaet Muenchen Schellingstr. 3/II/VG Tel.: (+89) 21 80 - 27 58 Fax: (+89) 28 00 362 e-mail: fipkm@phonetik.uni-muenchen.dbp.de
5. *Implementierungssprache:* FORTRAN
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Micro-Vax (VMS)

1.11 Akustisch-artikulatorische Sprachsignalanalyse

1. *Name der Komponente:* Akustisch-artikulatorische Sprachsignalanalyse
2. *Zweck der Komponente:* Programmpaket zur simultanen Erfassung und Auswertung von EMA- (elektromagnetische Artikulographie), EPG- (elektrisch-dynamische Palatographie) und akustischen Sprachsignaldaten
3. *Entwickelt und implementiert durch:* Mitarbeiter des Instituts
4. *Anschrift, Kontaktperson, e-mail:* PD Dr. Bernd Pompino-Marschall Institut fuer Phonetik und Sprachliche Kommunikation der Universitaet Muenchen Schellingstr. 3/II/VG Tel.: (+89) 21 80 - 27 58 Fax: (+89) 28 00 362 e-mail: fipkm@phonetik.uni-muenchen.dbp.de
5. *Implementierungssprache:* FORTRAN, Assembler, TMS-Assembler
6. *Zugrundeliegende Hardware / Betriebssystem(e):* AT (MS-DOS) mit Head-acoustics-Karte und IEEE-Interface, Sony PCM 2500 DAT-Recorder, John Hardy M-1 Mikrophon- Vorverstaerker, PDP-11 Laborrechner (RT-11) mit DSP board (TMS 320C25)

1.12 PHONDAT-Aufnahmesystem

1. *Name der Komponente:* PHONDAT-Aufnahmesystem
2. *Zweck der Komponente:* Zur schnellen digitalen Erfassung akustischer Sprachsignale gelesener Äußerungen (mit Textvorgabe via Bildschirm)
3. *Entwickelt und implementiert durch:* Mitarbeiter des Instituts
4. *Anschrift, Kontaktperson, e-mail:* PD Dr. Bernd Pompino-Marschall Institut fuer Phonetik und Sprachliche Kommunikation der Universitaet Muenchen Schellingstr. 3/II/VG Tel.: (+89) 21 80 - 27 58 Fax: (+89) 28 00 362 e-mail: fipkm@phonetik.uni-muenchen.dbp.de
5. *Implementierungssprache:* C und Assembler
6. *Zugrundeliegende Hardware / Betriebssystem(e):* AT (MS-DOS) mit Head-acoustics-Karte, Sony PCM 2500 DAT-Recorder, John Hardy M-1 Mikrophon-Vorverstaerker

1.13 Phonwork

1. *Name der Komponente:* PHONWORK

2. *Zweck der Komponente:* Phonetische Werkbank zur interaktiven Segmentation und Ettikettierung grosser Mengen akustischer Sprachsignaldaten unter auditiver und visueller Kontrolle (incl. digitales Oszillogramm)

3. *Entwickelt und implementiert durch:* Mitarbeiter des Instituts

4. *Anschrift, Kontaktperson, e-mail:* PD Dr. Bernd Pompino-Marschall Institut fuer Phonetik und Sprachliche Kommunikation der Universitaet Muenchen Schellingstr. 3/II/VG Tel.: (+89) 21 80 - 27 58 Fax: (+89) 28 00 362 e-mail: fipkm@phonetik.uni-muenchen.dbp.de

5. *Implementierungssprache:* FORTRAN, Assembler, TMS-Assembler und C

6. *Zugrundeliegende Hardware / Betriebssystem(e):* VAX-Station 3200 (VMS) mit DSP board (TMS 320C25) sowie programmierbarer clock zur externen Ansteuerung des DA-Wandlers

1.14 TYSIS

1. *Name der Komponente:* TYSIS

2. *Zweck der Komponente:* Sammlung von ca. 200 Sprachsignalverarbeituns-Routinen fuer PDP-11 (RT-11) mit LPS-11 System und GT 40

3. *Entwickelt und implementiert durch:* Mitarbeiter des Instituts

4. *Anschrift, Kontaktperson, e-mail:* PD Dr. Bernd Pompino-Marschall Institut fuer Phonetik und Sprachliche Kommunikation der Universitaet Muenchen Schellingstr. 3/II/VG Tel.: (+89) 21 80 - 27 58 Fax: (+89) 28 00 362 e-mail: fipkm@phonetik.uni-muenchen.dbp.de

5. *Implementierungssprache:* Assembler, FORTRAN

6. *Zugrundeliegende Hardware / Betriebssystem(e):* PDP-11 (RT-11) mit Peripherie

1.15 Prosodie-Modul

1. *Name der Komponente:* Prosodie-Modul
2. *Zweck der Komponente:* Erstellung einer Betonungsbeschreibung oder einer Frage/ Nicht-Frage-Klassifikation oder einer prosodischen Verifikation von Worthypothesen
3. *Entwickelt und implementiert durch:* Lehrstuhl fuer Informatik 5 (Mustererkennung)
4. *Anschrift, Kontaktperson, e-mail:* E. Nöth Lehrstuhl fuer Informatik 5 (Mustererkennung) Martensstr. 3 8520 Erlangen

   Tel.: 09131/85 7888 Fax.: 09131/303811

   e-mail: noeth@informatik.uni-erlangen.de
5. *Implementierungssprache:* in C und FORTRAN
6. *Zugrundeliegende Hardware / Betriebssystem(e):* DEC- Rechner unter ULTRIX
7. *Beginn / Ende der Entwicklung:* 1985 / laufend
8. *Projekt:* BMFT
9. *Input / Eingabeschnittstellen:* Abtastwerte oder Worthypothesengitter
10. *Output / Ausgabeschnittstellen:* Betonungsbeschreibung oder Frage/Nicht-Frage-Klassifikation oder prosodische Verifikation von Worthypothesen
11. *Aktueller Zustand:* laufendes Software-Produkt
12. *Abhängigkeiten / Anforderungen an andere Systeme:* nein
13. *Dokumentation:* [Noe91]
14. *Verfügbarkeit:* ja, auf Anfrage
16. *Fremdinstallationen:* nein
17. *Benutzte Techniken und zugrundeliegende Theorien:* siehe [Noe91]
18. *Allgemeine Beschreibung:* siehe [Noe91]

Literaturangaben:

[Noe91] E. Nöth Prosodische Information in der automatischen Spracherkennung - Berechnung und Anwendung Niemeyer, 1991.

1.16 ISADORA

1. *Name der Komponente:* ISADORA (Integrated System for Automatic Decoding of Observation sequences of Real-values Arrays)

2. *Zweck der Komponente:* Musteranalyse komplexer eindimensionaler Muster, insbesondere 'Automatische Spracherkennung'

3. *Entwickelt und implementiert durch:* Lehrstuhl fuer Informatik 5 (Mustererkennung), Universitaet Erlangen-Nuernberg

4. *Anschrift, Kontaktperson, e-mail:* Ernst G. Schukat- Talamazzini, Lehrstuhl fuer Informatik 5 (Mustererkennung), Universitaet Erlangen-Nuernberg, Martensstrasse 3, D-8520 ERLANGEN Telefon +49 (09131) 85-7873

5. *Implementierungssprache:* 'C'

6. *Zugrundeliegende Hardware / Betriebssystem(e):* UNIX, bisher installiert auf: DEC VAX Workstations 2000, 3000, 5000 (Uni Erl., Bayrisches FORWISS, Uni Bielefeld) Sun-3,4 (Uni Erl., Siemens Muenchen)

7. *Beginn / Ende der Entwicklung:* Ende 1988 - (dato)

8. *Projekt:* Entwicklung: Uni Erlangen Vorarbeiten: BMfT Einsatz in diversen Vorhaben: Esprit P2218 SUNDIAL (UniErl, SiemensMue, DaimlerUlm) MESSPERT /AUDI (FZ wissensbasierte Systeme Erl.) [LS-eigenes Vorhaben] (Uni Bielefeld, Ang. Informatik)

9. *Input / Eingabeschnittstellen:* PCM-Darstellung von Sprachsignalen akustisch-phonetisch-morphologisch-syntaktisches Regelwerk

10. *Output / Ausgabeschnittstellen:* hierarchische symbolische Beschreibungen

11. *Aktueller Zustand:* in Entwicklung; funktions- + einsatzfaehig

13. *Dokumentation:* Handbuch

14. *Verfügbarkeit:* n. Vereinb.

15. *Wartbarkeit:* n. Vereinb.

16. *Fremdinstallationen:* s.o.

17. *Benutzte Techniken und zugrundeliegende Theorien:* [a] statistisch orientierte Mustererkennung (HMM = Hidden Markov Models) [b] hierarchisches Netzwerk zur Repraesentation phonetischer, morphologischer und syntaktischer Spracheinheiten

18. *Allgemeine Beschreibung:*

    ISADORA ist ein automatisch lernendes System zur Analyse komplexer eindimensionaler Muster, das im folgenden allerdings nur in Bezug auf die Erkennung kontinuierlich gesprochener Sprache behandelt wird. Das System gliedert sich in einen deklarativen Teil, bestehend aus einem hierarchischen Konstituentennetzwerk und den statistischen Parametern akustischer Sprachmodelle (HMM), und einem prozeduralen Teil, der Algorithmen zur Emissionsdichteberechnung, Optimalpfadsuche und Parameterschätzung im Rahmen der HMM-Methodologie sowie deren geeignete Verallgemeinerungen zur Verarbeitung beliebiger Netzwerkkonzepte umfaßt.

    Die beiden Grundpfeiler, auf denen die Konzeption ISADORA's ruht, sind eine wirkungsvolle Nutzung der HMM-Technologie und eine Art Kompositionalitaets-Prinzip für die akustische Repräsentation phonetischer Spracheinheiten.

    Die Grundmotivation zum Aufbau eines Netzwerkformalismus sprachlicher Einheiten bildet die Beobachtung, daß in zahlreichen HMM-Algorithmen zur Spracherkennung zunehmend systematische Verknüpfungen kleinerer HMMs zu größeren akustischen Modellen vorgenommen wird. Beispiele sind die Hintereinanderschaltung, die Parallelschaltung, die Rückkopplung oder genereller die Anordnung von Markovmodellen in Form eines endlichen Zustandsnetzwerks.

    Allen genannten Konstruktionen ist gemeinsam, daß ihr Resultat wiederum von der Form eines HMM ist. Die obenstehenden Verknüpfungsmechanismen, die erstmalig in Algorithmen zur Einzel- und Verbundworterkennung, zur Modellbildung mit Wortuntereinheiten und zur syntaxgesteuerten Spracherkennung auftauchten, werden im Netzwerkteil ISADORA's zu einem generellen Mechanismus der sukzessiven Konstruktion immer komplexerer akustischer, HMM- basierter Modelle systematisiert.

19. *Entwicklungsumgebung zur Erstellung der Einträge:* Regelsystemgetriebene Zerlegung von Spracheinheiten (z.B. Wort -> Silbe -> Halbsilbe -> Cluster -> Phoneme) Integration von Korpora, Vokabularen, Syntaxen.

20. *Anzahl und Art der Einträge:* ca. 2000 Woerter (phonetische Repraesentation, s.o.)

Literaturangaben:

E.G. Schukat-Talamazzini, H. Niemann: Das ISADORA-System - ein akustisch-phonetisches Netzwerk zur automatischen Spracherkennung. Proc. 13. DAGM-Symposium, Springer, 1991, S. 251-258.

1.17 AKUSTIK-PHONETIK Tools

1. *Name der Komponente:* AKUSTIK-PHONETIK Tools

2. *Zweck der Komponente:* Signalvorverarbeitung Kurzzeitanalyse (Spektrum, Cepstrum, AKF, LPC) Merkmalauswahl, Diskriminanzanalyse Automatische Klassifikation / Vektorquantisierung überwachtes / unüberwachtes Lernen

3. *Entwickelt und implementiert durch:* Lehrstuhl fuer Informatik 5 (Mustererkennung), Universitaet Erlangen-Nuernberg

4. *Anschrift, Kontaktperson, e-mail:* Ernst G. Schukat- Talamazzini, Lehrstuhl fuer Informatik 5 (Mustererkennung), Universitaet Erlangen-Nuernberg, Martensstrasse 3, D-8520 ERLANGEN Telefon +49 (09131) 85-7873

5. *Implementierungssprache:* 'C' / UNIX

6. *Zugrundeliegende Hardware / Betriebssystem(e):* UNIX, bisher installiert auf: DEC VAX Workstations 2000, 3000, 5000, Sun-3,4

7. *Beginn / Ende der Entwicklung:* Ende 1986 - (dato)

8. *Projekt:* Entwicklung: Uni Erlangen, ESPRIT P 2218

11. *Aktueller Zustand:* funktions- + einsatzfaehig

13. *Dokumentation:* Handbuch

14. *Verfügbarkeit:* n. Vereinb.

15. *Wartbarkeit:* n. Vereinb.

16. *Fremdinstallationen:* s.o.

17. *Benutzte Techniken und zugrundeliegende Theorien:* Digitale Signalverarbeitung (s.o.) Numerische Klassifikation und Vektorquantisierung: BAYES-Klassifikator unter Normalverteilungsannahme

18. *Allgemeine Beschreibung:* (Handbuch)

1.18 Generierung von Worthypothesen

1. *Name der Komponente:* enerierung von Worthypothesen
2. *Zweck der Komponente:* Generierung von Worthypothesenketten oder -graphen zu einem Sprachsignal. Ein statistisches Sprachmodell (Bigramm Grammatik) wird verwendet.
3. *Entwickelt und implementiert durch:* Lehrstuhl fuer Informatik 5 (Mustererkennung)
4. *Anschrift, Kontaktperson, e-mail:* Prof. H. Niemann Lehrstuhl fuer Informatik 5 (Mustererkennung) Martensstr. 3 8520 Erlangen Tel.: 09131/85 7774 (Prof. H. Niemann) Tel.: 09131/85 7890 (T. Kuhn) Fax.: 09131/303811 e-mail: niemann@informatik.uni-erlangen.de
5. *Implementierungssprache:* C / UNIX
6. *Zugrundeliegende Hardware / Betriebssystem(e):* DecStation 5000/200, ULTRIX 4.2
7. *Beginn / Ende der Entwicklung:* Beginn: 1983 Ende: laufend
8. *Projekt:* ESPRIT P 2218 BMfT
9. *Input / Eingabeschnittstellen:* Framehypothesen aus weicher Vektorquantisierung
10. *Output / Ausgabeschnittstellen:* beste Wortkette, n-beste Wortketten, Worthypothesen, Wortgraph
11. *Aktueller Zustand:* laufendes Softwareprodukt (wird staendig weiterentwickelt)
12. *Abhängigkeiten / Anforderungen an andere Systeme:* nein
13. *Dokumentation:* teilweise vorhanden
14. *Verfügbarkeit:* ja, auf Anfrage
16. *Fremdinstallationen:* ja, auf einer SUN 4 bei Siemens Muenchen
17. *Benutzte Techniken und zugrundeliegende Theorien:* Hidden Markov Modelle, one-stage-Algorithmus, Bigramm Grammatik
18. *Allgemeine Beschreibung:*

    Das Verfahren basiert auf einem zweistufigen Ansatz mit dem zunaechst explizit segmentiert und dann Worthypothesen generiert werden. Jeder der realisierten Algorithmen arbeitet von links nach rechts um zu gewaehrleisten, dass das Sprachsignal zeitsynschron verarbeitet werden kann. Die Segmentierung gliedert sich in vier Phasen, die in [Kun89] ausfuehrlich beschrieben sind. In der ersten Phase werden homogene Bereiche in den Framehypothesen zu Segmenten zusammengefasst. Unter Betrachtung potentieller Endpunkte fuer jeden Anfangspunkt wird in der zweiten Phase ein Segmentgraph generiert, dessen Kanten in der dritten Phase unter

Verwendung von Hidden Markov Modellen anhand eines Inventars in lautaehnliche Einheiten klassifiziert werden. In der vierten Phase wird mit einer Best-First Suche nach dem besten Segmentpfad gesucht. Die Ausgabe der Segmentierung bildet damit eine lineare Segmentfolge, wobei jedem Segment mehrere lautaehnliche bewertete Hypothesen zugeordnet sein koennen. Der linerare Segmentstrom bildet die Eingabe fuer die zweite Stufe des Erkenners. Den Basisalogrithmus bildet der one-stage-Algorithmus. Waehrend der Analyse kann optional ein stochatsisches Bigramm Modell basierend auf Wortkategorien verwendet werden. Eine ausfuehrliche Beschreibung findet man in [Kuhn92].

Literaturangaben:

[Kunz91] S. Kunzmann Die Worterkennung in einem Dialogsystem fuer kontinuierlich gesprochene Sprache Niemeyer 1991

[Kuhn92] T. Kuhn, H. Niemann, E.G. Schukat-Talamazzini, W. Eckert, S. Rieck; Context-dependent modeling in a two-stage HMM word recognizer for continuous speech EUSIPCO 92

2 Sprachsynthese

2.1 DECtalk

1. *Name der Komponente:* DECtalk
2. *Zweck der Komponente:* Vollsynthetisches Sprachausgabesystem fuer deutsche Sprache.
3. *Entwickelt und implementiert durch:* Fraunhofer-Institut IAO
4. *Anschrift, Kontaktperson, e-mail:* Fraunhofer-Institut fuer Arbeitswirtschaft und Organisation Nobelstrasse 12 7000 Stuttgart 80 Dipl.-Ing. T. Renner, Dipl.-Ing. J. Brettschneider renner@iao.fhg.de, brettsch@iao.fhg.de
5. *Implementierungssprache:*
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Stand-alone Geraet, das ueber V24 Schnittstelle angesteuert wird.

2.2 HADIFIX

1. *Name der Komponente:* HADIFIX
2. *Zweck der Komponente:* Sprachsynthese
3. *Entwickelt und implementiert durch:* homas Portele, Walter F. Sendlmeier, Wolfgang Hess, Birgit Steffan, Rainer Preuss
4. *Anschrift, Kontaktperson, e-mail:* Thomas Portele, Institut fuer Kommunikationsforschung und Phonetik, Poppelsdorfer Allee 47, W-5300 Bonn 1, Tel.: 0228/735680 Fax: 0228/735639, e-mail: ueber wgh@wgh.ikp.uni-bonn.de
5. *Implementierungssprache:* C++
6. *Zugrundeliegende Hardware / Betriebssystem(e):* PC mit Intel 80386/80486 und DA-Konverter, MS-DOS
7. Beginn: 1.1.1989, Ende: in 1992
8. *Projekt:* FG-Projekt He 1019/4
9. *Input / Einagbeschnittstellen:* Lautzeichenfolge mit Akzentmarken
10. *Output / Ausagbeschnittstellen:* Sprachsignal
11. *Aktueller Zustand:* prototypischer Zustand
13. *Dokumentation:* vorhanden; Publikationen auf Anfrage erhaeltlich
14. *Verfügbarkeit:* Experimentelles System; Weitergabe nicht beabsichtigt; Benutzung ggf. auf Anfrage
16. *Fremdinstallationen:* keine
17. *Benutzte Techniken und zugrundeliegende Theorien:* SOLA, RELP, Fujisakis Intonationsmodell
18. *Allgemeine Beschreibung:* HADIFIX erzeugt aus einer Lautzeichenfolge mit Akzentmarken ein Sprach- signal. Hierfuer werden kurze Abschnitte natuerlicher Sprache (silbenorien- tierte Einheiten) im Zeitbereich konkateniert. Ein in einer speziellen Sprache formuliertes Regelsystem mit Regeln zur Einheitenauswahl, Konkate- nation und Prosodiesteuerung wird compiliert und auf die Eingabe angewandt. Als Ergebnis liegt eine Amweisungsfolge fuer das Konkatenationsprogramm vor. Dieses erzeugt daraus ein Sprachsignal durch die Konkatenation der in einem Inventar abgelegten Einheiten, erstellt eine Intonationskontur unter Ver- wendung des Modells von Fujisaki, nimmt die erforderlichen prosodischen Manipulationen am Signal vor (PSOLA), und macht das so manipulierte Sprach- signal durch Ausgabe ueber einen DA-Wandler hoerbar.

2.3 Pitchsynchrone Sprachsignalmanipulation

1. *Name der Komponente:* Pitchsynchrone Sprachsignalmanipulation
2. *Zweck der Komponente:* Programmsystem zur PSOLA-aehnlichen Sprachschallsynthese zum Zwecke der Erzeugung phonetischer Teststimuli
3. *Entwickelt und implementiert durch:* Mitarbeiter des Instituts
4. *Anschrift, Kontaktperson, e-mail:* PD Dr. Bernd Pompino-Marschall Institut fuer Phonetik und Sprachliche Kommunikation der Universitaet Muenchen Schellingstr. 3/II/VG Tel.: (+89) 21 80 - 27 58 Fax: (+89) 28 00 362 e-mail: fipkm@phonetik.uni-muenchen.dbp.de
5. *Implementierungssprache:* FORTRAN, Assembler, TMS-Assembler
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Micro-VAX (VMS) mit DSP board (TMS 320C25)

3 Lexikon und Morphologie

3.1 FSS-WASTL

1. *Name der Komponente:* FSS-WASTL

2. *Zweck der Komponente:* Interaktives Wissensakquisitionskomponente zur Erweiterung des semantischen Lexikons im natuerlichsprachlichen Zugangssystem XTRA

3. *Entwickelt und implementiert durch:* Alassane Ndiaye

4. *Anschrift, Kontaktperson, e-mail:* ndiaye@cs.uni-sb.de, Alassane NDIAYE, FB 14 - Informatik IV, PHONE:++49-681-302-4135, Universitaet des Saarlandes

5. *Implementierungssprache:* LISP

6. *Zugrundeliegende Hardware / Betriebssystem(e):* a) Symbolics (Genera) oder b) SUN/Solbournes (SUNOS + Lucid) oder c) HP (HP-UX + Lucid)

8. *Projekt:* XTRA

9. *Input / Eingabeschnittstellen:* natuerlichsprachlicher Beispielsatz mit einem unbekannten wort

10. *Output / Ausgabeschnittstellen:* Lexikoneintrag fuer das unbekannte Wort im semantischen Lexikon

12. *Abhängigkeiten / Anforderungen an andere Systeme:* das System legt SB-ONE (bzw. KL-ONE) und SEMANTIX (semantischer Parser in XTRA) zugrunde. Allerdings sind einige Teile (z.B. der interactive classifier) leicht "anpassbar")

17. *Benutzte Techniken und zugrundeliegende Theorien:* matching zur Hypothesenberechnung, interaktive klassifikation, paraphrasierung

18. *Allgemeine Beschreibung:* FSS-WASTL is a system for the acquisition of semantic knowledge within XTRA – a natural language access system to expert systems. Starting from user-supplied example utterances, hypotheses for lexical entries are induced, which then are refined top down through interactive classification. The refinement process of FSS-WASTL is a variant of the well-known KL-ONE classification procedure and is an extension of T. Finin's interactive classifier. The integration of XTRA's natural language generator, which is a highlight of FSS-WASTL, allows for the paraphrasing of the user's input for clarification purposes. FSS-WASTL is compared with other research systems. It is argued that this approach can be transfered to applications beyond the natural language domain.

Literaturangaben:

1. R.M. **Jansen-Winkeln**, A. **Ndiaye** und N. **Reithinger**. *FSS-WASTL – Interactive Knowledge Acquisition for a Semantic Lexicon.* In: E. Ardizonne, S. Gaglio, and F. Sorbello (eds.), Trends in Artificial Intelligence, Proceedings of the second AI*IA, Lectures Notes on Artificial Intelligence 529, pp. 108–116, Springer Verlag, 1991.

2. A. **Ndiaye**. *FSS-WASTL – Ein interaktives Wissensakquisitionssystem zur Erweiterung des semantischen Lexikons im natürlichsprachlichen Zugangssytem XTRA.* Diplomarbeit, Fachbereich Informatik, Universität des Saarlandes, Saarbrücken, 1990.

3.2 EVAR-Lexikon

1. *Name der Komponente:* EVAR-Lexikon
2. *Zweck der Komponente:* Lexikon fuer ein sprachverstehendes System mit sauberer Trennung von Grundwortschatz und Anwendungswortschatz
3. *Entwickelt und implementiert durch:* Lehrstuhl fuer Informatik 5 (Mustererkennung)
4. *Anschrift, Kontaktperson, e-mail:* Prof. H. Niemann Lehrstuhl fuer Informatik 5 (Mustererkennung) Martensstr. 3 8520 Erlangen

   Tel.: 09131/85 7774 Fax.: 09131/303811

   e-mail: niemann@informatik.uni-erlangen.de
5. *Implementierungssprache:* in FRANZ-Lisp unter hbase
6. *Zugrundeliegende Hardware / Betriebssystem(e):* DEC-CISC- Rechner unter ULTRIX 2.0
7. *Beginn / Ende der Entwicklung:* 1984 / laufend
8. *Projekt:* BMFT
9. *Input / Eingabeschnittstellen:* ASCII, menue-gesteuert
10. *Output / Ausgabeschnittstellen:* ASCII
11. *Aktueller Zustand:* laufendes Software-Produkt
12. *Abhängigkeiten / Anforderungen an andere Systeme:* nein
13. *Dokumentation:* [Ehr90]
14. *Verfügbarkeit:* Es ist ein ASCII-Abzug des Lexikons verfuegbar.
15. *Wartbarkeit:* Schwierig, da FRANZ-Lisp von DEC nicht mehr unterstuetzt wird und fehlerhaft ist.
16. *Fremdinstallationen:* nein
17. *Benutzte Techniken und zugrundeliegende Theorien:* siehe [Ehr90]
18. *Allgemeine Beschreibung:* siehe [Ehr90]
19. *Entwicklungsumgebung zur Erstellung der Einträge:* Ein in FRANZ-Lisp unter ULTRIX entwickeltes Lexikon-Programm, beschrieben in [Ehr90]
20. *Anzahl und Art der Einträge:* Vollformen: ca. 4600 Einträge mit Standardumschrift nach DUDEN davon ca. 2500 Einträge mit syntaktischer und semantischer Information davon ca. 160 Grundformen mit anwendungsabhaengiger Semantik (Zugauskunft)

Literaturangaben:

[NNM92] H. Niemann, E. Nöth, M. Mast, E.G. Schukat-Talamazzini Ein Lexikon für ein natürlich-sprachliches Dialogsystem in D. Reimann Beiträge des ASL-Lexikonworkshops ASL-TR-40-92/ZSB, Berlin, 1992.

[EHR90] U. Ehrlich Bedeutungsanalyse in einem sprachverstehenden System unter Berücksichtigung pragmatischer Faktoren Niemeyer, 1990.

3.3 PLAIN-D Lexikon

1. *Name der Komponente:* PLAIN-D Lexikon

2. *Zweck der Komponente:* Maschinenlesbares Lexikon des Deutschen fuer NLP- Systeme, enthaelt alle Information, die ein Parser benoetigt (Morphologische Angaben, Valenzangaben)

3. *Entwickelt und implementiert durch:* LCL, Uni Heidelberg

4. *Anschrift, Kontaktperson, e-mail:* Prof. P. Hellwig, Lehrstuhl fuer Computerlinguistik, Universitaet Heidelberg, Karlstr. 2, 6900 Heidelberg, Tel. 06221 - 543245, Fax: 06221 - 543242, E-mail: C87@DHDURZ1.BITNET

5. *Implementierungssprache:* fuer die Lexikonverwaltung C

6. *Zugrundeliegende Hardware / Betriebssystem(e):* Hardware: Sun workstations, UNIX; DOS Version bis ca 6/1992

7. *Beginn / Ende der Entwicklung:* fertiggestellt

8. *Projekt:* erstellt im ESPRIT Projekt „Translator's Workbench "

9. *Input / Eingabeschnittstellen:* als ASCII-File

11. *Aktueller Zustand:* fertig

12. *Abhängigkeiten / Anforderungen an andere Systeme:* Die ASCII Eingabe kann von anderen Programmen verarbeitet werden; normalerweise sollte man aber die speziellen PLAIN+ Lexikonverwaltungsprogramme benutzen.

13. *Dokumentation:* TWB Report 3/1992

14. *Verfügbarkeit:* sofort, Konditionen Verhandlungssache

15. *Wartbarkeit:* Die PLAIN+ Lexikonverwaltung schlie t komfortable Korrektur- und Erweiterungsroutinen ein.

16. *Fremdinstallationen:* bisher nur im TWB Projekt und in anderen Forschungsprojekten geplant

17. *Benutzte Techniken und zugrundeliegende Theorien:* Formalismus der Dependentielle Unifikationsgrammatik;

18. *Allgemeine Beschreibung:* Das Lexikon einer DUG besteht aus zwei Teilen: (i) dem morpho-syntaktischen Lexikon und (ii) dem Valenzlexikon.

    (i) Ersteres bildet Wortformen auf Lexeme und grammatische Kategorien ab. Die Wortformen sind in Staeme und Endungen aufgeteilt, wobei Zeiger von den Staemmen auf die entsprechenden Endungsparadigmen verweisen. (Es ist also ein Wortformenlexikon, das aber als Stammformenlexikon gespeichert ist. Intern bildet das

gesamte Lexikon ein pbergangsnetzwerk. Dies ist die Voraussetzung dafuer, dass Komposita erkannt und zerlegt werden, obwohl nur die einzelnen Bestandteile im Lexikon stehen.)

(ii) Das Valenzlexikon ordnet den Lexemen je einen oder mehrere Valenzrahmen zu. Jeder Valenzrahmen besteht aus Verweisen auf sog. Templates, d.s. frame- hnliche Konstrukte, welche die Erg nzungsf higkeit der einzelnen Woerter beschreiben.

19. *Entwicklungsumgebung zur Erstellung der Einträge:* Das Verwaltungsprogramm fuer den morpho- syntaktischen Teil des Lexikons erlaubt Abdrucke in verschiedensten Formen und bietet vor allem eine sehr leichte Erweiterbarkeit: Wenn ein Wort fehlt, braucht der Benutzer nur Stammformen einzugeben, z.B. DER BENUTZER BENUTZERS BENUTZER oder GEBEN GIBST GABST GEGEBEN. Das System erkennt selbst die Staeme sowie die dazugehoerige Wortklasse und die Endungsparadigmata (Flexionsklassen). Das Updaten des Lexikons ist also ohne weiteres von Laien zu bewaeltigen. Fuer das Valenzlexikon gibt es eine spezielle Entwicklungssoftware (IVAN) auf der Basis von DBASE. Dabei werdem dem Benutzer die moeglichen Templates zu der durch (i) ermittelten Wortklasse zur Auswahl gegeben. Er braucht die zutreffenden nur anzuklicken. IVAN erzeugt anschlie end das endgueltige Eingabeformat fuer PLAIN+. Die Software ist natuerlich unabhaengig von der Sprache, fuer die das Lexikon erstellt wird. Wir haben Erfahrung mit anderen Sprachen. Bis Ende 1992 werden wir wahrscheinlich im Folgeprojekt von TWB ein franzoesisches und englisches Lexikon mit derselben Technologie fertigstellen.

20. *Anzahl und Art der Einträge:* 13.000 Lemmata; diese wurden zusammengestelt (1.) durch komplette Uebernahme verschiedener Woerterbuecher ueber "Basic German"(u.a. Mattuat, Plikat) und (2.) durch Aufnahme saemtlicher Vokabeln, welche im LIMAS Corpus (ca. 1 Millionen Woerter) mehr als einmal vorkamen. Die zugehoerige Morphologie ist komplett und deckt jede moegliche Wortform ab. Bei der Valenzkodierung haben unsere Mitarbeiter zu diesem Vokabular das gro e Duden-Woerterbuch sowie den dtv-Wahrig zu Rate gezogen.

Literaturangaben: Report auf Anfrage

3.4 Grammatik/Lexikon fuer DISCO

1. *Name der Komponente:* Grammatik/Lexikon fuer DISCO
2. *Zweck der Komponente:* Grammatik/Lexikon fuer Dialog-Sys tem
3. *Entwickelt und implementiert durch:* J. Nerbonne, K. Netter
4. *Anschrift, Kontaktperson, e-mail:* Klaus Netter, DFKI GmbH, Stuhlsatzenhausweg 3, 6600 Saarbruecken 11
5. *Implementierungssprache:* TDL
6. *Zugrundeliegende Hardware / Betriebssystem(e):* SUN, Mac / Unix
7. *Beginn / Ende der Entwicklung:* 1990 - 1993
8. *Projekt:* DISCO
9. *Input / Eingabeschnittstellen:*

   Input Analyse: Liste von Feature-Strukturen mit Resultat von morphologischer Analyse

   Input Synthese: Semantische Repraesentationen
10. *Output / Ausgabeschnittstellen:* Semantische Repraesentationen / Deutsch

    Output Analyse: Semantische Repraesentationen

    Output Synthese: Liste von Feature-Strukturen als Eingabe fuer morphologische Synthese
11. *Aktueller Zustand:* Entwicklung
12. *Abhängigkeiten / Anforderungen an andere Systeme:* Erforderliche Komponenten: - Morphologie (auch Vollformen moeglich) - Parser - Generator - TDL (fuer Modifikationen der Grammatik)
13. *Dokumentation:* Artikel
14. *Verfügbarkeit:* BMFT-Software
15. *Wartbarkeit:* wartbar
16. *Fremdinstallationen:* - Polygloss, Stuttgart - Eurotra-Prototyp, EG-Commission, Luxemburg
17. *Benutzte Techniken und zugrundeliegende Theorien:* - Typed Feature Logics - HPSG
18. *Allgemeine Beschreibung:* Grammatik mit kleiner bis mittlerer Fragmentabdeckung des Deutschen - NP-Morphosyntax (Determination und Adjektiv-Attribution) - Nominale Ellipsen - Satztypen (Fragetypen, Deklarativsaetze) - Adverbiale Adjunktion (PP, AP) - Praedikativkonstruktionen (NP, PP, AP) - usw.

19. *Entwicklungsumgebung zur Erstellung der Einträge:* TDL

20. *Anzahl und Art der Einträge:* 6 Regeln , ca. 80 Staemme (Lexikon), ca. 330 Typ-Definitionen

3.5 Hypertext-System fuer die Lexikographie

1. *Name der Komponente:* Hypertext-System fuer die Lexikographie
2. *Zweck der Komponente:* on-line Unterstuetzung bei der lexikographischen Datenerfassung sowie semantische Analyse von Lexemen
3. *Entwickelt und implementiert durch:* siehe 4.
4. *Anschrift, Kontaktperson, e-mail:* Frank Wegmann, Sprachwissenschaftliches Institut, Ruhr-Universitaet Bochum Postfach 10 21 48, W-4630 Bochum 1 wegmann@ruba.rz.ruhr-uni-bochum.de
5. *Implementierungssprache:* HyperTalk
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Hardware / OS ab Intel 80286, MS-DOS 3.3, F & A 3.0 (fruehere Datenbasis im NRW-Projekt) ab Macintosh Plus, MacOS ab System 6.05, HyperCard 2.x
7. *Beginn / Ende der Entwicklung:* 1990-
8. *Projekt:* EG: Multilex (ab 1991), NRW: Zur Entwicklung syntaktisch-semantischer Lexikonkomponenten in rechnergestuetzten Anwendungen
9. *Input / Eingabeschnittstellen:* Keyboard, Mouse, Import von DIF-Dateien (erzeugt von F & A auf Intel 80x86, MS-DOS)
10. *Output / Ausgabeschnittstellen:* Reporterstellung, Export von TAB-Dateien (Felder e. Records durch TAB getrennt)
11. *Aktueller Zustand:* z.B. Produkt, noch in Entwicklung) Forschungsprototyp, seit 1,5 Jahren in der Anwendung, teilweise noch in Entwicklung
12. Abhängigkeiten / Anforderungen an andere Systeme Mac mit Motorola 680x0, mindestens 1 MB RAM, MacOS System 6.05-6.08 und HyperCard 2.x, bei System 7.x empfiehlt sich HyperCard 2.1
13. *Dokumentation:* Integrierte Hilfe
14. *Verfügbarkeit:* Demo-Version frei, Vollprogramm mit Datenbestand nur gegen Rueck sprache mit Herrn Prof. Dr. Schnelle
15. *Wartbarkeit:* Service-Teil ermoeglicht automatische Reparaturen, falls Inkonsistenz vorliegt
16. *Fremdinstallationen:* SuperCard 1.6 (MacOS), HyberCube (NeXT), reine Datenbasis laesst sich in beliebige Datenbank auf jeder Plattform einbinden, die TAB-Format lesen koennen.

17. *Benutzte Techniken und zugrundeliegende Theorien:* Nutzung von Message Passing in HyperCard; teilweise Realisierung von Hypertext- Konzepten bei der Analyse von Lexemen; Index zur schnellen Suche, Manipulation von Lexemen und Reporterstellung sowie zur Konsistenzueberwachung

18. *Allgemeine Beschreibung:* Das in HyperCard verwendete lexikalische Format soll eine optimale Online-Schnittstelle fuer die Lexikographie (insbes. in der Semantik) bieten. Ausserdem erfolgt eine (z.Zt. noch manuelle) semantische Reanalyse der Definitionen einer Lesart.

19. *Entwicklungsumgebung zur Erstellung der Einträge:* F&A 3.0 auf PC-Basis (frueher), HyperCard 2.x auf Macintoshs

20. *Anzahl und Art der Einträge:* ca. 1100 Einträge f. Deutsch (wird z.Z. stark erweitert), je ca. 600 Einträge f. Englisch u. Franzoesisch Wortfelder: Woerter im Bereich des Zusammensetzens und Zerlegens, des Oeffnens und Schliessens, unspezifische Woerter

Literaturangaben: (unvollstaendig)

Doguraev, Bran / Briscoe, Ted (Hrsg.) (1989) "Computational Lexicography for Natural Language Processing". London: Longman.

Hausmann, F.J. et al. (1989) "Woerterbucher: Ein internationales Handbuch zur Lexikographie". Berlin u. New York, de Gruyter (Handbuecher zur Sprach- und Kommunikationswissenschaften)

Schnelle, H. (o.J.) "Zur semantischen und syntaktischen Struktur des Lexikons" Manuskript. Ruhr-Universitaet Bochum.

Schnelle, H. (o.J.) "Natural Language Knowledge Bases for Language Industry" Manuskript. Ruhr-Universitaet Bochum.

Schwarze, Christoph / Wunderlich, Dieter (Hrsg.) (1985) "Handbuch der Lexikologie". Koenigstein/Ts, Athenaeum

3.6 BONNLEX

1. *Name der Komponente:* BONNLEX
2. *Zweck der Komponente:* Lexikalische Datenbank als Quelle fuer natuerlichsprachliche Systeme
3. *Entwickelt und implementiert durch:* Inst. f. Kommunikationsforschung und Phonetik (Leitung: Lenders)
4. *Anschrift, Kontaktperson, e-mail:* Prof.Dr. Winfried Lenders, IKP Bonn, UPK013@ ibm.rhrz.uni-bonn.de
5. *Implementierungssprache:* entfaellt, da Lexikon
6. Zugrundliegende Hardware/Betriebssystem(e): TSO/CMS, zum Teil DOS
7. *Beginn / Ende der Entwicklung:* ca. 1984-87
8. *Projekt:* Aufbau einer Wortdatenbank fuer das Deutsche (BMFT- Projekt)
9. *Input / Einagbeschnittstellen:* Zugriffssystem ALEXYS
10. *Output / Ausagbeschnittstellen:* Zugriffssystem ALEXYS
11. *Aktueller Zustand:* Forschungsergebnis; Produkt
12. *Abhängigkeiten / Anforderungen an andere Systeme:* Anforderungen an andere Systeme: urspruenglich unter TSO; Migration auf DOS und UNIX in Arbeit (vgl. die Beschreibungen von Alex_Pc)
13. *Dokumentation:* Arbeitsbericht 8 des IKP
14. *Verfügbarkeit:* wird durch IKS (Inst. fuer Angewandte Kommunikations- und Sprachforschung) vertrieben; sollte im Rahmen von Kooperationen weiterentwickelt werden.
15. *Wartbarkeit:* im Rahmen von ALEX_PC gegeben
16. *Fremdinstallationen:* entfaellt
17. *Benutzte Techniken und zugrundeliegende Theorien:* Datenbank ist moeglichst theorieneutral angelegt. Kumulativer Aufbau aus vorhandenen maschinellen Woerterbuechern.
18. *Allgemeine Beschreibung:* ca. 300000 Einträge, voll morphologisch, z.T. syntaktische und semantisch klassifiziert.
19. *Entwicklungsumgebung zur Erstellung der Einträge:* frueher Zugriffssystem ALEXYS, jetzt: Bildschirm- und Datenorientierte Schnittstelle Alex_PC fuer DOS und UNIX.

Literaturangaben (Auswahl):

Hornung, Erich: Entwurf und Implementierung einer betriebssystemunabhaengigen Benutzeroberflaeche fuer die Verwaltung lexikalischer Informationen. IKP-Arbeitsbericht Nr. 8, Bonn 1990.

Winter, Andreas: Entwurf und Implementierung eines abstrakten, dynamisch modifizierbaren Datenmodells fuer die Speicherung und Wiedergewinnung lexikalischer Information. IKP-Arbeitsbericht Nr. 8, Bonn 1990.

Lenders, Winfried (Hrsg.): Studien zur Maschinellen Lexikographie. IKP-Arbeitsbericht Nr. 11, Bonn 1991 (darin Beschreibung des Aufbaus von BONNLEX)

3.7 ALEX_Pc

1. *Name der Komponente:* ALEX_Pc
2. *Zweck der Komponente:* Bildschirmorientiertes Zugriffssys tem fuer die Bonner Wortdatenbank (BONNLEX)
3. *Entwickelt und implementiert durch:* Inst. f. Kommunikationsforschung und Phonetik (Leitung: Lenders)
4. *Anschrift, Kontaktperson, e-mail:* Prof.Dr. Winfried Lenders, IKP Bonn, UPK013@ ibm.rhrz.uni-bonn.de
5. *Implementierungssprache:* C
6. *Zugrundliegende Hardware/Betriebssystem(e):* DOS, UNIX
7. *Beginn / Ende der Entwicklung:* 1989-1992
8. *Projekt:* Entwicklung im Rahmen von Diplomarbeiten
9. *Input / Einagbeschnittstellen:* ASCII-Files aus BONNLEX
10. *Output / Ausagbeschnittstellen:* spezielles Exportformat, menugesteuert
11. *Aktueller Zustand:* Forschungsergebnis; Produkt
12. *Abhängigkeiten / Anforderungen an andere Systeme:* Anforderungen an andere Systeme: verfuegbar unter DOS 1 MB, UNIX-Version nur fuer morphologische Markierungen verfuegbar
13. *Dokumentation:* in Vorbereitung
14. *Verfügbarkeit:* bisher nur fuer Demo-Zwecke; zur Benutzung der samten Wortdatenbank ist sind umfangreiche Importfunktionen zu erledigen.
15. *Wartbarkeit:* aufgrund modularen Aufbaus gegeben
16. *Fremdinstallationen:* entfaellt
17. *Benutzte Techniken und zugrundeliegende Theorien:* entfaellt
18. *Allgemeine Beschreibung:* Bildschirmorientierte Bearbeitung der morpho-syntaktischen Angaben, Verwaltung der syntaktisch-semantischen Angaben der Wortdatenbank BONNLEX; Import neuer Daten, Export mit interativ verwaltbarem Formatkatalog, allgemeine und benutzerfreundliche Oberflaeche
19. *Entwicklungsumgebung zur Erstellung der Einträge:* PC (286/486), DOS

Literaturangaben (zusaetzlich zu den in Fragebogen BONNLEX gegebenen:

Holzmann, Ralf: Verwaltung und Verteilung lexikalischer Information in einem interaktiv modifizierbaren Format mit Hilfe einer Wortdatenbank. Diplomarbeit (Informatik/Computerlinguistik), Universität Bonn, 1992. Veröffentlichung als IKP-Arbeitsbericht wird vorbereitet.

3.8 LEU/2 Lexikonmanager

1. *Name der Komponente:* LEU/2 Lexikonmanager

2. *Zweck der Komponente:* Der Lexikonmanager bidet die lexikalische Komponente der "LILOG Experimentier Umgebung"(LEU/2). DIese Komponente stellt die lexikalische Information für die Analyse und Generierung natürlicher Sprache zur Verfügung. Das System ist dabei in der Lage, eine Menge interaktiver Strategien zur Behandlung unbekannter Wörter anzuwenden.

3. *Entwickelt und implementiert durch:* Dr. Werner Emde, Andrea Meyering, Wilfried Hötker

4. *Anschrift, Kontaktperson, e-mail:* Wilfried Hötker, Universität Osnabrück, Fachbereich Sprach- und Literaturwissenschaft, Arbeitsbereich Computerlinguistik und Künstliche Intelligenz, Postfach 4469, 4500 Osnabrück, Tel: 0541 - 969 2585, bitnet: hoetker@dosunil, Fax: 0541 - 969 2500

5. *Implementierungssprache:* Quintus Prolog Version 2.4 und 3.0

6. *Zugrundeliegende Hardware / Betriebssystem(e):* IBM PS/2, IBM RS/600, AIX Version 1.1

7. *Beginn / Ende der Entwicklung:* Januar 1990 – November 1992

8. *Projekt:* IBM Deutschland GmbH, Projekt LILOG

9. *Input / Eingabeschnittstellen:* siehe Punkt 18 (*Allgemeine Beschreibung:*

10. *Output / Ausgabeschnittstellen:* siehe Punkt 18 (*Allgemeine Beschreibung:*

11. *Aktueller Zustand:* Der Lexikonmanager ist voll integriert in LEU/2

13. *Dokumentation:* Die einzelnen Module sind sehr gut dokumentiert. Die Funktionsweise des Managers ist in [1] beschrieben.

14. *Verfügbarkeit:* IBM Deutschland GmbH, Wissenschaftliches Zentrum, IWBS

17. *Benutzte Techniken und zugrundeliegende Theorien:* siehe [2,3]

18. *Allgemeine Beschreibung:* Wird der Lexikonmanager von derGenerierungskomponente aufgerufen, erhält er als Input die Grundform eines Wortes. Die Two-Level Morphology [4] berechnet dann die flektirte Wortform. Diese Form übergibt der Lexikonmanager als Output wieder an die Generierungskomponente.

    Bei der Analyse natürlichsprachlicher Sätze wird der Lexikonmanager durch den Parser aufgerufen und liefert die lexikalische Information der Wörter des Eingabesatzes. Die Defaultstrategien zur Gewinnung lexikalischer Informationen sind dabei die folgenden:

- Zuerst wird vom Lexikonmanager überprüft, ob es sich bei dem Input um eine Zahl handelt. In diesem Fall konstruiert der Lexikonmanager die notwendigen lexikalischen Informationen.
- Sofern es sich bei dem Eintrag um keine Zahl handelt, wird überprüft, ob das gesuchte Wort im zugrunde gelegten Vollformenlexikon gespeichert ist.
- Ist das Wort nicht im Vollformenlexikon gespeichert, wird eine morphologische Analyse durch die Morphologiekomponente von LEU/2 (Two-Level Morphology)durchgeführt und der entsprechende Eintrag aus dem Grundformenlexikon geholt. Dieser Eintrag wird um die Ergebnisse der morphologischen Analyse ergänzt.

Sofern keine dieser Strategien erfolgreich ist, wird eine Analyse des unbekannten Wortes durchgeführt, um doch och die benötigten Informationen zu erlangen. Dabei kommen folgende Strategien zum Einsatz:

- Zugriff auf das temporäre Lexikon, das die vom Lexikonmanager konstruierten Lexikoneinträge eines Textverstehensprozesses enthält.
- Zugriff auf ein natürlichsprachliches Synonymlexikon.
- Analyse von Nominalkomposita.
- Derivation der Bedeutung von Nomen unter Verwendung bedeutungsnaher Verben.
- Anfrage an den Benutzer.

Literaturangaben:

[1] Werner Emde. Managing Lexical Knowledge in LEU/2. Otthein Herzog, Claus Rainer Rollinger (eds.). Text Understanding in LILOG: Integrating Computational Linguistics and Artificial Intelligence, Springer Verlag, Berlin, 1991.

[2] Petra Ludewig. Incremental Vocabulary Extensions in Textunderstanding Systems. Otthein Herzog, Claus Rainer Rollinger (eds.). Text Understanding in LILOG: Integrating Computational Linguistics and Artificial Intelligence, Springer Verlag, Berlin, 1991.

[3] Petra Ludewig. Inkrementelle wörterbuchbasierte Wortschatzerweiterungen in sprachverarbeitenden Systemen - Entwurf einer konstruktiven Lexikonkonzeption. Dissertationsschrift an der Universität Osnabrück, 1992.

[4] Anne Schiller, Petra Steffens. Morphological Processing in the Two-Level Paradigm. Otthein Herzog, Claus Rainer Rollinger (eds.). Text Understanding in LILOG: Integrating Computational Linguistics and Artificial Intelligence, Springer Verlag, Berlin, 1991.

3.9 x2morF (eXtended 2-level morphology with Filters)

1. *Name der Komponente:* x2morF (eXtended 2-level morphology with Filters)
2. *Zweck der Komponente:* Analyse und Generierung von Wortformen
3. *Entwickelt und implementiert durch:* H. Trost, R. Flassig, H. Pirker
4. *Anschrift, Kontaktperson, e-mail:* H. Pirker: pirker@dfki.uni- sb.de
5. *Implementierungssprache:* Allegro Common Lisp
6. *Zugrundeliegende Hardware / Betriebssystem(e):* MacIntosh System6; Portierung auf Sun-Unix in Vorbereitung.
7. *Beginn / Ende der Entwicklung:* ab ca. 1989 / ...
8. *Projekt:* DISCO
9. *Input / Eingabeschnittstellen:* Input bei Analyse: strings Input bei Generierung: (moeglicherweise unterspezifizierte) Merkmalsstrukturen
10. *Output / Ausgabeschnittstellen:* bei Analyse: Merkmalsstruktur mit den morphosyntaktischen Eigenschaften der Eingabeform bei Generierung: string
11. *Aktueller Zustand:* Lauffaehiger Prototyp existiert, an dem aber noch Modifikationen vorgenommen werden
12. *Abhängigkeiten / Anforderungen an andere Systeme:* Version1 - sog. "Wiener Unifikator"Version2 - UDINE (Unifikator des DISCO-Projektes.) In Zukunft auch Verwendung von TDL für die Erstellung des Lexikons
13. *Dokumentation:* lediglich interne Dokumentation (Draft) vorhanden sowie Papiere von H. Trost
14. *Verfügbarkeit:* nach Absprache
16. *Fremdinstallationen:* OEFAI (Prof.Trappl, Wien)
17. *Benutzte Techniken und zugrundeliegende Theorien:* - 2-Stufen-Morphologie (siehe u.a. Koskenniemi 1983) - Merkmalslogik
18. *Allgemeine Beschreibung:* x2morF ist eine modular aufgebaute, bidirektional arbeitende Morphologiekomponente auf Grundlage der 2-Stufen-Morphologie Es besteht aus einem 2-Stufen-Teil, in dem v.A. morphonologische Prozesse abgebildet werden, und einem Wortgrammatik-Teil, der die morphosyntaktischen Operationen durchfuehrt. 2-Stufen-Teil: Abweichend vom "Standardmodell"der 2-Stufen-Morphologie verfuegt x2morF ueber die Moeglichkeit, einzelne Regeln mit Filtern (arbitraere Merkmalsstrukturen) zu versehen. (Trost 1990). Damit wird eine klar definierte Kommunikation zwischen diesem Modul und der Wortgrammatik-Teil ermoeglicht.

Wortgrammatik-Teil: Eine Grammatik im HPSG-Stil, die die vom Teil-1 ausgegebenen Morph- Ketten parst (Analyse) bzw. zu eingegebenen Merkmalsstrukturen mgliche Morph-Kombinationen erzeugt. (Generierung)

19. *Entwicklungsumgebung zur Erstellung der Einträge:* Verwendung von TDL in der neuesten Version

20. *Anzahl und Art der Einträge:* Etwa 300 Morphe. Flexion wird voll abgedeckt, Derivation nur mit wenigen Beispielen.

Literaturangaben:

Theoretische Grundlagen:

u.a. Koskenniemi ,K. (1983): Two-Level Model for Morphological Analysis, IJCAI-83, Karlsruhe, BRD, 683-685.

Beschreibungen des Systems:

Trost, H. (1990): The application of two-level morphology to non- concatenative German morphology, COLING-90, Helsinki, Vol.II 371-376.

Trost, H. (1991): X2MorF: A Morphological Component Based on Two-Level Morphology, DFKI. Report No. RR-91-04.

3.10 Finite-State-Morphologie

1. *Name der Komponente:* Finite-State-Morphologie
2. *Zweck der Komponente:* Morphologische Analyse und Generierung
3. *Entwickelt und implementiert durch:* D. Paulus nach einer Anregung von M. Kay
4. *Anschrift, Kontaktperson, e-mail:* Prof. Dr. G. Görz Universität Erlangen-Nürnberg IMMD 8 — KI Am Weichselgarten 9 D-8520 ERLANGEN

   Tel. (091310 699-126; fax: -198 goerz@informatik.uni-erlangen.de
5. *Implementierungssprache:* TLC-LISP Portierung nach PC-Scheme ist in Arbeit
6. *Zugrundeliegende Hardware / Betriebssystem(e):* PC, IBM- kompatibel / DOS / TLC-LISP
7. *Beginn / Ende der Entwicklung:* 1986
8. *Projekt:* Morphologische Analyse deutscher Verben
9. *Input / Eingabeschnittstellen:* Terminal, Files
10. *Output / Ausgabeschnittstellen:* Terminal, Files
11. *Aktueller Zustand:* abgeschlossen
12. *Abhängigkeiten / Anforderungen an andere Systeme:* keine außer Systemumgebung (s.o.)
13. *Dokumentation:* Studienarbeit von D. Paulus
14. *Verfügbarkeit:* nach Rücksprache
15. *Wartbarkeit:* ja
16. *Fremdinstallationen:* keine
17. *Benutzte Techniken und zugrundeliegende Theorien:* einstufige morphologische Analyse und Generierung durch endliche Automaten mittels Zeichenkettenersetzung
18. *Allgemeine Beschreibung:* einstufige morphologische Analyse und Generierung durch endliche Automaten mittels Zeichenkettenersetzung

    Einzelheiten s. Literatur
19. *Entwicklungsumgebung zur Erstellung der Einträge:* keine
20. *Anzahl und Art der Einträge:* sämtliche unregelmäßigen deutschen Verben

Literaturangaben:

RRZE-IAB-259 (Regionales Rechenzentrum, Univ. Erlangen-Nürnberg) Paulus D Ein Programmpaket zur morphologischen Analyse Diplomarbeit (GWAI Eringerfeld, COLING Budapest) 11.1986

Paulus D.: Endliche Automaten zur Verbflexion und ein spezielles deutsches Verblexikon Proc. GWAI-87, Berlin: Springer (IFB 152), 340–344

Görz G., Paulus D.: A Finite-State Approach to German Verb Morphology. Proceedings of COLING-88, Budapest, 1988, 212–215

3.11 AMOS (= A MOrphosyntactical expert System)

1. *Name der Komponente:* AMOS (= A MOrphosyntactical expert System)
2. *Zweck der Komponente:* morphosyntaktische Analyse althebraeischer Texte
3. *Entwickelt und implementiert durch:* Dipl. Inform. Guenther Specht
4. *Anschrift, Kontaktperson, e-mail:* Guenther Specht, Institut fuer Informatik, TU Muenchen, 8000 Muenchen 80, specht@informatik.tu-muenchen.de
5. *Implementierungssprache:* (Spezifikation: Logiksprache auf deduktiver Datenbank) uebersetzt nach: Common Lisp Portiert nach: (Allegro CommonLisp, Austin Kyoto Common Lisp (AKCL), Vaxlisp Interlisp-D)
6. *Zugrundeliegende Hardware / Betriebssystem(e):* UNIX- Rechner (hier Sun4, auch: SUN3, VAX, ...)
7. *Beginn / Ende der Entwicklung:* 1987 - 1991, momentan: Wartung, Pflege, Weiterentwicklung
8. *Projekt:* Zusammenarbeit zwischen TU-Muenchen und Uni- Muenchen Inst. fuer Informatik und Inst fuer Assyriologie
9. *Input / Eingabeschnittstellen:* Relationen (jedes tupel der Relationen enthaelt die morphologische Analyse des Wortes. Diese wird Automatisch mithilfe des Systems SALOMO aus dem fortlaufenden Text erstellt.
10. *Output / Ausgabeschnittstellen:* Relationen (je komplexer Phrase eine)
11. *Aktueller Zustand:* im Einsatz, Die 5 Buecher Mose sind fertig analysiert, die anderen Teile des AT folgen.
12. *Abhängigkeiten / Anforderungen an andere Systeme:* Mindestens 16 MByte Hautspeicherausbau
13. *Dokumentation:* verfuegbar, als Buch (siehe unten)
14. *Verfügbarkeit:* Copyright und Urheberrechte und Verwertungsrecht liegt bei uns; Nutzungsrecht kann von nichtkomerziellen Institutionen nach Vorlage der Kaufbelege des entspr. Buecher (Doku und Eingabe) unentgeltlich erworben werden.
15. *Wartbarkeit:* gut Wartbar, da vollstaendig als Logikprogramm spezifiziert
16. *Fremdinstallationen:* LMU Muenchen, Uni Tuebingen, (5 weiterte sind angefragt, u.A. in Basel)

17. *Benutzte Techniken und zugrundeliegende Theorien:* Analyse als Query an eine deduktive Datenbank, die das Logikprogramm bottom-up auswertet, d.h. set-at-a-time (ungleich Prolog!) Linksrekursionen sind zugelassen (ungleich Prolog!) Sehr hohe Performanz auch bei vielen Fakten (ungleich Prolog!) D.h. es werden alle verbindungen eines Kapitels gleichzeitig berechent, (nicht Satzweise wie in allen anderen Theorien)

18. *Allgemeine Beschreibung:* gut einsetzbar, vorgestellt auf SYSTEMS, GWAI, IBM-Kongress, Althebraistischer Kongress etc.

19. *Entwicklungsumgebung zur Erstellung der Einträge:* Sys tem SALOMO Eingabe Althebraeischer Text in Ascii-Transcription Ausgabe: Relationen fuer AMOS

20. *Anzahl und Art der Einträge:* Fuer jedes Wort des Textes ein Eintrag in der Faktenbasis (hier: Deduktiven Datenbank)

Literaturangaben:

Das AMOS Buch:

Specht, G.: Wissensbasierte Analyse althebraeischer Morphosyntax; Das Expertensystem AMOS, EOS-Verlag, St.Ottilien, 1990

Das SALOMO Buch:

Eckardt W.: Computergestuetze Analyse althebraeischer Texte; Algorithmische Erkennung der Morphologie EOS-Verlag, St.Ottilien, 1987

weiterfuehrende Literratur:

Specht G.: AMOS - Ein Expertensystem zur morphosyntaktischen Analyse von Althebraeisch, Proc. Wissenschaftliches Forum '88, Informationsverarbeitung in Lehre und Forschung, IBM, Muenchen, 1988, pp.63-64

Specht G.: Salomo - Ein System zur morphologischen Analyse althebraeischer Texte, Rundbrief des Fachausschusses 1.2 Kuenstliche Intelligenz und Mustererkennung in der Gesellschaft fuer Informatik, St. Augustin Muenchen, 1987 (2) pp.83-84

Eckardt W., Specht G.: SALOMO (Version 2.1), System-Dokumentation 1989, Institut fuer Assyriologie und Hethitologie der Ludwig Maximilian Universitaet Muenchen, Abt. Ugaristik und Hebraistik, Geschwister-Scholl-Platz 1, D-8000 Muenchen 22.

Freitag B., Specht G.: A Parsing System based on a Deductive Database, Proc. of 13th German Workshop on Artificial Intelligence (GWAI '89), Springer, 1989, pp.280-289

3.12 MORPHIX-3

1. *Name der Komponente:* MORPHIX-3
2. *Zweck der Komponente:* Flexionsanalyse und -synthese
3. *Entwickelt und implementiert durch:* Wolfgang Finkler und Guenter Neumann
4. *Anschrift, Kontaktperson, e-mail:* am DFKI: finkler@dfki.uni- sb.de, neumann@dfki.uni-sb.de
5. *Implementierungssprache:* Common LISP
6. *Zugrundeliegende Hardware / Betriebssystem(e):* ueberall lauffaehig, wo Common LISP installiert ist
7. *Beginn / Ende der Entwicklung:* 1986 Prototype, 1988 Morphix-2, 1991 Morphix-3
8. *Projekt:* WIP
9. *Input / Eingabeschnittstellen:* Wort als string, Satz als string- liste, normale Ascii Textdatei
10. *Output / Ausgabeschnittstellen:* Listenform (BNF) mit Stamm und alle moeglichen Flexionsinformationen der aktuellen Wortform
11. *Aktueller Zustand:* fertig, an etliche Institute verteilt
12. *Abhängigkeiten / Anforderungen an andere Systeme:* keine
13. *Dokumentation:* Manual beiliegend
14. *Verfügbarkeit:* Common LISP Version umsonst C Version gegen Bezahlung (in Entwicklung)
16. *Fremdinstallationen:* ja
17. *Benutzte Techniken und zugrundeliegende Theorien:* klassifikationsbasierter Ansatz
18. *Allgemeine Beschreibung:* Informatik Fachberichte Nr. 176, Springer Verlag: 4. Oegai, 1988
19. *Entwicklungsumgebung zur Erstellung der Einträge:* Klaerungsdialog unabhaengig von Windowsystemen
20. *Anzahl und Art der Einträge:* ca. 6000 Staemme mit Klassifikationsinformation

4 Analyse natürlicher Sprache: Grammatiken, Parsing und Entwicklungsumgebungen

4.1 PLAIN-D Grammatik (Templates)

1. *Name der Komponente:* PLAIN-D Grammatik (Templates)
2. *Zweck der Komponente:* Grundlage fuer den PLAIN+ Parser
3. *Entwickelt und implementiert durch:* LCL, Uni Heidelberg
4. *Anschrift, Kontaktperson, e-mail:* Prof. P. Hellwig, Lehrstuhl fuer Computerlinguistik, Universitaet Heidelberg, Karlstr. 2, 6900 Heidelberg, Tel. 06221 - 543245, Fax: 06221 - 543242, E-mail: C87@DHDURZ1.BITNET
5. *Implementierungssprache:* fuer die Lexikonverwaltung C
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Sun workstations, UNIX; DOS Version angestrebt
7. *Beginn / Ende der Entwicklung:* Version 2 fertiggestellt; umfangreiche Revision und Anpassung an den neuen Parser bis 3/1994
8. *Projekt:* Forschung am Lehrstuhl
9. *Input / Eingabeschnittstellen:* als ASCII-File
11. *Aktueller Zustand:* alte Version fertig; milestones fuer Revision 3/1993 (Kern) und 3/1994 (vollstaendig).
12. *Abhängigkeiten / Anforderungen an andere Systeme:* die Beschreibungen koennen vom menschlichen Leser verstanden und von anderen Programmen ausgewertet werden; normalerweise gehoert aber der PLAIN+ Parser dazu.
13. *Dokumentation:* Interner Report „Bausteine des Deutschen “
14. *Verfügbarkeit:* erster Teil der neuen Version 3/1993 fuer wahrscheinlich ca 90Texten; vollstaendig 3/1994; Konditionen Verhandlungssache
15. *Wartbarkeit:* Die PLAIN+ Lexikonverwaltung schlie t komfortable Korrektur- und Erweiterungsroutinen ein.
16. *Fremdinstallationen:* bisher nur im TWB Projekt und in anderen Forschungsprojekten geplant
17. *Benutzte Techniken und zugrundeliegende Theorien:* Formalismus der Dependentielle Unifikationsgrammatik

18. *Allgemeine Beschreibung:*

    Die Dependentielle Unifikationsgrammatik ist theoretisch vollstaendig lexikalistisch; d.h. die Syntax der Sprache ergibt sich aus der Kombinationsfaehigkeit der einzelnen Woerter. Der Parser vergleicht parallel die zahlreichen Kombinationsangaben der einzelnen Woerter und bildet ggf. groessere Segmente bis hin zur vollstaendigen Analyse. Die Kombinationsfaehigkeit eines Wortes wird konkret dadurch beschrieben, das ihm eine Reihe von sog. Templates zugeordnet werden (siehe Valenzlexikon). Die Templates enthalten sog. Slots, in die ein Segment aus dem Kontext hineinpa t. Slots koennen ganz fein restringiert werden mit Hilfe von Kategorien und selektiven *lexikalischen Merkmalen. Obwohl es moeglich waere, im Lexikon jedes Template bei* jedem Wort neu aufzufuehren (sodass es dann gar keine Grammatik gaebe), kann man die Templates insofern als die Grammatik ansehen, da sie in ihrem Effekt den Regeln in regel-basierten Grammatiken entsprechen. Die Komponente enthaelt eben diese Templates/Regeln fuer die Syntax der deutschen Sprache.

19. *Entwicklungsumgebung zur Erstellung der Einträge:* Das eigentliche Entwicklungs-Tool ist der PLAIN+ Parser. Mit ihm kann man testen, ob die Grammatik korrekt ist, denn der Parser gibt fuer den Linguisten verschiedene Protokolle aus, aus denen hervorgeht, warum eine Analyse fehlgeschlagen ist. Diese Ausgaben werden benutzt, um die Templates zu verbessern.

20. *Anzahl und Art der Einträge:* ca. 600 Templates (entspricht ca 600 Regeln). Die bisher vorliegende Version deckt die grundlegenden valenz-basierten Bestandteile von Saetzen des Deutschen ab (Satzglieder, Nebensaetze, Hilfsverbgefuege usw.). In der revidierten Fassung werden auch Adverbiale, Koordination, Satzzeichen und Ellipsen beruecksichtigt sein. Die Weiterentwicklung wird vorgenommen auf der Basis unserer Beispieldatenbank fuer syntaktische Konstruktionen (zusammengestellt aus den wichtigtsen gedruckten Grammatiken des Deutschen), die ueber 5.000 Einträge hat.

Literaturangaben: Report auf Anfrage

4.2 Konnektionistischer Parser fuer kontextfreie Grammatiken

1. *Name der Komponente:* Konnektionistischer Parser fuer kontextfreie Grammatiken
2. *Zweck der Komponente:* Demonstration der Funktion und Arbeitsweise von konnektionistischen Parsern.
3. *Entwickelt und implementiert durch:* siehe 4.
4. *Anschrift, Kontaktperson, e-mail:* Rolf Wilkens, Sprachwissenschaftliches Institut, Ruhr-Universitaet Bochum Postfach 10 21 48, W-4630 Bochum 1 wilkens@ruba.rz.ruhr-uni-bochum.de
5. *Implementierungssprache:* C
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Hardware / OS Sun 4-110, Sun OS 4.1, SunView 1.8
7. *Beginn / Ende der Entwicklung:* 1990 - 1991
8. *Projekt:* DFG: Kognitive Linguistik
9. *Input / Eingabeschnittstellen:* stdin, Keyboard, Mouse
10. *Output / Ausgabeschnittstellen:* Graphikschirm
11. *Aktueller Zustand:* z.B. Produkt, noch in Entwicklung) fertiges Produkt
12. Abhängigkeiten / Anforderungen an andere Systeme SunView 1.8 kompatibilitaet
13. *Dokumentation:* man page, Magisterarbeit
14. *Verfügbarkeit:* – free –
15. *Wartbarkeit:* ....
16. *Fremdinstallationen:* makefile
17. *Benutzte Techniken und zugrundeliegende Theorien:* Chartparser nach Earley, Konnektionismus
18. *Allgemeine Beschreibung:* Das Programm besteht aus zwei Teilen: einem Compiler, der eine beliebige kontextfreie Grammatik in ein ein konnektionistisches Netz ueberfuehrt und einem Netzwerksimulator, der die Funktionsweise dieses Netzes demonstriert.

Literaturangaben:

Hoelter, M. (1990): Net-linguistic representation of complex feature structures, Ms. Ruhr-Universitaet Bochum

Schnelle, H. & Doust, R. (1989): Net-linguistic representation of complex feature structures, In: Reilly, R. & Sharkey, N. Connectionist approaches to languages. Vol. 1. Amsterdam: North Holland.

Wilkens, R. & Schnelle, H. (1990): A connectionist parser for context-free phrase structure grammars, In: Dorffner, G. (Hrsg) (1990) Konnektionismus in AI und Kognitionsforschung, 6. Oesterreichische AI Tagung (KONNAI) 1990, Berlin etc: Springer

Wilkens, R. (1990): Ein konnektionistischer Parser fuer kontextfreie Grammatiken, Magisterarbeit, Sprachwiss. Inst. Ruhr-Uni Bochum.

4.3 Konnektionistische Implementation eines Parsers zur Simulationen der syntaktischen Verarbeitung von Aphasiepatienten

1. *Name der Komponente:* vorlaeufiger Arbeitstitel: konnektionistische Implementation eines Parsers zur Simulationen der syntaktischen Verarbeitung von Aphasiepatienten.

2. *Zweck der Komponente:* Simulation der Verarbeitung von sprachlichen Aeusserungen bei Aphasiepatienten.

3. *Entwickelt und implementiert durch:* siehe 4.

4. *Anschrift, Kontaktperson, e-mail:* Rolf Wilkens, Sprachwissenschaftliches Institut, Ruhr-Universitaet Bochum Postfach 10 21 48, W-4630 Bochum 1 wilkens@ruba.rz.ruhr-uni-bochum.de

5. *Implementierungssprache:* C

6. *Zugrundeliegende Hardware / Betriebssystem(e):* Hardware / OS Sun 4/110, SunOS 4.1, XView 3.0, X11R5 PC 486 / MS-DOS 4.0

7. *Beginn / Ende der Entwicklung:* 1991 -

8. *Projekt:* DFG: Kognitive Linguistik

9. *Input / Eingabeschnittstellen:* stdin, Keyboard, Mouse

10. *Output / Ausgabeschnittstellen:* Graphikschirm. AVMs und Baeume koennen direkt nach TeX uebertragen werden.

11. *Aktueller Zustand:* in Entwicklung

12. *Abhängigkeiten / Anforderungen an andere Systeme:* X11R5, XView 3.0 Kompatibilitaet, SuperVGA (800x600x256) fuer MsDos Version

13. *Dokumentation:* man page, Dissertation (in Arbeit)

14. *Verfügbarkeit:* – free –

15. *Wartbarkeit:* ....

16. *Fremdinstallationen:* makefile

17. *Benutzte Techniken und zugrundeliegende Theorien:* Konnektionismus, GB, HPSG, Aphasieforschungen

18. *Allgemeine Beschreibung:* Das Programm simuliert die konnektionistische Verarbeitung von syntaktischem Wissen. Der Parser arbeitet mit mehreren Heuristiken wie sie in der Aphasieforschung entwickelt wurden. Damit kann das System zur Simulation von Sprachverahlten bei gestoerter Kompetenz eingesetzt werden. Eine entscheidende Frage, die in dem Projekt geklaert werden soll, ist wie die einzelnen Heuristiken zusammenarbeiten koennen. Der Parser soll also mit einem sehr geringen Aufwand (linear) bestimmte feste Muster erkennen. Wenn der Input jedoch zu komplex ist, muss auf eine anderes aufwendigeres Verfahren umgeschaltet werden.

Literaturangaben: (unvollstaendig)

Wilkens, R. (1991): Konnektionistische Repraesentation von grammatischem Wissen, In: Guesgen, H-W., Hoelldobler, S., Kurfess, F., Suttner, Ch. Massive Parallelitaet und Inferenz Forschungsbericht AIDA 91, 16; TU Darmstadt, Fachbereich Informatik

Wilkens, R. (1991): Representation of Principles and Parameters in a Connectionist Network, Ms. Ruhr-Universitaet Bochum.

4.4 ASL-Chart

1. *Name der Komponente:* ASL-Chart

2. *Zweck der Komponente:* Einsatz als Parser ueber Worthypothesen (und anderen Hypothesen) in einem stark interaktiven (parallelen) integrierten Speech-Language-System

3. *Entwickelt und implementiert durch:* Hans Weber, MA

4. *Anschrift, Kontaktperson, e-mail:* Hans Weber Universitaet Erlangen-Nuernberg IMMD 8 (ASL) Am Weichselgarten 9 8560 Erlangen-Tennenlohe weber@faui80.informatik.uni-erlangen.de

5. *Implementierungssprache:* Allegro Common Lisp 4.0.1 [Sun4] von FRANZ INC.

6. *Zugrundeliegende Hardware / Betriebssystem(e):* SPARCstation IPX mit 48 MB von Sun Microsystems unter SunOS Release 4.1.1-IPX

7. *Beginn / Ende der Entwicklung:* 1.1.1991 bis 31.12.1994

8. *Projekt:* Verbmobil-ASL-Nord, Projektpartner ASL-Nord-ER

9. *Input / Eingabeschnittstellen:* Worthypothesen (Bottom-Up) Hypothesen ueber Intonationsmerkmale (Bottom-Up) Erwartungen ueber Semantische Formen (Top-Down) Anforderung fuer Erwartungen ueber Worthypothesen

10. *Output / Ausgabeschnittstellen:* Hypothesen ueber Ausserungen, Semantische Formen (Bottom-Up) Erwartungen ueber Intonationsmerkmale Erwartungen ueber Worthypothesen

11. *Aktueller Zustand:* ASL-Chart Version 1.0 wird zum 1.8.1992 fertiggestellt sein.

12. *Abhängigkeiten / Anforderungen an andere Systeme:* ASL-Chart benoetigt: Eine Worthypothesen liefernde Komponente Eine semantische Formen interpretierende Komponente Optional: Eine Prosodische Komponente

13. *Dokumentation:* Arbeitsberichte in Verbmobil-ASL-Nord

14. *Verfügbarkeit:* Copyrights 1992 by Universitaet Erlangen-Nuernberg, IMMD8

    Zu Forschungszwecken kostenlos erhaeltlich

16. *Fremdinstallationen:* Ab 1.8.1992: Universitaet Hamburg FB Informatik, AB NatS Bodenstedtstrasse 16 2000 Hamburg 50

17. *Benutzte Techniken und zugrundeliegende Theorien:* Der Parser ist ein erweiterter Chartparser ueber Lattices fuer HPSG- aehnliche Grammatiken. Dem Entwurf zugrunde liegen: Chartparsingtechnologie, Graphsuchverfahren / Spracherkennung, Unifikationsformalismen (HPSG), Inkrementelle Verarbeitung, Parallelisierung, Forschung zur Syntax-Intonation.

18. *Allgemeine Beschreibung:* Siehe: Hans Weber: "Chartparsing in ASL-Nord", erscheint als Technischer Report des Projektes VERBMOBIL-ASL-Nord ASL- TR-28-92/UER, Universitaet Erlangen-Nuernberg, IMMD8

4.5 MediTAS 1.0

1. *Name der Komponente:* MediTAS 1.0
2. *Zweck der Komponente:* Syntax-Analyse von Sätzen (in deutscher Sprache) aus dem klinischen Routinebetrieb
3. *Entwickelt und implementiert durch:* Priv. Doz. Dr. P. M. Pietrzyk, Abt. Medizinische Informatik
4. *Anschrift, Kontaktperson, e-mail:* Robert-Koch-Str. 40, 3400 Göttingen, minfo@serversun.mdv.gwdg.de
5. *Implementierungssprache:* Lisp (PC-Scheme)
6. *Zugrundeliegende Hardware / Betriebssystem(e):* IBM- (kompatibel) PC (386), MS-DOS
7. *Beginn / Ende der Entwicklung:* 1986/1990
8. *Projekt:* MediTAS (Medizinisches Text-Analyse-System)
9. *Input / Eingabeschnittstellen:* Kategorisierte Wortformen gemäß LAG-Formalismus
10. *Output / Ausgabeschnittstellen:* LAG-Strukturen, Konstituenten- und Dependenzstrukturen
11. *Aktueller Zustand:* MediTAS 1.0: Prototyp, Rest von MediTAS: In Entwicklung
12. *Abhängigkeiten / Anforderungen an andere Systeme:* keine
13. *Dokumentation:* Auszugsweise beschrieben in angegebener Literatur
14. *Verfügbarkeit:* Z.Z nicht allgemein verfügbar
15. *Wartbarkeit:* -
16. *Fremdinstallationen:* keine
17. *Benutzte Techniken und zugrundeliegende Theorien:* Links- Assoziative-Grammatik (LAG) nach Hausser
18. *Allgemeine Beschreibung:* Erweiterung von DCAT (Hausser) und Anpassung an Textmaterial aus den Beurteilungsabschnitten zytopathologischer Befundberichte. Untersuchtes Textmaterial: 8790 unterschiedliche Textsätze.
19. *Entwicklungsumgebung zur Erstellung der Einträge:* Window- System
20. *Anzahl und Art der Einträge:* 3.760 Kategorisierte Wortformen (Deutsch)

Literaturangaben:

Pietrzyk PM. Survey of the Goettingen medical text analysis system. In: Hansen R. Solheim BG, O' Moore RR, Roger FH, eds. Medical Informatics Europe '88. Berlin: Springer Verlag, 1988: 128-32.

Pietrzyk PM. Analyse medizinischer Freitexte: Strukturierung und Syntaxanalyse. Habilitationsschrift. Göttingen: Georg-August-Universität, 1989.

Pietrzyk PM. A Medical Text Analysis System for Germna - Syntax Analysis. Methods of Information in Medicine 1991; 30:275-283.

4.6 Parsing im Project TOPIC

1. *Name der Komponente:*
2. *Zweck der Komponente:* Parsing
3. *Entwickelt und implementiert durch:* Udo Hahn
4. *Anschrift, Kontaktperson, e-mail:* Prof. Dr. U. Hahn, Univ. Freiburg, AG Linguistische Informatik, Friedrichstr. 50, 7800 Freiburg, hahn@supreme.coling.uni-freiburg.de
5. *Implementierungssprache:* C
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Sun-3, Sun-4 / Unix
7. *Beginn / Ende der Entwicklung:* 1982 – 1986
8. *Projekt:* TOPIC (Text-Oriented Procedures for Information Managment and Condensation of Expository Texts)
9. *Input / Eingabeschnittstellen:* beliebige Textdatei (Ascii)
10. *Output / Ausgabeschnittstellen:* semant. Repräsentationsstrukturen (Frames)
11. *Aktueller Zustand:* lauffähiger Prototyp
12. *Abhängigkeiten / Anforderungen an andere Systeme:* verlangt Existenz eines frame-basierten (hybriden) Wissensrepräsentationssystem, das bzgl. einer bel. Domäne instantiiert ist.
13. *Dokumentation:* vorhanden (Header in Sourcecode-Files)
14. *Verfügbarkeit:* für akademische Zwecke verfügbar
15. *Wartbarkeit:* keine Erfahrungswerte außerhalb der Entwicklungsgruppe
16. *Fremdinstallationen:* Univ. Konstanz, Suiss Life (Zürich)
17. *Benutzte Techniken und zugrundeliegende Theorien:* semanitscher Parser, Wortexpertenmodell
18. *Allgemeine Beschreibung:* lexikalisch-orientierter, semantischer, verteilter Parser; durch Schlüsselwörter getriggert, partielles Parsing; besonders adaptiert an Textphänomene (Anaphern, Ellipse, Kohärenzmuster).
19. *Entwicklungsumgebung zur Erstellung der Einträge:* Wortexperten-Editor
20. *Anzahl und Art der Einträge:* ca. 20 Prototypen-Netze, die über Matching mit Weltwissen instantiiert werden.

Literaturangaben:

U. Hahn Lexikalisch verteiltes Textparsing. Eine objektorientierte Spezifikation eines Wortexpertensystems auf der Grundlage des Aktorenmodells. Berlin etc.: Springer, 1990, 263pp. (Informatik-Fachberichte 243 - Subreihe Künstliche Intelligenz).

U. Hahn Making Understanders out of Parsers: Semantically Driven Parsing as a Key Concept for Realistic Text Understanding Applications. In: International Journal of Intelligent Systems, Vol.4, 1989, No.3, pp. 345-393.

U. Hahn Topic Parsing: Accounting for Text Macro Structures in Full-Text Analysis. In. Information Processing & Managment, Vol. 26, 1990, No. 1, pp 135- 170 (Special Issue on Natural Language Processing and Information Retrieval)

U. Hahn, A Generalized Word Expert Model of Lexically Distributed Text Parsing. In: B. du Boulay, D. Hogg, L. Steels (EDs.), Advances in Artificial Intelligence - II. 7th European Conference an Artificial Intelligence - ECAI-96. Brighton, U. K., July 20-25, 1986. Amsterdam etc.: North-Holland, 1987, pp. 417-425.

4.7 SCAN

1. *Name der Komponente:* SCAN

2. *Zweck der Komponente:* konnektionistische und hybride (symbolisch/konnektionistische) Verarbeitung von natuerlicher Sprache, insbesondere Verarbeitung von syntaktischen und semantischen konnektionistischen Repraesentationen fuer strukturelle Disambiguierung und semantische Klassifizierung von Nominalphrasen

3. *Entwickelt und implementiert durch:* Stefan Wermter

4. *Anschrift, Kontaktperson, e-mail:* Stefan Wermter Universtiaet Hamburg Fachbereich Informatik Bodenstedtstr. 16 2000 Hamburg 50 wermter@nats4.informatik.uni-hamburg.de

5. *Implementierungssprache:* CommonLisp, Planet

6. *Zugrundeliegende Hardware / Betriebssystem(e):* ExplorerII, System 6.0 Sun OS 4.1

11. *Aktueller Zustand:* noch in der Entwicklung

4.8 Parser fuer Grammatiken im Format von TDL/Udine

1. *Name der Komponente:* Parser fuer Grammatiken im Format von TDL/Udine
2. *Zweck der Komponente:* Analyse von Saetzen in natuerlicher Sprache auf der Grundlage einer in TDL/Udine erstellten Grammatik.
3. *Entwickelt und implementiert durch:* Bernd Kiefer
4. *Anschrift, Kontaktperson, e-mail:* Bernd Kiefer Deutsches Forschungszentrum fuer Kuenstliche Intelligenz Stuhlsatzenhausweg 3 6600 Saarbruecken email: kiefer@dfki.uni-sb.de
5. *Implementierungssprache:* Lisp (Franz Allegro Common Lisp 4.0)
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Common Lisp System
7. *Beginn / Ende der Entwicklung:* 1. Aug. 1991 - 1. Aug. 1992 (vorraussichtlich)
8. *Projekt:* DISCO
9. *Input / Eingabeschnittstellen:* erfordert eine Grammatik im TDL/Udine-Format, ein Anschluss an TDL wird dieses zu einem vollstaendigen Entwicklungssystem ergaenzen.
10. *Output / Ausgabeschnittstellen:* Merkmalstrukturen im Udine-Format, graphische Ausgabe der Chart.
11. *Aktueller Zustand:* in Entwicklung
12. *Abhängigkeiten / Anforderungen an andere Systeme:* erfordert den Unifikator Udine, ein Modul zum Anschluss an den Featureeditor sowie ein Modul zur graphischen Darstellung der chart sind vorhanden.
13. *Dokumentation:* in der Entstehung
14. *Verfügbarkeit:* Weitergabe fuer wissenschaftliche Zwecke
18. *Allgemeine Beschreibung:* Es handelt sich um einen active-chart Parser, wie er in (Erbach 1991) beschrieben ist (Nur ein Teil der dort beschriebenen Funktionalitaet ist implementiert). Er ist so modifiziert, dass er ohne explizites kontextfreies Backbone auskommt.

Literaturangaben:

(Erbach 1991): Erbach, Gregor: A Flexible Parser for a Linguistic Development Environment, Proceedings of IJCAI 1991.

4.9 Parser fuer GPS-Grammatiken

1. *Name der Komponente:* Parser fuer GPS-Grammatiken (konstruktive Version)
2. *Zweck der Komponente:* syntaktische Analyse von natuerlichsprachlichen Saetzen
3. *Entwickelt und implementiert durch:* Wilhelm Weisweber
4. *Anschrift, Kontaktperson, e-mail:* Wilhelm Weisweber TU Berlin Projekt KIT-FAST, Sekr. FR 5-12 Franklinstr. 28/29 W-1000 Berlin 10 ww@kit.cs.tu-berlin.de
5. *Implementierungssprache:* Quintus Prolog / Arity Prolog
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Sun / Unix oder AT/XT kompatibler PC / MS-DOS 3.31
7. *Beginn / Ende der Entwicklung:* Nov 1986 - Mai 1987
8. *Projekt:* Anapherninterpretation in der maschinellen Uebersetzung (KIT-FAST II, EUROTRA-D Begleitforschung Berlin)
9. *Input / Eingabeschnittstellen:* GPSG-Komponenten, d.h. - Merkmaldefinition - Aliases - ID-Regeln - LP-Aussagen - Metaregeln - FCRs natuerlichsprachlicher Satz
10. *Output / Ausgabeschnittstellen:* syntaktische Struktur des Satzes
11. *Aktueller Zustand:* fertig, von Zeit zu Zeit einige Adaptionen
12. *Abhängigkeiten / Anforderungen an andere Systeme:* keine, der Parser ist Komponente eines experimentellen MUe-Systems
13. *Dokumentation:* keine explizite Systemdokumentation, kurze Beschreibung der ausfuehrbaren Operationen
14. *Verfügbarkeit:* kostenlos, gegen Rueckmeldung
15. *Wartbarkeit:* Quellcode ist kommentiert
16. *Fremdinstallationen:* z.T. in Verbindung mit MUe-System - Universitaet Tuebingen, SNS - Universitaet Nijmegen/Niederlande, Research Group for Corpus Linguistics - JATE, Szeget/Ungarn - Universitaet Koblenz-Landau, FB Informatik, Computerlinguistik - Universitaet Duesseldorf, Seminar fuer allgemeine Sprachwissenschaft - University of Sussex/Grossbritannien, School of Cognitive Science - Universitaet Bielefeld, Fakultaet fuer Linguistik und Literaturwiss. - Universitaet Wien, Inst. fuer Medizin. Kybernetik und Artif. Intel. - Zentralinstitut fuer Sprachwissenschaft, Berlin - Universitaet Regensburg, Linguistische Informationswissenschaft - DFKI Saarbruecken
17. *Benutzte Techniken und zugrundeliegende Theorien:* konstruktive Version der GPSG

18. *Allgemeine Beschreibung:* Der Chart-Parser ist integriert in ein experi- mentelles Uebersetzungssystem. Er interpretiert direkt das ID/LP-Format und die Metaregeln des GPSG-Formalismus und arbeitet bottom-up.

Literaturangaben:

Christa Hauenschild, Stephan Busemann: A constructive Version of GPSG for machine translation, in: E. Steiner, P. Schmidt, C. Zellinsky-Wibbelt: From Syntax to Semantics - Insights from Machine Translation, Frances Pinter, London 1988, S. 216-238

Wilhelm Weisweber: Ein Dominanz-Chart-Parser fuer generalisierte Phrasenstrukturgrammatiken, KIT-Report 45, TU Berlin 1987

Wilhelm Weisweber, Susanne Preuss: Direct Parsing with Metarules, erscheint in den Procs. der Coling-92, Nantes 1992

4.10 PAULA Parsing Algorithm for UG-based Language

1. *Name der Komponente:* PAULA Parsing Algorithm for UG-based Language Analysis

2. *Zweck der Komponente:* Parsing und Syntaxanalyse natuerlicher Sprache

3. *Entwickelt und implementiert durch:* DFG-Projekt "PAULA"Prof. Dr. Sascha W. Felix Jan Olsen, Diplominformatiker (Univ.)

4. Anschrift Kontaktperson: Prof. Dr. Sascha W. Felix e-mail: felix@pille.phil.uni-passau.de Jan Olsen e-mail: olsen@pille.phil.uni-passau.de Universitaet Passau Lehrstuhl fuer Allgemeine Linguistik Innstrasse 40 8390 Passau

5. *Implementierungssprache:* Modula-2

6. *Zugrundeliegende Hardware / Betriebssystem(e):* SUN 3/50 SunOS 4.0 (UNIX)

7. *Beginn / Ende der Entwicklung:* Beginn: 12.87 Ende: —

9. *Input / Eingabeschnittstellen:* ASCII

10. *Output / Ausgabeschnittstellen:* ASCII oder Graphik/HPGL

11. *Aktueller Zustand:* In der Entwicklung

12. *Abhängigkeiten / Anforderungen an andere Systeme:* keine ausser der Graphik-Schnittstelle

13. *Dokumentation:* Zwischenberichte an die DFG, weitere Dokumentation auf Anfrage

15. *Wartbarkeit:* Das Modula-2 Modul-System wurde umfassend genutzt, daher einfache Wartung und Modifikation moeglich.

17. *Benutzte Techniken und zugrundeliegende Theorien:* Government and Binding (Chomsky 1981, 1986)

4.11 Syntaktische Analyse

1. *Name der Komponente:* Syntaktische Analyse
2. *Zweck der Komponente:* Identifizierung der korrekten akustischen Satzhypothese und Erzeugung eines syntaktischen Strukturbaumes
3. *Entwickelt und implementiert durch:* Siemens ZFE ST SN 74
4. *Anschrift, Kontaktperson, e-mail:* G.Niedermair, ZFE ST SN 74 Otto-Hahn-Ring 6 8000 Muenchen 83 tel: 089/ 636 2374 e-mail: nie
5. *Implementierungssprache:* TI-Common Lisp, Flavours
6. *Zugrundeliegende Hardware / Betriebssystem(e):* TI- Explorer
7. *Beginn / Ende der Entwicklung:* B: 1986 E: 1991
8. *Projekt:* SPICOS II, SUNDIAL
9. *Input / Eingabeschnittstellen:* Liste von akustisch bewerteten Satzhypothesen
10. *Output / Ausgabeschnittstellen:* 1-n Syntaxbaeume mit entsprechenden syntaktischen,semantischen Merkmalen
11. *Aktueller Zustand:* Entwicklung im SUNDIAL Projekt mit dem Demonstrator vorerst abgeschlossen.
12. *Abhängigkeiten / Anforderungen an andere Systeme:* abhaengig vom Lexikon
13. *Dokumentation:* BMFT - Abschlussbericht,
14. *Verfügbarkeit:* ?
15. *Wartbarkeit:* gering, da Implementatoren voruebergehend nicht verfuegbar
16. *Fremdinstallationen:* auf Micro explorer bei Philips und IPO Eindhoven
17. *Benutzte Techniken und zugrundeliegende Theorien:* Augmentierte Phrasenstrukturgrammatik in Verbindung mit einem Chartparser, der erweitert ist um Behandlung von Bewegungsregeln und Bounding-Nodes. Ein erweiterter Tomita-Parser fuer dieselbe Grammatik steht ebenfalls zur Verfuegung. Die Phrasenstruktur beschreibung ist angelehnt an die X-bar Syntax.
18. *Allgemeine Beschreibung:*

    Die Abdeckung beinhaltet komplexe NP, PP, Genitivstrukturen, Apposition, Relativsaetze, wh-Saetze, y-n-Fragen, Aussagen, Imperative, Modalverben, Dialogpartikel und -phrasen, phrasale Saetze, Uhrzeiten, Datum, Namen.

    Parserzeiten: durchschn: 1.8 cpu sec/Satzhypothese bei durschnittl. Laenge von ca. 5-10 Woertern/Aeusserung

19. *Entwicklungsumgebung zur Erstellung der Einträge:*

    Graphikoberflaeche zur Menue-gesteuerten Eingabe von Einträgen und Merkmalen. Die Eintragung erfolgt ueber kategorieabhaengige Auswahl der Merkmale in Untermenues. Des weiteren komfortable, graphikorientierte Umgebung fuer die Grammatikentwicklung und Test mit komfortablem Rule-tracing, online-Aenderungen der Grammatik und des Lexicons, batch-Eingaben, Mouse-sensitive Inspection der Baeume und Merkmale, etc.

20. *Anzahl und Art der Einträge:* Anzahl: ca: 1000- 1200 Lexikon Einträge je Projekt Vollformen mit allen syntaktischen und semantischen Merkmalen Grammatik: ca. 150 Regeln; die Regeln sind anwendungsunabhaengig.

Literaturangaben:

G.Th.Niedermair, M.Streit, H.Tropf: Linguistic Processing Related to Speech Understanding in SPICOS II. in: Speech Communication 9 (1990), North-Holland, pp. 565-585.

H.Tropf: Syntax in the Spoken Dialogue System SPICOS II. in: Proceeding of the EUROSPEECH Conference 1989, pp:30-33

4.12 Linguistischer Kernprozessor (LKP)

1. *Name der Komponente:* Linguistischer Kernprozessor (LKP)
2. *Zweck der Komponente:* 1) Analyse einer deutschen Eingabe 2) Generierung einer deutschen Ausgabe
3. *Entwickelt und implementiert durch:* Siemens AG, ZFE ST SN 74
4. *Anschrift, Kontaktperson, e-mail:* Hans Ulrich Block, Siemens AG, ZFE ST SN 74, Otto Hahn Ring 6, 8000 Muenchen 83, Tel. 089 / 636-44537, FAX: 089 / 636-42284, e-mail: block@ztivax.zfe.siemens.de
5. *Implementierungssprache:* PROLOG (IF-PROLOG, Sicstus, Quintus)
6. *Zugrundeliegende Hardware / Betriebssystem(e):* SUN 3/60, SUN SPARC, OS 4.1.1
7. *Beginn / Ende der Entwicklung:* 1990 / ...
8. *Projekt:* interne Projekte und ASL
9. *Input / Eingabeschnittstellen:* Deutsche Eingabe als ASCII- String oder als word lattice bzw. Quasi logical form (vgl. CLE vom SRI)
10. *Output / Ausgabeschnittstellen:* 1) Quasi logical form (vgl. CLE vom SRI) bzw. Deutsche Ausgabe als ASCII-String Anmerkung: QLF wird derzeit durch unterspezifizierte LF ersetzt.
11. *Aktueller Zustand:* Forschungsprototyp
12. *Abhängigkeiten / Anforderungen an andere Systeme:* keine
13. *Dokumentation:* teilweise
14. *Verfügbarkeit:* im Einzelfall zu klaeren
16. *Fremdinstallationen:* keine
17. *Benutzte Techniken und zugrundeliegende Theorien:* Tomita-Parser, Semantic-Head-Driven Generator, Constraints

    Unifikationsgrammatik mit Bewegungsregeln (Trace and Unification Grammar, TUG
18. *Allgemeine Beschreibung:*

    Mit dem LKP verfolgen wir die Entwicklung eines allgemeinen linguistischen Tools fuer unterschiedliche Anwendungen. Der LKP besteht zur Zeit aus:

    - Grammatikformalismus TUG
    - Parser-Generator (Grammatikcompiler)
    - Parser

- Generator-Generator (Grammatikcompiler)
- Generator
- Beschreibung eines Fragments des Deutschen in TUG mit Gleichungen zur Abbildung in eine semantische Form
- Lexikonerfassungstools fuer linguistisch ungeschulte Benutzer
- Beschreibung eines kleinen Fragments des Chinesischen in TUG
- Integrierte Sortenhierarchie fuer Selektionsrestriktionen nach Mellish 88.

Eine Komponente zur Diskursrepraesentation ist geplant.

Die Grammatik des LKP ist voellig reversibel und wird ohne Einschraenkung sowohl fuer die Analyse als auch fuer die Synthese benutzt. Die Analyse oder Generierung eines Satzes von ca. 10 Woertern dauert etwa 1-5 sec. auf einer SPARC 1.

Der LKP wird in verschiedenen Projekten unserer Fachgruppe zur Analyse und zur Generierung eingesetzt.

In Kooperation mit SRI Cambridge und Swedish Institute of Computer Science wurde der LKP zur reversiblen Uebersetzung Deutsch/Englisch und Deutsch/Chinesisch mit QLF als Transferebene eingesetzt.

Im Rahmen des ASL-Projekts wurde der LKP optimiert und zur Analyse von word-lattices nutzbar gemacht.

19. *Entwicklungsumgebung zur Erstellung der Einträge:*

    Morphologie-Tool fuer Adjektive, Verben und Substantive erzeugt halbautomatisch Vollformen. Syntax-Tool fuer Adjektive, Verben und Substantive erzeugt interaktiv vollstaendigen Lexikoneintrag.

    Beide Tools setzen keine linguistischen Kenntnisse voraus. Der Benutzer wird jeweils nur gefragt, ob ein bestimmter Ausdruck grammatisch wohlgeformt ist oder nicht, bzw. welcher von einer Menge von Ausdruecken wohlgeformt ist. Ein etwas geuebter Lexikonerfasser kann ca. 60-100 Woerter/Stunde erfassen.

20. *Anzahl und Art der Einträge:*

    ca. 1000 Einträge (abhaengig von der Anwendung)

Literaturangaben:

Block, H. U., H. Alshawi, e.a., 1991, Communications Multilingual par forme quasi logique. Proc. Avignon 91 Vol 8 (Spec. Conference Natural Language Processing and its Applications), pp. 245-252.

Block, H. U. und P. Peng, 1991: A Trace and Unification Grammar for Chinese. Proceedings of the R.O.C. Conference on Computational Linguistics. Taiwan.

Block, H. U., 1991, Compiling Trace and Unification Grammar for Parsing and Generation. Proc. Reversible Grammar Workshop at the ACL-Conference, Berkeley.

Block, H. U. und S. Schachtl, 1992, Trace and Unification Grammar. COLING 92.

Gehrke, M., 1991: Natural Interaction with Expert Systems: The GNEIS experience. Proc. Avignon 91 Vol 8 (Spec. Conference Natural Language Processing and its Applications), pp. 175-184.

Schachtl, S., 1991, Die Verbsyntax im LKP. Siemens Bericht INF2-NL-4-91.

Schachtl, S., 1991, A Disjunctive Treatment of the Distributional Properties of Attributive Relative Clauses in German. ASL-Bericht 7.

4.13 GuLP (General unification-based Linguistic Processor)

1. *Name der Komponente:* GuLP (General unification-based Linguistic Processor)
2. *Zweck der Komponente:* Parsing
3. *Entwickelt und implementiert durch:* G. Görz, C. Beckstein mit Anregungen von M. Kay
4. *Anschrift, Kontaktperson, e-mail:* Prof. Dr. G. Görz Universität Erlangen-Nürnberg IMMD 8 — KI Am Weichselgarten 9 D-8520 ERLANGEN

   Tel. (091310 699-126; fax: -198 goerz@informatik.uni-erlangen.de
5. *Implementierungssprache:* InterLISP
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Siemens Gro"rechner / BS2000 / InterLISP Portierung nach XeroxLISP (Sun SPARC) in Vorbereitung
7. *Beginn / Ende der Entwicklung:* 1980–1987
8. *Projekt:* GuLP in Zusammenarbeit mit EVAR (Prof. Niemann, Erlangen)
9. *Input / Eingabeschnittstellen:* Terminal, Files
10. *Output / Ausgabeschnittstellen:* Terminal, Files
11. *Aktueller Zustand:* abgeschlossen
12. *Abhängigkeiten / Anforderungen an andere Systeme:* keine außer Systemumgebung (s.o.)
13. *Dokumentation:* s. Literatur sowie Studienarbeit von C. Beckstein
14. *Verfügbarkeit:* nach Rücksprache
15. *Wartbarkeit:* ja
16. *Fremdinstallationen:* keine
17. *Benutzte Techniken und zugrundeliegende Theorien:* Chart- Parsing, LFG-ähnlicher Formalismus
18. *Allgemeine Beschreibung:* unifikationsbasierter Chart-Parser zur Strukturanalyse gesprochener und geschriebener Sprache

    Einzelheiten s. Literatur
19. *Entwicklungsumgebung zur Erstellung der Einträge:* InterLISP und Systemeditor, sonst keine
20. *Anzahl und Art der Einträge:* ca. 100 Einträge, nur morphosyntaktische Merkmale

Literaturangaben:

G. Görz, Strukturanalyse natürlicher Sprache. Bonn: Addison-Wesley, 1988 sowie mehrere Aufsätze und Tagungsbeiträge

4.14 ATNP

1. *Name der Komponente:* ATNP
2. *Zweck der Komponente:* Parsing
3. *Entwickelt und implementiert durch:* C. Pyka
4. *Anschrift, Kontaktperson, e-mail:* Prof. Dr. G. Görz Universität Erlangen-Nürnberg IMMD 8 — KI Am Weichselgarten 9 D-8520 ERLANGEN

   Tel. (091310 699-126; fax: -198 goerz@informatik.uni-erlangen.de
5. *Implementierungssprache:* UTLISP
6. *Zugrundeliegende Hardware / Betriebssystem(e):* CDC Cyber Großrechner / NOS / UTLISP
7. *Beginn / Ende der Entwicklung:* 1984–1985
8. *Projekt:* Diplomarbeit Pyka
9. *Input / Eingabeschnittstellen:* Terminal, Files
10. *Output / Ausgabeschnittstellen:* Terminal, Files
11. *Aktueller Zustand:* abgeschlossen
12. *Abhängigkeiten / Anforderungen an andere Systeme:* keine außer Systemumgebung (s.o.)
13. *Dokumentation:* Studienarbeit von C. Pyka
14. *Verfügbarkeit:* nach Rücksprache
15. *Wartbarkeit:* ja
16. *Fremdinstallationen:* in modifizierter Form bei Prof. Niemann, IMMD 5, Erlangen
17. *Benutzte Techniken und zugrundeliegende Theorien:* ATN- Parsing, inkl. Interpreter für kaskadierte ATNs
18. *Allgemeine Beschreibung:* ATN-Interpreter zur syntaktischen Analyse natürlicher Sprache
19. *Entwicklungsumgebung zur Erstellung der Einträge:* keine
20. *Anzahl und Art der Einträge:* ca. 100 Einträge, nur morphosyntaktische Merkmale

Literaturangaben:

RRZE IAB-220 (Regionales Rechenzentrum, Univ. Erlangen-Nürnberg) Pyka C ATNP - Ein portables Programmier- und Anwendungssystem für ATN- Grammatiken Diplomarbeit 1.1985

4.15 DCG-Workbench

1. *Name der Komponente:* DCG-Workbench
2. *Zweck der Komponente:* Entwicklung von DCG-basierten Grammatiken fuer verschiedene Sprachen
3. *Entwickelt und implementiert durch:* SNI
4. *Anschrift, Kontaktperson, e-mail:* Konrad Jablonski, SNI AG - AP 333, Anwendersoftware und Projekte, KI-Zentrum Paderborn, Riemekestr. 160, 4790 Paderborn jablonski.pad@sni.de
5. *Implementierungssprache:* C und PASCAL
6. *Zugrundeliegende Hardware / Betriebssystem(e):* UNIX, SNI-SINIX-Maschinen (TAI GON)
7. *Beginn / Ende der Entwicklung:* ab 1.4.90 bis 28.2.1991
8. *Projekt:* ESPRIT-Projekt 311/ ADKMS Gesamtprojektleiter: Konrad Jablonski (ab 1.1.1990) weitere Partner: TA Triumph Adler AG, Olivetti, TU Berlin, Datamont Uni Hildesheim
9. *Input / Eingabeschnittstellen:* High-Level-Grammatikformalismus mit Konzepten wie GPSG und LFG
10. *Output / Ausgabeschnittstellen:* DCG-Regeln direkt lauffaehig unter Prolog
11. *Aktueller Zustand:* Projekt-Prototyp lauffaehig
13. *Dokumentation:* Deliverables, EG-Bericht
14. *Verfügbarkeit:* Copyright SNI Einzellizenz auf Anfrage
15. *Wartbarkeit:* im Augenblick keine Wartung; nur im Rahmen eines Entwicklungsauftrags
17. *Benutzte Techniken und zugrundeliegende Theorien:* bewaehrte Compiler-Techniken
18. *Allgemeine Beschreibung:* Unterstuetzung von Multilingualitaet (Portierung von Grammatiken fuer andere Zielsprachen)

Literaturangaben: (gesamte NL-Publikationen SNI)

K.Jablonski, J.Samuel: Natural Language Information Access System; in: ESPRIT - Information Processing Systems; Brüssel 1990

4.16 TFS-System

1. *Name der Komponente:* TFS-System

2. *Zweck der Komponente:* Repräsentation und constraint-basierte Verarbeitung von einsprachigem und kontrastivem lexikalischem und grammatischem Wissen für sprachverarbeitende Systeme

3. *Entwickelt und implementiert durch:* Martin C. Emele und Rémi Zajac

4. *Anschrift, Kontaktperson, e-mail:* Martin C. Emele Projekt Polygloss IMS-CL Universität Stuttgart Azenbergstraße 12 D-W7000 Stuttgart 1, polygloss@adler.philosophie.uni-stuttgart.de

5. *Implementierungssprache:* Common Lisp (z.B. MCL, Allegro, Lucid, AKCL)

6. *Zugrundeliegende Hardware / Betriebssystem(e):* wo immer ein halbwegs vernünftiges Common Lisp läuft (existierende Implementierungen auf Macintosh, Sun4, Symbolics, TI Explorer; Portierungsaufwand i.d.R. minimal)

   Basisversion einer graphische Benutzerschnittstelle liegt fertig vor für Macintosh und Symbolics.

   Eine CLIM-basierte Version (Common Lisp/ X Window System / Interface Manager), die noch unabhängiger von der jeweiligen Hardware-Umgebung ist, wird derzeit entworfen und implementiert.

7. *Beginn / Ende der Entwicklung:* ab 1989 (wird weiterentwickelt)

8. *Projekt:* Polygloss (BMFT Projekt 08 B 3116 3 / 08 B 3120 6)

9. *Input / Eingabeschnittstellen:* interaktiv über Command-line interface (Format sind textuelle Darstellungen von Feature Strukturen sowohl als Repräsentations- als auch als Abfrage-Sprache)

10. *Output / Ausgabeschnittstellen:* Feature-Strukturen in textueller Form oder als LaTeX-Code (zur Weiterverarbeitung für graphische Darstellung);

11. *Aktueller Zustand:* Forschungsprototyp, aktuelle Version: TFS v5.6

12. *Abhängigkeiten / Anforderungen an andere Systeme:* stand-alone system

13. *Dokumentation:* Vollstaendige Dokumentation liegt nur für Version 4 vor.

    Version 5.6 wird gerade dokumentiert; erhebliche Unterschiede in der Spezifikationssyntax; neue Syntax durch extended BNF und anhand von Beispielspezifikationen dokumentiert

14. *Verfügbarkeit:* Copyright Institut für maschinelle Sprachverarbeitung, kostenlos im akademischen Bereich, binary distribution only

15. *Wartbarkeit:* keine externe Wartbarkeit, da keine source-code distribution; interne Wartbarkeit sehr gut durch inhärent modularen Aufbau

16. *Fremdinstallationen:* ca. 20 Installationen weltweit; ca. die Hälfte davon wird aktiv benutzt.

17. *Benutzte Techniken und zugrundeliegende Theorien:* constraint-logisches Programmieren mit typisierter Feature-Logik als Constraintsprache.

18. *Allgemeine Beschreibung:* siehe 17.

    TFS wurde als Spezifikationssprache für linguistische Faktenbeschreibungen entworfen. Die Besonderheit der Spezifikationssprache ist, daß sie gleichzeitig deklarativ und ausführbar ist. Deklarativ deshalb, weil sie es erlaubt, linguistische Fakten frei von Informationen über den Ablauf der jeweiligen Verarbeitung zu beschreiben. Ausführbar deshalb, weil die Objekte der Beschreibungssprache gleichzeitig die Datenobjekte charakterisieren, mit denen der Constraint-Solver arbeitet.

    Die zentralen Merkmale von TFS sind

    - die Unterstützung der Arbeit mit partiellen Beschreibungen, wie sie in Unifikationsgrammatiken üblich ist
    - die Möglichkeit des Aufbaus von Spezialisierungshierarchien von Typen (Klassen oder Mengen von Objekten) und deren Ausnutzung zur redundanzfreien Repräsentation linguistischen Wissens
    - der Einsatz von Constraints zur Beschreibung der Wohlgeformheitsbedingungen von linguistischen Objekten

    Es existieren Beispielmodellierungen, die auf einer Reihe aktueller Grammatiktheorien basieren. So sind Wohlgeformtheitsbedingungen aus Theorien wie LFG, HPSG und Systemic Functional Grammar modellierbar und als ausführbare Spezifikationen testbar.

19. *Entwicklungsumgebung zur Erstellung der Einträge:* momentan nur via textuelle Repräsentation von (partiellen) Merkmalsstrukturen. An einer komfortableren Benutzerumgebung wird momentan gearbeitet.

20. *Anzahl und Art der Einträge:* keine Angaben; es wurden jeweils beispielhafte Modellierungen vorgenommen, die die Machbarkeit der Kodierung und die Verwendbarkeit für jeweils Parsing und Generierung zeigen. Regelanzahl und Zahl der Lexikoneinträge waren für solche Modellierungen stets irrelevant.

Literatur

[Emele/Zajac 1990] Martin C. Emele and Rémi Zajac: "Typed Unification Grammars". In: Hans Karlgren (ed.): *Proceedings of the 13th International Conference on Computational Linguistics (CoLing90)*, Helsinki, August 1990.

- beschreibt die Beispielmodellierungen für DCG, PATR-II und HPSG-*Grammatiken; Referenzpapier für den Formalismus (nach der Syntax für Ver*sion 4)

[Emele/Zajac 90a] Emele, Martin C./Zajac, Rémi: A Fixed-Point Semanticü for Feature Type Systems. In: *Proceedings of The Second International Workshop on Conditional and Typed Rewriting Systems*, Concordia University, Montréal, Canada, June 11 - 14 1990; erschienen in den Springer Lecture üotes in Computer Science, Bd.

- Referenzpapier für die formale Semantik des TFS-Formalismus

[Zajac 1991f] Rémi Zajac: "Notes on the Typed Feature System". Project POLYGLOSS 1991.

- Referenzpapier im Sinne einer Benutzeranleitung; beschreibt Version-4 des Systems. Eine neuere Version, die die Veränderungen in der Syntax berücksichtigt, ist in Vorbereitung

[Bateman/Momma 1991] John A. Bateman and Stefan Momma: "The nondirectional representations of Systemic Functional Grammars and Semantics as Typed Feature Strutctures". Technical Report; Darmstadt/Stuttgart: GMD/Institut für Integrierte Publikations- und Informationssysteme; Institut für Maschinelle Sprachverarbeitung, Universität Stuttgart, January 1991.

- beschreibt die Modellierung von systemischen Grammatiken in TFS anhand der NIGEL-Grammatik

[Emele 1988] Martin C. Emele. "A typed feature structure unification-based approach to generation". In: *Proceedings of the WGNLC of the IECE*, Oita, Japan, 1988. IECE, Oita University.

- erste Beschreibung einer tatsächlich implementierten HPSG-Grammatik, die für Analyse und Generierung funktioniert

[Emele 1991] Martin C. Emele: "Unification with Lazy Non-Redundant Copying". In: *Proceedings of the 29th annual meeting of the ACL*, Berkeley, CA, June 1991. Association for Computational Linguistics.

- beschreibt einen wesentlichen Teil der internen Implementierung, der entscheidend zur Effizienz des implementierten Prototypen beiträgt.

[Emele et al. 1990a] Martin Emele, Ulrich Heid, Stefan Momma and Rémi Zajac: "Organizing linguistic knowledge for multilingual generation". In: Hans Karlgren (ed.), *Proceedings of the 13th International Conference on Computational Linguistics (CoLing90)*, Helsinki, August 1990.

- beschreibt mögliche Organisationsprinzipien für linguistische Wissensquellen in einer constraint-basierten Verarbeitungsumgebung

[Emele et al. 1990] Martin Emele, Ulrich Heid, Stefan Momma and Rémi Zajac: "Interactions between Linguistic Constraints: Procedural vs. Declarative Approaches". *Machine Translation - Special issue on natural language generation (ed.: Richard Kittredge)*, 1991.

- wesentlich ausführlichere Version des obigen CoLing-Papiers

[Zajac 1989] Rémi Zajac. "A transfer model using a Typed Feature Structure rewriting system with inheritance". In: *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics*, Vancouver, 1989.

- zur Anwendung des TFS-Formalismus im Rahmen von MÜ-Systemen

[Heid 1990] Ulrich Heid. "Syntactic Information in (Machine) Translation Dictionaries - towars a modular architecture for bilingual dictionaries". To appear in: Karl Hyldgaard-Jensen and Arne Zettersten (Eds.): *Symposium on Lexicography, 5, Proceedings of the 5th International Symposium on Lexicography, Kobenhavn, May 1990*, Tübingen, 1992.

[Heid 1991a] Ulrich Heid. "Towards reusable lexical resources for natural language processing. Some proposals for linguistic knowledge representation". In: *Proceedings of the 11th International Conference on Expert Systems and their Applications, Natural Language Section*, Avignon, 1991.

[Heid 1991b] Ulrich Heid. "Zur Lexikonarchitektur für ein constraint-basiertes maschinelles Übersetzungssystem". In: Jürgen Rolshoven and Dieter Seelbach (Eds.): *Romanistische Computerlinguistik. Theorien und Implementationen*, Tübingen, 1991.

[Zajac 1991a] Rémi Zajac. "Computing partial information using approximations - semantics of typed feature structures". To appear in: *Proceedings of the International Workshop on Constraint Based Formalisms for Natural Language Generation*, IMS, University of Stuttgart, 1991.

[Zajac 1991b] Rémi Zajac: "Inheritance and Constraint-based Grammar Formalisms". Project POLYGLOSS; internal paper.

[Zajac 1991c] Rémi Zajac. "Inheritance networks of typed feature structures". To appear in: *Computational Linguistics - special issue on Inheritance in natural language processing*, (Walter Daelemans, ed.) 1991.

[Zajac 1991d] Rémi Zajac. "Issues in the design of a language for representing linguistic information based on inheritance and feature structures". In: *Proceedings of the ACQUILEX Workshop on Default Inheritance*, Cambridge, April 1991.

[Zajac 1991e] Rémi Zajac: "Modularity and Stratification in Unification Grammars". Technical report, IMS-CL, University of Stuttgart, January 1991, internal paper.

[Zajac 1991g] Rémi Zajac: "Towards Computer-Aided Linguistic Engineering". Project POLYGLOSS 1991, submitted for CoLing '92.

[Zajac 1991h] Rémi Zajac. "Towards object-oriented constraint logic programming". In: *Proceedings of the ICLP'91 Workshop on Advanced Logic Programming Tools and Formalisms for Natural Language Processing*, Paris, June 1991.

[Zajac 1991i] Rémi Zajac. "A uniform architecture for parsing, generation and transfer". In: *Proceedings of the ACL'91 workshop on reversible grammars in natural language processing*, Berkeley, CA, June 1991. Umfrage zu existierenden Software-Komponenten im Bereich Verarbeitung natuerlicher Sprache

4.17 Context Feature Structure System (CFS-System)

1. *Name der Komponente:* Context Feature Structure System (CFS-System)

2. *Zweck der Komponente:* Graphunifikationsformalismus

   Anwendungsbereich - Unifikationsgrammatiken und Wissensrepraesentation - Entwicklung von Grammatiken und Lexika - integrierte syntaktische und semantische Textanalyse - Generierung

   zur Zeit benutzt fuer Textanalyse (integriertes Parsing ueber 5 Schichten (Syntax, thematische Struktur, Referenzstruktur, semantische Repraesentation des Textes, Hintergrund-Wissensrepraesentation)

3. *Entwickelt und implementiert durch:*

   Context Feature Structure System: Michael Koenyves-Toth, Martin Boettcher Parser fuer Dependenzgrammatiken: Roland Stuckardt

4. *Anschrift, Kontaktperson, e-mail:* Dr. Karin Haenelt, Institut fuer Integrierte Publikations- und Informationssysteme (IPSI), der GMD, Dolivostrasse 15, 6100 Darmstadt, Tel. 06151/869-811, Fax 06151/869-818, email haenelt@darmstadt.gmd.de

5. *Implementierungssprache:* LISP

6. *Zugrundeliegende Hardware / Betriebssystem(e):* Symbolics / Sun /

7. *Beginn / Ende der Entwicklung:* Beginn: 1989 Ende: Mai 1992 (Basissystem) das System wird danach weiter ausgebaut werden (z.B. Inferenzkomponenten)

8. *Projekt:* KONTEXT Knowledge Oriented Processing of Natural Language Texts (Textanalyse)

9. *Input / Eingabeschnittstellen:* Natuerlichsprachiger Text Feature Structures (Grammatik, Lexikon, Wissensbasis)

10. *Output / Ausgabeschnittstellen:* Feature Structures

11. *Aktueller Zustand:* Basissystem in Benutzung das System wird weiter ausgebaut (z.B. Inferenzkomponenten)

13. *Dokumentation:* Michael Koenyves-Toth: Das Context Feature Structure System. Dissertation. TH Darmstadt 1992 (erscheint) Martin Boettcher: The CFS System User Manual. IPSI-report Juni 1991

14. *Verfügbarkeit:* mit IPSI / GMD zu verhandeln

16. *Fremdinstallationen:*

    Universitaet Heidelberg, Lehrstuhl fuer Computerlinguistik, Prof. Dr. P. Hellwig

    Forschungsgruppe Computerlinguistik (ehem. ZISW, demnaechst Humboldt Universitaet) Berlin Prof. Dr. Juergen Kunze

18. *Allgemeine Beschreibung:*

    inkrementeller Graphunifikationsformalismus fuer rekursive, disjunktive und negierte Attributgraphen

    besondere Effizienz durch - Vermeidung aufwendigen Kopierens mehrfach verwendeter Teilgraphen - unabhaengige Behandlung der logischen und der topologischen Struktur bei Disjunktionen - Vermeidung des aufwendigen kompletten Aufbaus des Verbandes der Klassen- (Subsumptions-)Hierarchie

    Eigenschaften - Verarbeitung zyklischer Graphen - Verarbeitung rekursiver Typen - dynamisches Definieren von Konzepten aus Instanzen - verteilte Disjunktionen - sequentielle Schnittstelle

19. *Entwicklungsumgebung zur Erstellung der Einträge:*

    Hybrider Feature Editor mit integriertem Evaluator

20. *Anzahl und Art der Einträge:*

    Anzahl: wachsend Art: lexikalisierte Textgrammatik: Feature Structures, in denen der Beitrag sprachlicher Ausdruecke zum Aufbau von Satzstruktur, Referenzstruktur, thematischer Struktur und konzeptueller Struktur beschrieben ist. Theoretische Basis: Satzstruktur: PLAIN-Grammatik (P.Hellwig) Thematische Struktur: Prager Textlinguistik, Emphase-Modell (J.Kunze) Referenzstruktur: (Ch.Habel), Emphase-Modell (J.Kunze), Situation-Semantics Konzeptstrukturen: Emphase-Modell (J.Kunze), Negation und Praesupposition (U.Jung/H.Kuestner) Situation Semantics

Literaturangaben:

***** KONTEXT-Modell *****

Haenelt, Karin / Koenyves-Toth, Michael: The Textual Development of Non-Stereotypic Concepts. In: Proceedings of the 5th Conference of the European Chapter of the Association for Computational Linguistics, April 9-11, 1991, Berlin 1991.

***** CFS-System *****

Koenyves-Toth, Michael: Das Context Feature Structure System. Dissertation. TH Darmstadt 1992 (erscheint)

Boettcher Martin: The CFS System User Manual. IPSI-report Juni 1991

Koenyves-Toth, Michael: Incremental Evaluation of Disjunctive Feature Terms. Arbeitsberichte der GMD. Sankt Augustin, November 1991

***** Lexikon und Grammatik *****

Firzlaff, Beate / Haenelt, Karin: Applying Text Linguistic Principles to Modelling Meaning Paraphrases. In: Proceedings of Fifth EURALEX International Congress, University of Tampere, Finland, August 4-9, 1992

Firzlaff, Beate / Haenelt, Karin: Textual Modelling of Meaning Paraphrases for the Acquisition of Conceptual Definitions. In: Proceedings of the 14th International Conference on Computational Linguistics, July 1992, Nantes, France 1992

Haenelt, Karin: Towards a Quality Improvement in Machine Translation: Modelling Discourse Structure and Including Discourse Development in the Determination of Translation Equivalents. In: Proceedings of the 4th International Conference on Theoretical and Methodological Issues in Machine Translation, June 25-27, 1992, Montreal, Cannada

a detailed description (book) of the dictionary is in preparation: Boettcher, Martin / Firzlaff, Beate / Haenelt, Karin / Kunze, Juergen / Kuestner, Herbert: Sememe Structures and Field Structures. Components of Meaning Descriptions in an Electronic Dictionary

4.18 T D L ExtraLight

1. *Name der Komponente:* T D L ExtraLight

2. *Zweck der Komponente:* Erstellung und Test von unifikationsbasierten Lexika und Grammatiken (d.h. von Merkmalstypen a la HPSG) für NL Systeme

3. *Entwickelt und implementiert durch:* Hans-Ulrich Krieger, Ulrich Schäfer

4. *Anschrift, Kontaktperson, e-mail:* DFKI/Saarbrücken, Hans-Ulrich Krieger, krieger @dfki.uni- sb.de

5. *Implementierungssprache:* CommonLISP (Franz und Macintosh Allegro CL/sowohl CLtL1 als auch CLtL2)

6. *Zugrundeliegende Hardware / Betriebssystem(e):* Sun/Solbourne SPARC-Architektur/UNIX (Sun OS) und Apple Macintosh / System 6.0.5 und 6.0.7

7. *Beginn / Ende der Entwicklung:* Beginn im Winter '90, erste lauffähige, stabile und auch eingesetzte Version im Herbst '91 (WBR-Sitzung des DFKI); seit dem weitere Verbesserungen (kleine Erweiterungen und Änderungen, um das Gesamtsystem schneller zu machen); TDL ExtraLight wird abgelöst, bis ein erster Prototyp vom Nachfolgersys tem TDL vorliegt (siehe gesonderten Bogen zu TDL)

8. *Projekt:* DISCO (DFKI)

9. *Input / Eingabeschnittstellen:* Sowohl ASCII-Terminal als auch window-basierte Oberfläche

10. *Output / Ausgabeschnittstellen:* Sowohl ASCII-Terminal als auch window-basierte Oberfläche

11. *Aktueller Zustand:* laufendes System, mit dem schon eine relativ grosse(s) Grammatik/Lexikon (6 Regeln/Regelschemata, 80 Stämme, 330 Typdefinitionen) im HPSG-Stil für das DISCO-Projekt im DFKI entwickelt wurde (siehe Fragebogen zur "Grammatik/Lexikon für DISCO") Zu TDL ExtraLight existiert eine kleine Grammatikentwicklungsumgebung (Feature- Editor, Grapher, Übersetzer nach LaTeX hin), wie auch einen Parser mit Chart- Display als auch einen Generator (s. dazu die gesonderten Bögen !!!)

12. *Abhängigkeiten / Anforderungen an andere Systeme:* + CommonLISP und CommonLISP Object System (CLOS) als Wirtssprache + Für die Oberflächensyntax wird der Public-Domain Parser-Generator/ Compiler-Compiler ZEBU (ähnlich YACC) verwendet; + Ungetypter Unifikator, der es aber erlaubt, den Typ einer Merkmalsstruktur in der Struktur zu vermerken (die Kommunikation zwischen Typsystem und Unifikator findet über eine wohldefinierte Schnittstelle statt)—im Augenblick wird der im DISCO-Projekt entwickelte, mächtige Unifikator Udine eingesetzt (siehe gesonderten Bogen)

13. *Dokumentation:* Es gibt eine Kurzdoku von etwa 10 Seiten, eine grosse Grammatik mit vielen Beispielen (s.o. bzw. Bogen zur Grammatik), eine BNF zur Oberflächensprache in der die Typen eingegeben werden und mehrere Ansprechpartner im DISCO- Projekt

14. *Verfügbarkeit:* Ja (Copyright/Preis möglicherweise nach Absprache)

15. *Wartbarkeit:* Modular aufgebaut; Mehrere Ansprechpartner im DISCO-Projekt (mind. bis Ende '93);

16. *Fremdinstallationen:* Anfragen von ASL (Berlin und Erlangen), Uni Bielefeld (Prof. D. Gibbon), GuK-Projekt (Leiter: Prof. M. Pinkal), BiLD-Projekt (Leiter: Prof. H. Uszkoreit)

17. *Benutzte Techniken und zugrundeliegende Theorien:* + Vererbung im Stil von HPSG + Ungetypte Feature-Logik in Form eines Unifikators (s. Bogen zu 'Udine') + schwaches Domänenkonzept + Anfragen an das Typsystem werden zum Teil effizient über CLOS gelöst + z.Zt. wird mit voll expandierten Merkmalsstrukturen gearbeitet + Möglichkeit Pfade auszublenden, d.h. vor der Unifikation zu schützen + verschiedenste Informationen über Autor, Evaluationsdatum, Dokumentation etc. werden in den Typen verzeichnet + Functional Constraints (Funktionen), die man im Unifikator formuliert, werden auch vererbt

18. *Allgemeine Beschreibung:* siehe dazu Punkt 17

4.19 T D L

1. *Name der Komponente:* T D L
2. *Zweck der Komponente:* Erstellung und Test von unifikationsbasierten Lexika und Grammatiken (d.h. von Merkmalstypen a la HPSG) für NL Systeme
3. *Entwickelt und implementiert durch:* Hans-Ulrich Krieger, Stephan Diehl, Karsten Konrad, Ulrich Schäfer
4. *Anschrift, Kontaktperson, e-mail:* DFKI/Saarbrücken, Hans-Ulrich Krieger, krieger@dfki.uni- sb.de
5. *Implementierungssprache:* CommonLISP (Franz und Macintosh Allegro CL/CLtL2)
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Sun/Solbourne SPARC-Architektur/ UNIX (Sun OS) und Apple Macintosh / System 6.0.5 und 6.0.7
7. *Beginn / Ende der Entwicklung:* Beginn im Dezember '91, Ende ca. Anfang '93, erster Prototyp Mitte '92; Reimplementierung und Erweiterung von TDL ExtraLight (s. gesonderten Bogen)
8. *Projekt:* DISCO (DFKI)
9. *Input / Eingabeschnittstellen:* Sowohl ASCII-Terminal als auch window-basierte Oberfläche
10. *Output / Ausgabeschnittstellen:* Sowohl ASCII-Terminal als auch window-basierte Oberfläche
11. *Aktueller Zustand:* Noch in der Entwicklung, s. dazu Punkt 7
12. *Abhängigkeiten / Anforderungen an andere Systeme:* + CommonLISP als Wirtssprache + Für die Oberflächensyntax wird der Public-Domain Parser-Generator/ Compiler-Compiler ZEBU (ähnlich YACC) verwendet; + Ungetypter Unifikator, der es aber erlaubt, den Typ einer Merkmals- struktur in der Struktur zu vermerken (die Kommunikation zwischen Typsystem und Unifikator findet über eine wohldefinierte Schnittstelle statt)—im Augenblick wird der im DISCO-Projekt entwickelte, mächtige Unifikator Udine eingesetzt (siehe gesonderten Bogen)
13. *Dokumentation:* Zum Vorgänger TDL ExtraLight gibt es Dokumentation; Zu TDL nur interne Doku (voraussichtlich in einigen Wochen aber auch zwei DFKI Reports und ein DFKI Technical Document)
14. *Verfügbarkeit:* Ja (Copyright/Preis möglicherweise nach Absprache)
15. *Wartbarkeit:* Extrem modular aufgebaut; Mehrere Ansprechpartner im DISCO-Projekt (mind. bis Ende '93);
16. *Fremdinstallationen:* Fremdinstallationen bisher nur bei TDL ExtraLight, da TDL sich noch in der Entwicklung befindet

17. *Benutzte Techniken und zugrundeliegende Theorien:* + Ungetypte Feature-Logik in Form eines Unifikators (s. Bogen zu 'Udine') + 'low-level' vererbungsbasiertes Schliessen a la Ait-Kaci (verschiedenste Algorithmen und Datenstrukturen zur Kodierung der Typhierarchie) + 'high-level' Schliessen mittels symbolischer Typsimplifikation + Integration von Appropriateness-Bedingungen, Type-Checking und Typinferenz + neue, von DISCO-Mitarbeitern entwickelte Verfahren * eine sowohl theoretisch interessante als auch praktisch effizient zu implementierende Form von nicht-monotoner Vererbung * dynamisches Definieren, als auch Redefinieren von konjunktiven und disjunktiven Typbeschreibungen * Generierung von neuen Typen 'on the fly' zur Laufzeit—verschie- denste Expansionsstrategien * Einsatz von Memo-Techniken * Domänenkonzept * rekursive Typen mit 'partial evaluation' * Ersetzung der ungetypten Unifikation durch extensiven Einsatz des Typsystems—Unterscheidung zwischen nicht-expandierten, teilweise expandierten und vollständig expandierten Merkmalsstrukturen * Functional Constraints, die man im Unifikator definiert, werden vererbt

18. *Allgemeine Beschreibung:* (s. Punkt 2 und besonders Punkt 17) Es existiert eine kleine, schon in TDL ExtraLight eingestzte Grammatikent- wicklungsumgebung, bestehend aus Merkmalseditor und Grapher. Zusätzlich werden weitere Informationen verzeichnet, wie etwa Autorenname, Datum der Evaluation, Doku-String etc. Das Verhalten des Systems kann über Schalter gesteuert werden, etwa Warnungen beim Redefinieren, An- und Ausschalten des statischen Typcheckings, etc.

4.20 Playmobild

1. *Name der Komponente:* Playmobild
2. *Zweck der Komponente:* Testumgebung zur Entwicklung reversibler Grammatiken und Verfahren; umfasst Parser, Generator, Patr aehlichen Merkmalsformalismus
3. *Entwickelt und implementiert durch:* Gertjan van Noord, Guenter Neumann
4. *Anschrift, Kontaktperson, e-mail:* Guenter Neumann, neumann@dfki.uni-sb.de
5. *Implementierungssprache:* Prolog
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Macintosh, Unix
7. *Beginn / Ende der Entwicklung:* 1.1991
8. *Projekt:* SFB 314, Projekt N3, BiLD
9. *Input / Eingabeschnittstellen:* Parsing: string, Generierung: pred/arg-liste
10. *Output / Ausgabeschnittstellen:* Parsing: pred/arg Liste, Generierung: string wahlweise in beiden Faellen auch detaillierte Merkmalsstrukturen
11. *Aktueller Zustand:* noch in Entwicklung
12. *Abhängigkeiten / Anforderungen an andere Systeme:* Prolog, Modularitaet, Reversibilitaet
13. *Dokumentation:* kaum
14. *Verfügbarkeit:* Copyright
15. *Wartbarkeit:* wird nicht uebernommen
17. *Benutzte Techniken und zugrundeliegende Theorien:* kopfgesteuerte Parser, Generator, lexikalisch-basierte Grammatiken, Merkmalslogik, strikte Integration von Parsing und Generierung
18. *Allgemeine Beschreibung:* Neben den kopfgesteuerten Verfahren ist eine Monitoringkomponente entwickelt worden. Diese erlaubt die Generierung von nicht-ambigen Saetzen und Generierung von Paraphrasen. Diese Methoden basieren ebenfalls auf reversiblen Grammatiken. Zur Zeit operieren sie auf Derivationsbaeumen von Saetzen. Es wurde damit begonnen (Maerz 92), die Generierung und Parsing inkrementell zu integrieren. Damit wird es moeglich, waehrend der Generierung erzeugte Teilstrings auf ihren Ambiguitaetsgrad hin zu ueberpruefen und gegebenenfalls zu revidieren.
19. *Entwicklungsumgebung zur Erstellung der Einträge:* Patr aehnlicher Formalismus, einfaches Typsystem

20. *Anzahl und Art der Einträge:* gering

Literaturangaben:

1. Neumann, Guenter: A Bidirectional Model for Natural Language Processing, in Proc. of 5th EACL, Berlin, 1991. 2 Neumann, Guenter: Reversibility and Modularity in Natural Language Generation, in: workshop "Reversible Grammar in Natural Language Processingöf 29th ACL, Berkeley, CA, 1991.

3. Shieber, van Noord, Moore and Pereira: Semantic-head driven Generation, in Computational Linguistics 16(1), 1990.

4. van Noord, Gertjan: Head corner parsing for discontinuous constituency, in Proc. 29th ACL, Berkeley, CA.

5. Neumann and van Noord: Self-Monitoring with Reversible Grammars, submitted to Coling92, Nantes, France.

4.21 TAGDevEnv - Tree Adjoining Grammar Development Environment

1. *Name der Komponente:* TAGDevEnv - Tree Adjoining Grammar Development Environment

2. *Zweck der Komponente:*

   TAGDevEnv ist eine Werkbank zur Unterstuetzung der Arbeit eines Grammatikentwicklers fuer den Grammatikformalismus der Tree Adjoining Grammars. Es stehen unter einer benutzerfreundlichen Menueoberflaeche verschiedene Dienste zur Verfuegung:

   1. Graphischer Modus bei der Grammatikeingabe, d.h. der Grammatikschreiber muss mit keiner Hilfsdarstellung fuer die Eingabe von Baeumen vertraut sein.Die Komponente erlaubt auch innerhalb einer Sitzung mit mehrere Grammatiken zu arbeiten, zwischen Ihnen zu kopieren usw. Desweiteren erfordert eine Sitzung keine komplette Grammatikspezifikation, d.h. man kann auch inkorrekte Grammatiken abspeichern und gegebenenfalls in einer spaeteren Sitzung korrigieren.

   Es stehen verschiedene Tools zur Grammatikueberpruefung zur Verfuegung: a. Hilfreich - besonders bei der Entwicklung einer grossen Grammatik - ist ein Checker, der die syntaktische Korrektheit der Grammatik ueberprueft. Dieser wird aber nur bei Bedarf explizit aufgerufen, so dass wie oben angesprochen zeitweise falsche Grammatiken bearbeitbar bleiben.

   b. Ein sogenannter "Stepperërlaubt die Durchfuehrung der Rekursions- operation Adjunktion mit konkreten Baeumen der Grammatik. Man kann so also intendierte Ableitungen von Hand testen. Das kann gerade bei unuebersichtlichen Situtation, wie sie bei mehreren geschachtelten Adjunktionen entstehen, leichter zum Erkennen von falschen oder unvollstaendigen Regeln fuehren.

   c. Eine einfachere und globalere Grammatikueberpruefung ist mithilfe des Parsers fuer TAGs moeglich. Dazu wird eine Grammatik und ein Lexikon aus der gegebenenfalls vorhandenen Menge an bearbeiteten Wissensbasen, sowie in einem eigenen Fenster abgelegte zu parsende Saetze ausgewaehlt. Als Ausgabe berechnet der Parser alle Ab- leitungsbaeume. Als zusaetzliche Information kann mittels Anklicken eines Knotens die komplette Regel aufblinken. Das ist eine wichtige Hilfe, da bei Adjunktionen die Regeln sehr zer- stueckelt werden und man so bei der Bewertung und dem Vergleich der einzelnen Ableitungen unterstuetzt wird.

   2. Fuer die Lexikoneingabe und Wartung steht ebenfalls eine menuebasierte Komponente zur Verfuegung, die eine effiziente und konsistente Haltung von verschiedenen Lexika erlaubt. In TAGDevEnv ist derzeit keine Morphologiekomponente angeschlossen, so dass man nur mit Vollformenlexika arbeiten kann. Die Schnittstelle fuer eine solche Anbindung, z.B. von MORPHIX, das auf der Maschine in der gleichen Programmiersprache vorliegt, ist aber bereits vorgesehen.

   3. Um den sicheren Umgang mit grossen Datenmengen zu gewaehrleisten, sind verschiedene Recovery Routinen realisiert worden. Diese erlauben auch einem ungeu-

ebten und mit der Technik der Maschine wenig vertrauten Benutzer immer in einen definierten Zustand zurueckzukehren, so dass die Daten, die bis dahin vielleicht noch nicht gesichert worden sind, nicht verloren gehen.

3. *Entwickelt und implementiert durch:* Klaus Schifferer als Diplomarbeit am Lehrstuhl von Prof. Dr. W. Wahlster

4. *Anschrift, Kontaktperson, e-mail:* Prof. Dr. W. Wahlster Universitaet des Saarlandes Informatik IV Im Stadtwald 15 W - 6600 Saarbruecken 11 wahlster@dfki.uni-sb.de

5. *Implementierungssprache:* INTERLISP-D unter Release „Koto"

6. *Zugrundeliegende Hardware / Betriebssystem(e):* XEROX der 1100 Reihe bzw. SIEMENS EMS

7. *Beginn / Ende der Entwicklung:* ca. 1987 - 1989

8. *Projekt:* keine direkte Projekteinbindung

9. *Input / Eingabeschnittstellen:* Interaktives System zur Unterstuetzung der Entwicklung von konkreten Tree Adjoining Grammars und deren Austesten mithilfe von eingegebenen Test- saetzen.

10. *Output / Ausgabeschnittstellen:* Grammatiken, Lexika, Ableitungsbaeume zu konkreten Eingabesaetzen.

11. *Aktueller Zustand:* fertig

12. *Abhängigkeiten / Anforderungen an andere Systeme:* keine

13. *Dokumentation:* Ausfuehrliche Dokumentation der theoretischen Ueberlegungen, sowie des Programmkodes in der Diplomarbeit von Klaus Schifferer: "TAGDevEnv - Eine Werkbank fuer TAGsäm Lehrstuhl fuer Kuenstliche Intelligenz und Datenbanken, an der Universitaet des Saarlandes, Saarbruecken, 1989.

14. *Verfügbarkeit:* fuer Forschungszwecke freie Kopien erhaeltlich

15. *Wartbarkeit:* modular aufgebaut, z.B. sind einige Erweiterung wie die Anbindung einer Morphologiekomponenten schon vorgesehen.

16. *Fremdinstallationen:* Kopien an CSLI, UPenn

17. *Benutzte Techniken und zugrundeliegende Theorien:*

    TAGDevEnv unterstuetzt die Grammatikentwicklung fuer Tree Adjoining Grammars, ein Formalismus der 1975 von Joshi, Levi und Takahashi [Joshi et al. 75] fuer die Beschreibung natuerlicher Sprache vorgestellt wurde. Dafuer ist besonders die Eigenschaft der groesseren Lokalitaets- domaene als bei kontextfreien Grammatiken verantwortlich (z.B. koennen long-distancy Problem in einer Regel charakterisiert werden).

TAGDevEnv nutzt eine weitere Eigenschaft von TAGs, naemlich, dass sie in polynomieller Zeit parsbar sind. Auf dieser Erkenntnis baut der integrierte Parser auf und kann so in fuer den Benutzer akzeptabler Zeit die Ableitungsbaeume auch fuer komplexe Eingabesaetze erzeugen.

18. *Allgemeine Beschreibung:*

    siehe z.B. die die 25 seitige Beschreibung in [Schifferer 88]

19. *Entwicklungsumgebung zur Erstellung der Einträge:* TAGDevEnv ist gerade eine solche Entwicklungsumgebung fuer den Formalismus der Tree Adjoining Grammars.

20. *Anzahl und Art der Einträge:* Da die konkreten Grammatiken und Lexika uns selbst nur zu Testzwecken dienen, existieren nur eine Reihe von Spielzeuggrammatiken.

Literaturangaben:

[Joshi et al 75] A. K. Joshi, S. Levy, M. Takahashi Tree Adjoining Grammars, Journal of Computer and Systems Science 10:1, Seite 136-163, 1975.

[Schifferer 88] K. Schifferer TAGDevEnv - Eine Werkbank fuer TAGs, im Tagungsband des "Symposium - Computerlinguistik und ihre theoretischen Grundlagen", Saarbruecken, 1988 bzw. Bericht 39 des SFB 314 und des KI-Labors am Lehrstuhl fuer Informatik IV, Saarbruecken, 1988.

Eine gute Einleitung in den Formalismus der Tree Adjoining Grammars: [Joshi 85] A. K. Joshi An Introduction to Tree Adjoining Grammars, Technical Report MS-CIS-86-64, Dept. of Computer and Information Science, Moore School, Univ. of Pennsylvania, Philadelphia, Pennsylvania, 1985

Fuer ein grundlegendes Verstaendnis der Idee, dass TAGs fuer die Beschreibung natuerlicher Sprache besonders geeignet sind:: [Kroch, Joshi 85] A. Kroch, A. K. Joshi *Linguistic Relevance of Tree Adjoining Grammars, Technical Report MS-CIS-85-16, Dept. of* Computer and Information Science, Moore School, Univ. of Pennsylvania, Philadelphia, Pennsylvania, 1985

4.22 CUF-System-Kern

1. *Name der Komponente:* CUF-System-Kern
2. *Zweck der Komponente:* Entwicklung und Verarbeitung von CUF-Programmen (z.B. HPSG-artigen Grammatiken, die je nach Form des Goals Sätze parsen bzw. generieren)
3. *Entwickelt und implementiert durch:* Andreas Eisele und Jochen Dörre
4. *Anschrift, Kontaktperson, e-mail:* jochen@adler.philosophie.uni- stuttgart.de
5. *Implementierungssprache:* Prolog (Sicstus oder Quintus)
6. *Zugrundeliegende Hardware / Betriebssystem(e):* woimmer man Sicstus zum Laufen bringt (Sun, IBM RS6000, Next ...)
7. *Beginn / Ende der Entwicklung:* 1990-1991
8. *Projekt:* DYANA (ESPRIT Basic Research Action 3175)
9. *Input / Eingabeschnittstellen:* je nach Programm
10. *Output / Ausgabeschnittstellen:* Feature-Strukturen als Prolog-Terme oder als Postscript-Code (graph. Darstellung)
11. *Aktueller Zustand:* Forschungsprototyp, wird z.Zt. um ein Typsystem erweitert
12. *Abhängigkeiten / Anforderungen an andere Systeme:* -
13. *Dokumentation:* ist geplant
14. *Verfügbarkeit:* Copyright Inst. f. masch. Sprachverarbeitung, kostenlos im akademischen Bereich
15. *Wartbarkeit:* gut
16. *Fremdinstallationen:* ca. 5 Kopien wurden weitergegeben
17. *Benutzte Techniken und zugrundeliegende Theorien:* constraint-logisches Programmieren mit Feature-Logik als Constraintsprache
18. *Allgemeine Beschreibung:* siehe 17. Ein Programm besteht aus einem Logikteil, etwa als (pure) Feature-Prolog charakterisierbar, und einem Kontrollinformationsteil, in dem z.B. Information bzgl. der Indizierung von Klauseln und Information bzgl. Verzögerung von bestimmten Goals angegeben werden kann. Die Kontrollinformation hat dabei keinen Einfluss auf die Logik des Programms, sondern dient nur dazu, die Suche effizient zu gestalten. Beweisstrategie: Constraint-SLD-Resolution mit Selektionsstrategie "Deterministische Goals zuerst"(wobei versucht wird, Determinismus dynamisch zu bestimmen).

Fuer das System existiert eine HPSG-artige Demo-Grammatik, die einem Satz (als Prolog-Liste von Worten) eine Praedikat-Argument-Struktur zuordnet. Die Grammatik ist möglichst originalgetreu den Lösungen von Pollard/Sag (1987) Information-Based Syntax and Semantics"(CSLI Lect. Notes 13, Stanford Univ.) nachempfunden (excl. Adjunktion).

Literaturangaben:

[Dörre/Eisele91] Jochen Dörre and Andreas Eisele. A Comprehensive Unification-Based Grammar Formalism. DYANA Deliverable R3.1.B, ESPRIT Basic Research Action BR3175, Jan. 1991.

- diskutiert u.a. die Theorie der Kombination CLP+Featurelogik sowie ihre Anwendung auf Grammatikformalismen

4.23 TACTILUS

1. *Name der Komponente:* TACTILUS
2. *Zweck der Komponente:* Zeigegestenanalyse fuer NL- Dialogsysteme
3. *Entwickelt und implementiert durch:* Juergen Allgayer
4. *Anschrift, Kontaktperson, e-mail:* ali@helen.uni- sb.de
5. *Implementierungssprache:* LISP, FLAVORS
6. *Zugrundeliegende Hardware / Betriebssystem(e):* LISP- Machine, Genera 6/7
7. *Beginn / Ende der Entwicklung:* 87-88
8. *Projekt:* SFB314:N1 XTRA
9. *Input / Eingabeschnittstellen:* In: Mouse Eingabe, (semantische) Constraints aus Parser
10. *Output / Ausgabeschnittstellen:* Out: Nach Prioritaeten geordnete Liste von Kandidaten fuer die Referenten der Zeigehandlung
11. *Aktueller Zustand:* abgeschlossene Diplomarbeit; Fortsetzungsarbeit von M. Wille (TACTILUS II) mit aufwendiger Fokussierungsmoeglichkeit
12. *Abhängigkeiten / Anforderungen an andere Systeme:* Betreitstellen semantischer Information vom Parser
13. *Dokumentation:* Juergen Allgayer: TACTILUS Univ. Kaiserslautern, 1988 Michael Wille: TACTILUS II, Univ. d. Saarlandes, 1990

4.24 UDINE

1. *Name der Komponente:* UDINE
2. *Zweck der Komponente:* Unifikator fuer Merkmalslogik
3. *Entwickelt und implementiert durch:* Rolf Backofen, Christoph Weyers
4. *Anschrift, Kontaktperson, e-mail:* Rolf Backofen DFKI Saarbruecken Stuhlsatzenhausweg 3 6600 Saarbruecken 11 EMAIL: backofen@dfki.uni-sb.de
5. *Implementierungssprache:* Common Lisp (Franz Allegro Common Lisp 4.0)
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Common Lisp System
7. *Beginn / Ende der Entwicklung:* Winter 90. Erste stabile Version Fruehjahr 91. Laufende Weiterentwicklungen.
8. *Projekt:* DISCO
9. *Input / Eingabeschnittstellen:* Ascii, graphische Schnittstelle (siehe Feature Editor).
10. *Output / Ausgabeschnittstellen:* Ascii, graphische Schnittstelle (siehe Feature Editor), LaTeX (siehe TDL2LATEX)
11. *Aktueller Zustand:* Es existiert eine stabile Version. Insgesamt befindet sich das System noch in der Weiterentwicklung.
12. *Abhängigkeiten / Anforderungen an andere Systeme:* keine.
13. *Dokumentation:* Die Eingabesyntax liegt in BNF vor, die wichtigsten Benutzerfunktionen sind zusammengefasst und dokumentiert. Eine Kurzdokumentation ist geplant. Arbeiten.
14. *Verfügbarkeit:* Weitergabe fuer wissenschaftliche Zwecke.
15. *Wartbarkeit:* Wegen Verwendung im DISCO Projekt wird der Unifikator staendig gewartet.
16. *Fremdinstallationen:* vorraus. im ASL-Projekt.
17. *Benutzte Techniken und zugrundeliegende Theorien:* Strukturbasierter Unifikator, verteilte Disjunktionen. Der Unifikator arbeitet destruktiv mit chronologischem Ruecksetzen. Desweiteren gibt es Kopierfunktionen fuer interen Strukturen (und damit nicht destruktive Unifikation).

    Der Unifikator ist fuer den Anschluss an eine (weitgehend) beliebige Typverwaltung praepariert.

18. *Allgemeine Beschreibung:*

Der Unifikator ist die zentrale Verarbeitungsmaschine fuer das Projekt DISCO und wird in den meisten der DISCO-Modulen verwendet (Parser, Generator, Typssystem TDL). Er behandelt Merkmalslogik mit Disjunktionen, wobei er auf verteilten Disjunktionen basiert. Desweiteren werden neben Koreferenzen auch negierte Koreferenzen behandelt. Das Eingabemodul kann auch Negation von Strukturen ohne verteilte Disjunktionen verarbeiten, indem diese in Disjunktionen und negierte Koreferenzen uebersetzt werden.

Es existiert eine Funktion zur Herstellung der disjunktiven Normalform. Ausserdem gibt es ein Testool zur Ueberpruefung von Erweiterungen des Unifkators.

Funktionale Constraints werden parteill durch den Unifkator verarbeitet. Die Constraints koennen mit dem Standardein/ausgabemodul verarbeitet werden. Dabei wird eine Liste mit funktionalen Constraints geliefert. Es existiert eine Funktion, die die Auswertbarkeit der Constraints ueberprueft und bei Erfolg das Ergebnis in die Struktur unifiziert.

Derzeit wird an folgenden Erweiterungen gearbeitet: - Depth-first Auswertung von Disjunktionen mit Präferenzbehandlung - Einbindung der Unifikators in einen Prolog-Resultionsbeweiser. (Aehnlich der Sprache LOGIN)

Literaturangaben:

Rolf Backofen and Lutz Euler and Guenther Goerz (1991): Distributed Disjunktions For LIFE, in: Proc. of the Inter. Workshop on Processing Declarative Knowledge (PDK'91), 161-170, Ed: H. Boley and M. M. Richter, Springer Verlag, Berlin

Gert Smolka (1988): A Feature Logic with Subsorts, IBM Deutschland GmbH, LILOG-Report 33, Stuttgart

Gert Smolka (1989): Feature Constraint Logics for Unification Grammars, IBM Deutschland GmbH, LILOG-Report 93, Stuttgart

5 Semantische, Dialog- und Wissensverarbeitung

5.1 Dialogkomponente von NAUDA

1. *Name der Komponente:* Dialogkomponente von NAUDA
2. *Zweck der Komponente:* Erweiterung einer NL-DB-Schnittstelle (LanguageAccess) um kooperative Aspekte: - Erkennung / Korrektur von Praesuppositionsverletzungen bezueglich Anzahl- und Rollenrestriktionen des begrifflichen Wissens - Ueberbeantwortung bei E-Fragen nach numerischen Werten - Information ueber Begriffe, wenn eine zurueckgewiesene Frage nur leicht modifiziert erneut gestellt wird.
3. *Entwickelt und implementiert durch:* Detlef Kuepper
4. *Anschrift, Kontaktperson, e-mail:* Detlef Kuepper FAW, Postf. 2060, 7900 Ulm KUEPPER at DHDIBM1

   bzw. Dr. Dietmar Roesner (Tel.: 0731 / 501 - 530) ROESNER at DULFAW1A
5. *Implementierungssprache:* Prolog (Quintus-Prolog)
6. *Zugrundeliegende Hardware / Betriebssystem(e):* PS/2 / AIX 1.1 oder 1.2
7. *Beginn / Ende der Entwicklung:* 06-91 / 02-92
8. *Projekt:* AUDA (Natuerlichsprachlicher Zugang zu Umweltdatenbanken)
9. *Input / Eingabeschnittstellen:* logische Form von LanguageAccess
10. *Output / Ausgabeschnittstellen:* logische Form von LanguageAccess
11. *Aktueller Zustand:* fertiger Prototyp
12. *Abhängigkeiten / Anforderungen an andere Systeme:* eingebettet in die Lilog Experimentierumgebung LEU/2; verwendet LLilog zur Repraesentation des begrifflichen Wissens

18. *Allgemeine Beschreibung:* s.2

Literaturangaben: zu NAUDA:

R. Becker, W. Gotterbarm, A. Karduck, D. Kuepper, F. Liske, D. Roesner: Natuerlichsprachliches Zugangssystem zu Umweltdatenbanken. In Informatik fuer den Umweltschutz, Proceedings 5. Symposium (Wien, September 1990), Informatik-Fachberichte 256, W. Pillmann, A. Jaeschke, Hrsg. Springer, Berlin, Heidelberg, 1990, 38-46. ebenfalls erschienen als FAW-TR-90019, FAW Ulm

5.2 NLL

1. *Name der Komponente:* NLL
2. *Zweck der Komponente:* SOWOHL semantische Repraesentation ALS AUCH semantische Verarbeitung (semantische Inferenzen)
3. *Entwickelt und implementiert durch:* John Nerbonne, Joachim Laubsch, und Kader Diagne
4. *Anschrift, Kontaktperson, e-mail:* John Nerbonne DFKI Stuhlsatzenhausweg 3 66 Saarbruecken 11
5. *Implementierungssprache:* Zwei Implementierungen: einmal Allegro Common LISP, einmal REFINE
6. *Zugrundeliegende Hardware / Betriebssystem(e):* SUN Sparcs, HP UNIX
7. *Beginn / Ende der Entwicklung:* 1989/1991 (Grundsystem)
8. *Projekt:* HP-NL, DISCO, ASL
9. *Input / Eingabeschnittstellen:* Schnittstellenwerkzeug (Compiler) in NLL hinein existieren! 1. Merkmalsformalismus Leser/Compiler (string Eingaben) ODER 2. Constructor Funktionen, deren Aufruf ueber Syntaxbaeume erfolgt "Rule-to-rule Interpretation in diesem Fall
10. *Output / Ausgabeschnittstellen:* Schnittstellenwerkzeug (Compiler) aus NLL existieren! 1. logische Formeln ODER 2. SQL
11. *Aktueller Zustand:* Forschungssoftware mit fertiger Version - neuere Versionen werden ins Auge gefasst
12. *Abhängigkeiten / Anforderungen an andere Systeme:*
13. *Dokumentation:* unvollstaendig, aber 30 Seiten existieren bereits
14. *Verfügbarkeit:* Copyright HP, und bei Autoren Verbale Zusage seiten HPs, dass das Software zum "Public Domainërkalert wird
15. *Wartbarkeit:* da das Software im Gebrauch bei DISCO und ASL ist, wird es staendig gewartet.
16. *Fremdinstallationen:* sowohl in HP Labs, Palo Alto als auch im DFKI demnaechst in Hamburg (ASL Nord)
17. *Benutzte Techniken und zugrundeliegende Theorien:* Theorien: theorie der generalisierten Quantoren, Plurallogik, aber vgl. Punkt (4) unter "*Allgemeine Beschreibung:* Techniken: Compilerwerkzeug, um (i) das Schnittstellenproblem zu loesen, und (ii) durch Programtransformationen semantische Simplifizierungen (Aequivalenzreduktionen) zu implementieren

18. *Allgemeine Beschreibung:*

NLL ist eine logische Sprache, die im Computer natuerlichsprachliche Bedeutungen darstellt. Ihrer Entwurf verfolgt fuenf Ziele:

(1) ein echtes SemantikMODUL zu liefern, das sowohl unabhaengig als auch in verschiedenen NLP Systemen funktionieren kann; (2) einige semantische Inferenzen zu unterstuetzen; (3) eine Basis fuer Disambiguierung und domaenenspezifische Interpretation (z.B., Transduktion in SQL oder in eine Kommandosprache) zu liefern; (4) das Experimentieren mit verschiedenen semantischen Ideen zu erleichtern. Z.B. existiert eine DRT Darstellung, mit Uebersetzen in standard NLL. (5) die Darstellung von natuerlichsprachlichen Sonderheiten etwas knapper und anschaulicher zu ermoeglichen

Literaturangaben:

Ueberblick (Dokumentation): Joachim Laubsch and John Nerbonne (1991): An Overview of NLL, SSL Report, Hewlett-Packard Laboratories, Palo Alto

Inferenzfaehigkeiten: Joachim Laubsch (1989): Logical Form Simplification, STL Report, Hewlett-Packard

Anwendungsschnittstelle: Joachim Laubsch (1992): The Semantics Application Interface, in: Applied Natural Language Processing, Ed: Hans Haugeneder

Einordnung, Philosophie der Arbeit: John Nerbonne and Joachim Laubsch (1990): Logics for Meaning Representation, unpub. documentation, Hewlett-Packard Laboratories, Palo Alto

Syntaxschnittstelle: John Nerbonne (1992): A Feature-Based Syntax/Semantics Interface, in: Proceedings of the Second International Conference on the Mathematics of Language, Ed: Alexis Manaster-Ramer and Wlodek Zadrozny, Annals of Mathematics and Artificial Intelligence

Spezialthema komplexe Determinatoren: John Nerbonne (1992): Nominal Comparatives and Generalized Quantifiers, in: Processing Plurals and Quantifiers, Ed: Jürgen Allgayer, CSLI, Stanford

5.3 ELF/ELR Repraesentation

1. *Name der Komponente:* ELF/ELR Repraesentation
2. *Zweck der Komponente:* Erzeugung einer formal-semantischen Repraesentation
3. *Entwickelt und implementiert durch:* Siemens ZFE ST SN 74
4. *Anschrift, Kontaktperson, e-mail:* M.Streit, Siemens ZFE ST SN 74, 8000 Muenchen 83 tel: 089/ 636 42283
5. *Implementierungssprache:* Lisp
6. *Zugrundeliegende Hardware / Betriebssystem(e):* TI-Explorer
7. *Beginn / Ende der Entwicklung:* B: 1987 E: 1990
8. *Projekt:* SPICOS II
9. *Input / Eingabeschnittstellen:* Syntaktischer Analyse-Baum
10. *Output / Ausgabeschnittstellen:* Formallogische Representation
11. *Aktueller Zustand:* Entwicklung bis Projektende
12. *Abhängigkeiten / Anforderungen an andere Systeme:* formalsemantische Kompositionsregeln sind gegenwaertig auf SPICOS Grammatik abgestimmt, die Anapherndarstellung erwartet einen Verweis auf die formale Representation des Antezedens.
13. *Dokumentation:* BMfT-Abschlussbericht
14. *Verfügbarkeit:*
17. *Benutzte Techniken und zugrundeliegende Theorien:* modelltheoretische Semantik, ELF/ ELR Sprache
18. *Allgemeine Beschreibung:* Der Formalismus baut auf der in H.Bunt beschrieben formalen Darstellung auf. Aufgrund eines zugrundeliegenden Syntaxbaumes bzw. der syntaktischen Struktur wird weitgehend kompositionell anhand der angwendeten syntaktischen Regeln eine semantische Form aufgebaut, die sich an den natuerlichsprachlichen ausdruecken orientiert (ELF). Dies geschieht unter Zuhilfenahme des ELF Lexicons. Die anaphorischen Bezuege werden in dieser Stufe bereits aufgeloest unter Einbeziehung einer Komponente, die in der Dialoggeschichte anaphorische Beziehungen herstellt (mit syntaktischen Mitteln) und somit auf die formalsemantische Darstellung der Antezedenten verweisen kann. In einem zweiten Schritt wird unter Verwendung eines Transformationslexikonx ELF in ELR kontextfrei uebersetzt. Die so gewonnenen Ader natuerlichsprachlichen Konstanten Ausdruecke der anwendungsabhaenigen Welt, z.B. der Datenbank.
20. *Anzahl und Art der Einträge:* Fuer SPICOS II sind die Einträge fuer die Woerter der Anwendung im ELF und ELR Lexicon implementiert.

Literaturangaben:

H.Bunt: Mass Nouns and Model Theoretic Semantics. Cambridge. Cambridge University Press, 1985

G.Th.Niedermair, M.Streit, H.Tropf: Linguistic Processing Related to Speech Understanding in SPICOS II. in: Speech Communication 9 (1990), North-Holland, pp. 565-585.

M.Streit: Repraesentation von Pluralanaphern. in: Proceedings der 4.OEGAI Konferenz, Wien 1988

M.Streit: Presuppostions and Anaphora in a Question Answering Speech Sys tem. in: J.P.Tubach, J.J.Mariani (eds.),Proceedings of the EUROSPEECH Conference 1989, Paris

5.4 MAUS (Meaning Acquisition And Understanding System)

1. *Name der Komponente:* MAUS (Meaning Acquisition And Understanding System)
2. *Zweck der Komponente:* Empirische Ermittlung von semantischen, dispositionellen Dependenz- strukturen (DDS) aus Corpora natuerlich-sprachlicher Texte
3. *Entwickelt und implementiert durch:* Rieger/Badry/Reichert
4. *Anschrift, Kontaktperson, e-mail:* Petra Badry und Maria Reichert e-mail:badry@ utrurt.uucp.de Fachbereich II (LDV/CL) reichert@utrurt.uucp.de Universitaet Trier Postfach 38 25 5500 Trier
5. *Implementierungssprache:* C / HPGL
6. Zugrundeliegende Hardware/Betriebssystem: CADMUS/MUNIX
7. *Beginn / Ende der Entwicklung:* '86 - offen
8. *Projekt:* SATUS (Semantische Analyse von Texten Und Situationen)
9. *Input / Eingabeschnittstellen:* Natuerlich-sprachliche Texte in lemmatisierter Form
10. *Output / Ausgabeschnittstellen:* gewichtete Baumgraphen semantischer Dependenzen zwischen Wortbedeutungen
11. *Aktueller Zustand:* Grundlagenforschung
12. *Abhängigkeiten / Anforderungen an andere Systeme:* Standard-C, 4MB Hauptspeicher
13. *Dokumentation:* ja
14. *Verfügbarkeit:* Verhandlungssache
16. *Fremdinstallationen:* Max-Planck-Institut fuer Geschichte, Goettingen
17. *Benutzte Techniken und zugrundeliegende Theorien:* Verteilte Repraesentation, Fuzzy Reasoning und Unscharfe Semantik
18. *Allgemeine Beschreibung:* s. *Literaturangaben:*

Literaturangaben:

B. Rieger: Situations and Dispositions. Some formal and empirical tools for semantic analysis, in: Bahner, W./ Schildt, J./ Viehweger, D. (Hrsg): Proceedings of the XIVth International Congress of Linguists 1987 (CIPL), Berlin (Akademie) 1990, Vol.II, S. 1233-1235

B. Rieger: Unscharfe Semantik: numerische Modellierung von Wortbedeutungen als 'Fuzzy'-Mengen, in: Friemel, H.J./ Mueller-Schoenberg, G./ Schuett, A. (Hrsg): Forum '90 -

Wissenschaft und Technik. Neue Anwendungen mit Hilfe aktueller Computer Technologien. (Informatik-Fachberichte 259) Berlin/Heidelberg/New York/London/Paris/Tokyo (Springer) 1990, S. 80–104

B. Rieger: Distributed Semantic Representation of Word Meanings, in: Becker, J. (Hrsg): Proceedings des 6th Workshop on Parallel Processing, Logic, Organization and Technology, (WOPPLOT 89), [im Erscheinen]

B. Rieger: On Distributed Representation and Word Semantics, ICSI-Report TR-91-012, International Computer Science Institute, UC Berkeley, CA 1991

B.Rieger: Reconstructing Meaning from Texts. A Computational View on Natural Language Understanding, in: Raubold, E. (Ed.): Innovative Developments and Applications of Microelectronics and Information Technology, Proceedings 2nd German-Chinese Electronics Week, Shanghai 1991, Berlin/ Offenbach (VDE-Verlag) 1991, S. 193–200

5.5 Evaluationsverfahren zur Anaphern-Resolution

1. *Name der Komponente:* Evaluationsverfahren zur Anaphern- Resolution
2. *Zweck der Komponente:* Finden des besten Antezedenten fuer ein Pronomen durch Anwendung verschiedener Bewertungskriterien und Gewichte. Aktualisierung der Funktor-Argument-Strukturen (FAS).
3. *Entwickelt und implementiert durch:* Guido Dunker
4. *Anschrift, Kontaktperson, e-mail:* Carla Umbach TU Berlin Projekt KIT-FAST, Sekr. FR 5-12 Franklinstr. 28/29 W-1000 Berlin 10 umbach@kit.cs.tu-berlin.de
5. *Implementierungssprache:* Quintus Prolog
6. *Zugrundeliegende Hardware / Betriebssystem(e):* SUN / Unix
7. *Beginn / Ende der Entwicklung:* 1991 - heute
8. *Projekt:* Anapherninterpretation in der maschinellen Uebersetzung (KIT-FAST II, EUROTRA-D Begleitforschung Berlin)
9. *Input / Eingabeschnittstellen:* FAS-Ausdruck ABox des Wissensrepraesentationssystems BACK Kriterien fuer die Anaphern-Resolution Gewichte der Kriterien Reihenfolge der Kriterien
10. *Output / Ausgabeschnittstellen:* aktualisierter FAS-Ausdruck
11. *Aktueller Zustand:* lauffaehige Version 91 existiert, wird aber weiterentwickelt
12. *Abhängigkeiten / Anforderungen an andere Systeme:* benutzt Wissensrepräsentationssystem BACK, entwickelt durch das Projekt KIT-BACK an der TU Berlin (siehe auch entspr. Mitteilung).
13. *Dokumentation:* vorhanden
14. *Verfügbarkeit:* kostenlos gegen Rueckmeldung
15. *Wartbarkeit:* gut - da modularisiert und kommentiert
16. *Fremdinstallationen:* keine
17. *Benutzte Techniken und zugrundeliegende Theorien:* Selbstentwickelte Anaphern-Theorien in Anlehnung an [Pause 86] und [Asher/Wada 87] und anderen.
18. *Allgemeine Beschreibung:* Das Verfahren ist im Moment Teil eines maschinellen Uebersetzungssystems und es wurden bis jetzt Kriterien fuer Possesiv- und Personalpronomina entwickelt. Es kann auch auf andere anaphorische Bezuege erweitert werden.

Literaturangaben:

Birte Schmitz, Susanne Preuss, Christa Hauenschild: Textrepraesentation und Hintergrundwissen fuer die Anaphernresolution im Maschinellen Uebersetzungssystem KIT-FAST, KIT-Report 93, TU Berlin 1992

Christa Hauenschild: Anapherninterpretation in der Maschinellen Uebersetzung, KIT-Report 94, TU Berlin 1991

Peter Pause: Zur Modellierung des Uebersetzungsprozesses, in: I. Batori, H.-J. Weber: "Neue Ansaetze in Maschineller Sprachuebersetzung: Wissensrepraesentation und Textbezug, Niemeyer, Tuebingen 1986, S. 45-74

Nicholas Asher, Hajime Wada: A Computational Account of Syntactic, Semantic and Discourse Principles for Anaphora Resolution, in: Journal of Semantics 6, 1987, S. 309-344

5.6 *MODALYS*

1. *Name der Komponente:* MODALYS
2. *Zweck der Komponente:*

   Mittels einer semantisch-pragmatischen Analyse von Modalverben werden die Systemreaktionen auf modalisierte Eingaben in natürlichsprachlichen Dialogsystemen gesteuert. Eine Teilaufgabe dabei ist die Disambiguierung zwischen den moeglichen Lesarten von Modalverben.
3. *Entwickelt und implementiert durch:* Bernhard Kipper
4. *Anschrift, Kontaktperson, e-mail:* Lehrstuhl Prof. Wahlster, Fachbereich 14: Informatik, Universität des Saarlandes, 6600 Saarbrücken 11, Email: kipper@cs.uni-sb.de
5. *Implementierungssprache: CommonLisp und OPS5*
6. *Zugrundeliegende Hardware / Betriebssystem(e):* keine speziellen Anforderungen
7. *Beginn / Ende der Entwicklung:* 1990 / 1991
8. *Projekt:* SINIX-Consultant (im Rahmen der III-Kooperation zwischen Siemens und der Universität des Saarlandes)
9. *Input / Eingabeschnittstellen:* von einem Parser erzeugte Repräsentationstruktur eines Eingabesatzes
10. *Output / Ausgabeschnittstellen:* transformierte Repräsentationsstruktur
11. *Aktueller Zustand:* Entwicklung abgeschlossen und vollständig implementiert
12. *Abhängigkeiten / Anforderungen an andere Systeme:* kann nach Anpassung an die jeweils verwendetete Repräsentationsstruktur für Eingabesätze in anderen Systemen eingesetzt werden (hohe Flexibilität insbesondere bei Disambiguierungskomponente)
13. *Dokumentation:* Bernhard Kipper, "Semantisch-pragmatische Analyse von Modalverben in natürlichsprachlichen Dialog- und Beratungssystemen", Diplomarbeit, Universität des Saarlandes, 1991
16. *Fremdinstallationen:* keine

5.7 Linguistik-Komponente

1. *Name der Komponente:* Linguistik- Komponente
2. *Zweck der Komponente:* Semantisch-pragmatische Verarbeitungskomponente fuer ein sprachverstehendes System
3. *Entwickelt und implementiert durch:* Lehrstuhl fuer Informatik 5 (Mustererkennung)
4. *Anschrift, Kontaktperson, e-mail:* Prof. H. Niemann Lehrstuhl fuer Informatik 5 (Mustererkennung) Martensstr. 3 8520 Erlangen

   Tel.: 09131/85 7774 Fax.: 09131/303811

   e-mail: niemann@informatik.uni-erlangen.de
5. *Implementierungssprache:* in FRANZ-Lisp
6. *Zugrundeliegende Hardware / Betriebssystem(e):* DEC-CISC- Rechner unter ULTRIX 2.0
7. *Beginn / Ende der Entwicklung:* 1982 / 1990
8. *Projekt:* BMFT
9. *Input / Eingabeschnittstellen:* Gitter von syntaktischen Phrasenhypothesen
10. *Output / Ausgabeschnittstellen:* Interpretation der Aeusserung
11. *Aktueller Zustand:* fertiges Software-Produkt
12. *Abhängigkeiten / Anforderungen an andere Systeme:* nein
13. *Dokumentation:* [Ehr90]
14. *Verfügbarkeit:* ja, auf Anfrage
15. *Wartbarkeit:* Schwierig, da FRANZ-Lisp von DEC nicht mehr unterstuetzt wird und fehlerhaft ist.
16. *Fremdinstallationen:* nein
17. *Benutzte Techniken und zugrundeliegende Theorien:* siehe [Ehr90]
18. *Allgemeine Beschreibung:* siehe [Ehr90]

Literaturangaben:

[EHR90] U. Ehrlich Bedeutungsanalyse in einem sprachverstehenden System unter Berücksichtigung pragmatischer Faktoren Niemeyer, 1990.

.8 Conceptual Modelling Language CML

1. *Name der Komponente:* Conceptual Modelling Language CML
2. *Zweck der Komponente:* Wissensrepraesentationskomponente fuer logikbasierte Modellierung von Domaenen- und Weltwissen, Benutzermodellierung, Dialogmodellierung. Schlussfolgerungskomponenten fuer deduktives Schliessen, Konsistenzpruefung, Defaults, Abduktion; Erstellung von wissensbasierten Anwendungen
3. *Entwickelt und implementiert durch:* CAP debis BeCom
4. *Anschrift, Kontaktperson, e-mail:* Dr. R. Haidan CAP debis BeCom GmbH Oehleckerring 40 D-2000 Hamburg 62 Tel.: 040 53103-312 Fax: 040 53103-574 e-mail: hai@hh.scs.de
5. *Implementierungssprache:* BIMProlog, Portierung auf andere Prolog-Implementierungen moeglich
6. *Zugrundeliegende Hardware / Betriebssystem(e):* SUN / SUN OS Portierung auf andere Umgebungen moeglich
7. *Beginn / Ende der Entwicklung:* 1986 -1989
8. *Projekt:* ESPRIT Projekte P107 LOKI und P892 DAIDA
9. *Input / Eingabeschnittstellen:* a. graphische Entwicklungsumgebung b. Text: Sprache CML c. Programmierschnittstelle zum CML Kernel
10. *Output / Ausgabeschnittstellen:* wie 9.
11. *Aktueller Zustand:* Produkt, Weiterentwicklung im ESPRIT Projekt PLUS (Pragmatics Based Language Understanding)
12. *Abhängigkeiten / Anforderungen an andere Systeme:* BIMProlog Lizenz
13. *Dokumentation:* Manual
14. *Verfügbarkeit:* Vertrieb; Preise auf Anfrage
15. *Wartbarkeit:* durch CAP debis
16. *Fremdinstallationen:* 10
17. *Benutzte Techniken und zugrundeliegende Theorien:* Verbindung von objektorientierten und logikbasierten Techniken; Klassen mit Regeln und Constraints, Vererbung, Metaklassen; Resolutionssystem fuer Hornklauseln + Erweiterung auf "General Clauses"(Negation und Disjunktion); Beweisprozeduren fuer Konsistenzpruefung; Constraints, Integritaetbedingungen, Defaults; Beweisprozeduren fuer Abduktion (Weiterentwicklung);

18. *Allgemeine Beschreibung:* Entwicklungsumgebung fuer Wissensrepraesentationskomponenten

Literaturangaben:

R. Haidan, R. Meyer, Requirements Modelling and System Specification in a Logic Based Knowledge Representation Framework

R. Haidan, R. Meyer, I. Roepke, SML System Modelling Language Support Tool - Reference Manual (Release 3.6)

5.9 LILOG-Inferenzmaschine

1. *Name der Komponente:* LILOG-Inferenzmaschine
2. *Zweck der Komponente:* Inferenzsystem zur Verarbeitung der hybriden Wissensrepraesentationssprache L-LILOG
3. *Entwickelt und implementiert durch:* T.Bollinger, U.Pletat, u.a.
4. *Anschrift, Kontaktperson, e-mail:* bollinge@ds0lilog.bitnet
5. *Implementierungssprache:* Quintus-Prolog
6. *Zugrundeliegende Hardware / Betriebssystem(e):* IBM PS/2, RS/6000, AIX
7. *Beginn / Ende der Entwicklung:* 1989-91
8. *Projekt:* LILOG
9. *Input / Eingabeschnittstellen:* Wissensbasis (Taxonomie+ Regeln+Fakten), Goals bzw. neue Fakten
10. *Output / Ausgabeschnittstellen:* Loesungssubstitutionen bzw, abgeleitete Fakten
11. *Aktueller Zustand:* lauffaehiger Prototyp
12. *Abhängigkeiten / Anforderungen an andere Systeme:* Quintus-Prolog, XWindows
13. *Dokumentation:* ja
14. *Verfügbarkeit:* Copyright IBM, nicht frei verfuegbar, Vereinbarung mit IBM notwendig
15. *Wartbarkeit:* eingeschraenkt
16. *Fremdinstallationen:* > 10
17. *Benutzte Techniken und zugrundeliegende Theorien:* Theorembeweiser basierend auf Modellelimination, ordnungssortierte Unifikation, taxonomisches Schliessen
18. *Allgemeine Beschreibung:* Vorwaerts- und Rueckwaertsinferenzen koennen ausgefuehrt werden, Inferenzkalkuele koennen ausgewechselt werden, Kopplung mit externen Inferenzkomponeten moeglich (in LILOG Depiktionen), Unterstuetzung nichtmonotonen Schliessens u.a. durch TMS

    Ausfuehrliche Beschreibung in "The LILOG Inference Engine", T.Bollinger, U.Pletat, S.Lorenz, in: "Text Understanding in LILOG", LNAI 546

5.10 ERNEST (ERlanger NEtzwerk SysTem)

1. *Name der Komponente:* ERNEST (ERlanger NEtzwerk SysTem)

2. *Zweck der Komponente:* Systemschale zur Repraesentation von deklarativem und prozeduralem Wissen und zur effizienten Verarbeitung (problemunabhaengig, Basis A*-Algoritmus, Bewertungsvektor) des dargestellten Wissens auf der Basis eines semantischen Netzes

3. *Entwickelt und implementiert durch:* Entwickelt und implementiert durch: AG Angewandte Informatik, Universitaet Bielefeld (Prof. Dr. Sagerer, Dr. Kummert) Informatik 5 (Mustererkennung), Universitaet Erlangen (Prof. Dr. Niemann, Dipl.-Inf Prechtel)

4. *Anschrift, Kontaktperson, e-mail:* Prof. Dr. G. Sagerer, AG. Angewandte Informatik, Technische Fakultaet, Postfach 10 01 31, W-4800 Bielefeld 1 e-mail: sagerer@techfak.uni-bielefeld.de

5. *Implementierungssprache:* C

6. *Zugrundeliegende Hardware / Betriebssystem(e):* Hardwareunabhaengig, Unix

7. *Beginn / Ende der Entwicklung:* 1984

9. *Input / Eingabeschnittstellen:* C-Strukturen, z.B. Wortgraphen, Bilder

10. *Output / Ausgabeschnittstellen:* Zwischenrepräsentation (Instanzennetzwerk/benutzerdefinierte Ausgaben)

11. *Aktueller Zustand:* stabile lauffaehige Version vorhanden Verbesserungen ueber update

13. *Dokumentation:* Benutzerhandbuch vorhanden Theoretische Beschreibung ueber Veroeffentlichungen

14. *Verfügbarkeit:* nach Absprache

16. *Fremdinstallationen:* Uni Karlsruhe, Uni Paderborn

17. *Benutzte Techniken und zugrundeliegende Theorien:* Prozedurale semantische Netze, Heuristische Suche (A*-Algorithmus)

18. *Allgemeine Beschreibung:*

    In dem semantischen Netzwerk werden 3 Knoten- und 5 Kantentypen unterschieden. Jeder der 3 Knotentypen wird durch eine Datenstruktur charakterisiert. Es sind dies die Strukturen Konzept, modifiziertes Konzept und Instanz. Konzepte dienen zur Modellierung von Begriffen in der Wissensbasis eines Systems. Instanzen verbinden Konzepte mit einer tatsaechlichen Auspraegung des durch das Konzept modellierten Begriffs im Signal. Sie sind ebenso wie modifizierte Konzepte (Zwischen-)Ergebnisse

eines Analyseprozesses. Waehrend Instanzen eine vollstaendige Zuordnung zwischen einem Konzept und einem Singalausschnitt etablieren repraesentiert der dritte Knotentyp eine Modifikation eines Konzepts aufgrund von Eingabedaten. Damit wird eine Adaption der Wissensbasis an Signalaustritte ermoeglicht, ohne dassa eine vollstaendige Zuordnung aufgebaut wird. Drei der Kantentypen definieren Hierarchien im Netzwerk. Durch Spezialisierungskanten werden Unter-/Oberbegriff-Beziehungen festgelegt. Dekompositionen eines Begriffs in Teile werden durch die Bestandteilskante ermoeglicht. Dabei wird zusaetzlich zwischen kontextabhaengigen und kontextunabhaengigen Bestandteilen unterschieden. Konzepte, die zu unterschiedlichen *Abstraktionsebenen z.B. gemaess dem geschichteten* linguistischen Modell gehoeren, koennen durch Konkretisierungskanten zueinander in Beziehung gesetzt werden. Die beiden anderen Kantentypen dienen der Verbindung zwischen Wissensbasis und Ergebnisspeicher bzw. der automatischen Wissensakquisition. Numerische, messbare Eigenschaften eines Konzepts werden als Attribute, spezielle Zusicherungen zwischen Attributen, Bestandteilen und/oder Konkretisierungen als strukturelle Relationen bezeichnet. Zusaetzlich wird jedem Konzept eine Bewertungsfunktion fuer damit assoziierte Instanzen und modifizierte Konzepte zugeordnet. Bestandteils- und Konkretisierungskanten sowie Attribute und strukturelle Relationen sind selbst ueber komplexe Datenstrukturen definiert, die jeweils von einer Datenstruktur Konzept referiert werden. Neben diesen epistemologischen Primitiven sind weitere ergaenzende Beschreibungen fuer Konzepte moeglich. Eine Unterscheidung verschiedener Gruppierungen von obligatorischen und optionalen Bestandteilen wird durch die Angabe verschiedener Modalitaetsmengen ermoeglicht. Die Angabe von Adjazenzmatrizen dient der kompakten Definition zeitlicher Bezuege zwischen Bestandteilen eines Konzepts. Die Netzwerksprache besitzt zusaetzlich einen prozeduralen Charakter. So koennen an verschiedene Datenstrukturen Berechnungsfunktionen gebunden werden. Global fuer alle Konzepte sind sechs anwendungsunabhaengige Inferenzregeln definiert, die sowohl daten- als auch modellgetriebene Inferenzen subsumieren. Zur Unterstuetzung einer effizienten Analyse sind Einträge zur lokalen Steuerung in das Netzwerk integriert. Lokal bedeutet dabei, daa diese Informationen jeweils an ein Konzept gebunden sind. Die Definition der Netzwerksprache umfasst ihre Syntax, Semantik und Pragmatik.

Der Kontrollalgorithmus basiert auf den sechs Inferenzregeln der Netzwerksprache sowie dem A* - Algorithmus. Die Initialisierung der Suche erfolgt ueber eine Schaetzung potentieller Analyseziele aufgrund von Vorverarbeitungsergebnissen. Jeder nicht verworfene Zustand des Analyseprozesses ist durch eine konsistente Menge von Teilinterpretationen definiert. Konsistent bedeutet dabei, daa eine Teilinterpretation bzgl. der gesamten modellierten Kompetenz des Systems zulaessig ist. Ein Zustand entspricht einer Adaption der gesamten Wissensbasis an bisher interpretierte Bereiche des Signals. Bewertungen derartiger Hypothesen sind so gewaehlt, dass sie deren Zulaessigkeit, Qualitaet, Sicherheit, Relevanz und Prioritaet reflektieren. Die Bewertung von Zustaenden des Suchraums erfolgt generell in Abhaengigkeit des mit dem Zustand assoziierten Analyseziels.

Literaturangaben:

H. Niemann, G. Sagerer, S. Schroeder, F. Kummert: ERNEST: A Semantic Network System for Pattern Understanding, IEEE Transactions on Pattern Analysis and Machine Intelligence, 12: S. 883-905, 1990

G. Sagerer: Automatisches Verstehen gesprochener Sprache, Reihe Informatik, Band 74, Wissenschaftsverlag Mannheim/Wien/Zuerich, 1990

F. Kummert: Flexible Steuerung eines sprachverstehenden Systems mit homogener Wissensbasis, Dissertation, Universitaet Erlangen-Nuernberg, 1991

5.11 ERNEST

1. *Name der Komponente:* ERNEST
2. *Zweck der Komponente:* Eine Software-Umgebung zur wissensbasierten Musteranalyse
3. *Entwickelt und implementiert durch:* Lehrstuhl fuer Informatik 5 (Mustererkennung)
4. *Anschrift, Kontaktperson, e-mail:* Prof. H. Niemann Lehrstuhl fuer Informatik 5 (Mustererkennung) Martensstr. 3 8520 Erlangen

   Tel.: 09131/85 7774 Fax.: 09131/303811

   e-mail: niemann@informatik.uni-erlangen.de
5. *Implementierungssprache:* in C
6. *Zugrundeliegende Hardware / Betriebssystem(e):* DEC- Rechner unter ULTRIX
7. *Beginn / Ende der Entwicklung:* 1982 / laufend
8. *Projekt:* Univ. Erlangen, DFG
9. *Input / Eingabeschnittstellen:* Sensordaten
10. *Output / Ausgabeschnittstellen:* Analyse der Sensordaten
11. *Aktueller Zustand:* laufendes Software-Produkt
12. *Abhängigkeiten / Anforderungen an andere Systeme:* nein
13. *Dokumentation:* [NSS90]
14. *Verfügbarkeit:* ja, auf Anfrage
16. *Fremdinstallationen:* nein
17. *Benutzte Techniken und zugrundeliegende Theorien:* siehe [NSS90]
18. *Allgemeine Beschreibung:* siehe [NSS90]

Literaturangaben:

[NSS90] H. Niemann, G. Sagerer, S. Schröder, F. Kummert ERNEST: A Semantic Network for Pattern Understanding IEEE Trans. on Pattern Analysis and Machine Intelligence 12/9, 1990.

5.12 LEU/2, LILOG-KR, Ergaenzungen zur Integration von Wissenspaketen

1. *Name der Komponente:* LEU/2, LILOG-KR, Ergaenzungen zur Integration von Wissenspaketen

2. *Zweck der Komponente:* Aufbau strukturierter Hintergrundwissensbasen (Ontologie und Axiomatik)

3. *Entwickelt und implementiert durch:* Dem Bielefelder LILOG-Team gehoerten an: Barbara Gaengler (Mitarb.), Marianne Greten (Stud.), Thomas Linke (Stud.), Ipke Wachsmuth (verantw.)

4. *Anschrift, Kontaktperson, e-mail:* Technische Fakultaet, Uni Bielefeld, 4800 Bielefeld Kontaktpersonen: Ipke Wachsmuth (Tel. 0521/106-2910, ipke@techfak.uni-bielefeld.de), Barbara Gaengler (Tel. 0521/106-2919, barbara@techfak.uni-bielefeld.de) Die Erweiterungen wurden in Kooperation mit IBM Stuttgart implementiert; bester Ansprechpartner fuer Technisches: *Jan Wilms-Harutunian, Muenchner Str.* 67, 8011 Kirchheim, Tel. 089/9043063

5. *Implementierungssprache:* L-LILOG (auf Quintus-Prolog aufbauend)

6. *Zugrundeliegende Hardware / Betriebssystem(e):* PS/2 Model 80, AIX

5.13 BACK: Berlin Advanced Computational Knowledge Representation System

1. *Name der Komponente:* BACK: Berlin Advanced Computational Knowledge Representation System Version 4.4 - 4. October 1991

2. *Zweck der Komponente:* Repräsentation terminologischen und assertionalen Wissens (Sprach- und Weltwissen) in der Tradition terminologischer Logiken (KL-ONE).

3. *Entwickelt und implementiert durch:* TU Berlin Projektgruppe BACK (zur Zeit Carsten Kindermann, Christof Peltason, Joachim Quantz, Albrecht Schmiedel)

4. *Anschrift, Kontaktperson, e-mail:* Joachim Quantz, TU Berlin, Projektgruppe KIT, FR 5-12 Franklinstr. 28/29 1000 Berlin 10 jjq@cs.tu-berlin.de

5. *Implementierungssprache:* Prolog (Quintus,C-Prolog) (C++ Implementierung in Vorbereitung)

6. *Zugrundeliegende Hardware / Betriebssystem(e):* SUN Sparc station SLC (Kernsystem betriebssystemunabhaengig)

7. *Beginn / Ende der Entwicklung:* 1.1.1985 / 31.10.1993

8. *Projekt:* ESPRIT Projekt 5210 AIMS (Advanced Information Management System)

9. *Input / Eingabeschnittstellen:* Textuelle Schnittstellensprache in Prolog (Graphikschnittstelle in Vorbereitung)

10. *Output / Ausgabeschnittstellen:* Textuelle Schnittstellensprache in Prolog (Graphikschnittstelle in Vorbereitung)

11. *Aktueller Zustand:* in der Entwicklung, prototypischer Einsatz bei Projektpartnern

12. *Abhängigkeiten / Anforderungen an andere Systeme:* keine (Datenbankkopplung in Vorbereitung)

13. *Dokumentation:* Technische Reports mit Referenzkarte (Tutorial in Vorbereitung)

14. *Verfügbarkeit:* frei fuer registrierte Benutzer (eine experimentelle Version mit Erweiterungen, Micro-Back ist ebenfalls erhaeltlich)

15. *Wartbarkeit:* Fehlermeldungen werden dankbar entgegengenommen

16. *Fremdinstallationen:* 6 x Projektpartner 20 x prototypischer Einsatz bei Forschungsinstitutionen

17. *Benutzte Techniken und zugrundeliegende Theorien:* Terminologische Logiken

18. *Allgemeine Beschreibung:* Prototypische Implementation eines logisch fundierten Wissensrepraesentationsystems. Striktes Sprach- und Weltwissen kann repraesentiert werden. Das System berechnet implizit gegebene Konsequenzen (Konsistenzueberpruefung, Vererbung, Propagierung, Vervollstaendigung). Gegenwaertig wird die Integration von Zeitlogiken und Defaults untersucht.

Literaturangaben: siehe KIT-Publikationsliste

C. Peltason, A. Schmiedel, C. Kindermann, and J. Quantz: 'The BACK System Revisited.' KIT Report 75, Technische Universität Berlin, September 1989.

J. Quantz and C. Kindermann: 'Implementation of the BACK System Version 4.' KIT Report 78, Technische Universität Berlin, September 1990.

5.14 Semantische Constraint-Modellierung

1. *Name der Komponente:* Semantische Constraint- Modellierung

2. *Zweck der Komponente:* Prueft gleichzeitig mit der syntaktischen Analyse, ob bestimmte allgemeine, und domainspezifische Constraints zwischen den Phrasen oder Phrasenteilen erfuellt sind, um so falsche akustische Hypothesen zu eliminieren und falsches Attachment in der Analyse zu verhindern. Die semantischen Beziehungen werden auch waehrend der Anaphernresolution genutzt: erstens um Pronomen eine temporaere, vom Satzkontext abhaengige semantische Kategorie zuzuweisen; zwietens: um die semantische Vertraeglichkeit von Antezedenz-Beziehungen zu gewaehrleisten.

3. *Entwickelt und implementiert durch:* Siemens ZFE ST SN 74

4. *Anschrift, Kontaktperson, e-mail:* G.Niedermair, ZFE ST SN 74 Otto-Hahn-Ring 6 8000 Muenchen 83 tel: 089/ 636 2374 e-mail: nie

5. *Implementierungssprache:* Prolog

6. *Zugrundeliegende Hardware / Betriebssystem(e):* TI Explorer

7. *Beginn / Ende der Entwicklung:* 1987 1991

8. *Projekt:* SPICOS II, SUNDIAL

9. *Input / Eingabeschnittstellen:* Funktionsaufruf mit den semantischen Typen der zu pruefenden Konstituente zuzueglich syntaktischer Merkmale als Parameter

10. *Output / Ausgabeschnittstellen:* t/f Wert fuer die semantische Korrektheit

11. *Aktueller Zustand:* Wurde fuer SPICOS II und SUNDIAL vollstaendig implementiert und an umfangreichem Material getestet. Wird gegenwaertig an den Unifikationsformalismus des LKP von Siemens angepasst.

12. *Abhängigkeiten / Anforderungen an andere Systeme:* Da es parell zu der syntaktischen Analyse laeuft und in Einklang mit den syntaktischen Regelanwendungen, muessen entspr. syntaktische wie semantische Merkmale zum Analysezeitpunkt ansprechbar sein.

13. *Dokumentation:* BMFT-Abschlussbericht, Sundial-deliverables

14. *Verfügbarkeit:* ?

17. *Benutzte Techniken und zugrundeliegende Theorien:* semantische Netze, Typenhierarchien, Regeln, die die erlaubten Transitionen in einem semantischen Netz beschreiben. Moegliche Transitionen stellen gueltige semantische Selektionen dar.

18. *Allgemeine Beschreibung:* Das semantische Netzwerk ist weitgehend abhaenig von der Domaene (FahrplanauskunfEs enthaelt zu beiden Domaenen ca. 100 - 200 semantische Konzepte, die in hierarchischen und anderen Beziehungen zueinander stehen. Regeln fuer die Korrektheit semantischer Beziehungen sind implementiert fuer nomen-verb Beziehungen, nomen-pp Beziehungen, nomen-nomen_genitiv, nomennamen (in der Apposition), verb-adverb. Die semantischen Beziehungsbeschreibungen wurden im SUNDIAL Projekt an ca. 1000 Aesserugen getestet. Die semantischen Tests sind als Funktionen in die Grammatik integrierbar und koennen so wie andere Merkmalstests gleichzeitig mit der Verarbeitung der Phrasenstrukturregeln als Constraints genutzt werden.

19. *Entwicklungsumgebung zur Erstellung der Einträge:* Die Einträge der semantischen Klassen der Woerter, bzw. ihre Zugehoerigkeit zu bestimmten Concepten erfolgt ueber das Lexikon.(s.o) Das Lexikon-Tool, wie fuer die Syntaxanalyse in SPICOSII beschrieben, unterstuetzt auch diese Eintragungen.

20. *Anzahl und Art der Einträge:* Einträge bestehen vollstaendig fuer die jeweils 1000 - 1200 Woerter der beiden Anwendungen, ebenso sind die semantischen Netze fuer beide Anwendungen vollstaendig aufgebaut.

Literaturangaben:

G.Th.Niedermair: The Use of a Semantic Network in Speech Dialogue. in: J.P.Tubach, J.J.Mariani (eds.),Proceedings of the EUROSPEECH Conference 1989, Paris, pp: 26-29

5.15 SBLITTERS

1. *Name der Komponente:* SBLITTERS
2. *Zweck der Komponente:* erweiterte ABox fuer SBONE (Termdefinitionssprache)
3. *Entwickelt und implementiert durch:* Juergen Allgayer
4. *Anschrift, Kontaktperson, e-mail:* ali@helen.uni- sb.de
5. *Implementierungssprache:* Lisp, Prolog
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Common- Lisp auf irgendwelchen Rechnern
7. *Beginn / Ende der Entwicklung:* Beginn 1988, unter Entwicklung
8. *Projekt:* SFB314:N1 XTRA / PRACMA
11. *Aktueller Zustand:* in Entwicklung
12. *Abhängigkeiten / Anforderungen an andere Systeme:*
13. *Dokumentation:* SBLITTERS Handbuch, Memo (noch in Bearbeitung)
14. *Verfügbarkeit:* PD
15. *Wartbarkeit:* ??
16. *Fremdinstallationen:* keine
17. *Benutzte Techniken und zugrundeliegende Theorien:* Termdefinitionssprachen; Prolog-Verarbeitung;
18. *Allgemeine Beschreibung:*

    SBLITTERS stellt einen Formalismus zur Verfuegung, der auf eine gegebene (aber im prinzip beliebige) Termdefinitionssprache abbilden kann und den Aufbau und die Abfrage der aufgebauten Abox unterstuetzt. Dabei werden Erweiterungen integriert, die auf eine semantische Repraesentationssprache zielen. Dies sind z.B.

    - die Integration von Mengen als epistemologische Primitiva; damit koennen Plurale beschrieben werden; Partitivkonstruktionen sind darstellbar;
    - Reasoning ueber Mengen ist integriert
    - Lesarten (distributiv/kollektiv/comulativ) konnen abgebildet werden
    - Diskursoperatoren (iota, eta, alpha [fuer attributive Beschreibungen], pi [fuer prototypische Beschreibungen]) sind teilweise integriert
    - Anfrage mit komplexen Pattern werden durch Prolog-Metainterpreter unterstuetzt

- Abbildung und Verarbeitung Generalisierter Quantoren (teilweise realisiert)

Literaturangaben:

J. Allgayer: SBONE+ – how to deal with sets efficiently. In: ECAI-90

J. Allgayer: SBLITTERS-Handbuch. Memo, Univ. d. Saarlandes, to appear.

J. Allgayer, C. Reddig: What's in a 'DET'? Steps towards determiner-dependant

5.16 MAIDAI

1. *Name der Komponente:* MAIDAI
2. *Zweck der Komponente:* Begruendungsverwaltende ABox fuer Termdefinitionssprachen (bevorzugt solche, die Default Information beschreiben koennen)
3. *Entwickelt und implementiert durch:* Reinhold Schmitt
4. *Anschrift, Kontaktperson, e-mail:* ali@helen.uni-sb.de
5. *Implementierungssprache:* LISP, Except-II
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Common- Lisp Rechner
7. *Beginn / Ende der Entwicklung:* Diplomarbeit 1992
8. *Projekt:* Innrhalb des Projektes XTRA (SFB314:N1)
11. *Aktueller Zustand:* Abgeschlossene Diplomarbeit
13. *Dokumentation:* siehe Literatur
14. *Verfügbarkeit:* nach Absprache
15. *Wartbarkeit:* ??
16. *Fremdinstallationen:* keine
17. *Benutzte Techniken und zugrundeliegende Theorien:* Reason Maintenance Techniken; Except-II System von Uli Junker (GMD Bonn)
18. *Allgemeine Beschreibung:*

    MAIDAI ist eine begruendungsverwaltende ABox, die an beliebige Termdefinitionssysteme angeschlossen werden kann. Sie interpretiert die Konzeptdefinitionen als Constraints, deren Gueltigkeit waehrend der Lebensdauer einer ABox beruecksichtigt werden. Ruecknahmen von Fakten sind moeglich; erforderliche Ruecknahmen von abgeleiteten Fakten werden unterstuetzt.

Literaturangaben:

R. Schmitt: MAIDAI: Ein System zur Begruendungsverwaltung von Default Assertionen. Diplomarbeit, Univ. d. Saarlandes, 1992

5.17 SPREADIAC - a SPREADIng ACtivator

1. *Name der Komponente:* SPREADIAC - a SPREADIng ACtivator

2. *Zweck der Komponente:*

   SPREADIAC ist ein Spreading Activation System, das auf KL-ONEartigen Wissensnetzen arbeitet. Dabei muessen die Kanten dieser Wissensnetze nach dem Aspekt der semantischen Naehe gewichtet sein. Ein besonderes Merkmal von SPREADIAC besteht darin, dass die Pfadsuche effizient ablaeuft, da sie durch Pfadmuster gesteuert wird.

   Ein in sprachverarbeitenden Systemen auftretendes Problem ist die definite Referenz auf implizit eingeführte Objekte. Dabei müssen explizite Relationen zwischen nur indirekt verbundenen Strukturen gemäß einer zugrundeliegenden Wissensbasis gefunden bzw. bewertet werden. Das durch diese Pfadsuche abgeleitete Wissen kann in die Wissensbasis aufgenommen werden.

   Die Auflösung der beschriebenen Referenzen ist die Hauptanwendung von SPREADIAC. Daneben kann das System bei der Auflösung lexikalischer Mehrdeutigkeiten und zur Bedeutungsanalyse von Nominalkomposita eingesetzt werden.

3. *Entwickelt und implementiert durch:* Ralph Schäfer

4. *Anschrift, Kontaktperson, e-mail:* Ralph Schäfer, ralph@cs.uni- sb.de, Universität des Saarlandes, FB14 - Informatik, Lehrstuhl Prof. Wahlster

5. *Implementierungssprache:* COMMON-LISP

6. *Zugrundeliegende Hardware / Betriebssystem(e):* Laeuft auf Solbournes und Symbolics LISP-Boards. Betriebssystem der Solbournes ist UNIX.

7. *Beginn / Ende der Entwicklung:* System ist seit 1991 fertig implementiert.

9. *Input / Eingabeschnittstellen:* Graphikschnittstelle fuer stand- alone Betrieb. Ansonsten aufrufbar als LISP-Funktion.

10. *Output / Ausgabeschnittstellen:* Ausgabe als LISP-Funktionswert. Weitergehende Informationen in globaler Variable abgelegt.

11. *Aktueller Zustand:* Implementiert.

12. *Abhängigkeiten / Anforderungen an andere Systeme:* Benutzt SB- ONE, kann aber auf andere KL-ONEartige Systeme portiert werden.

13. *Dokumentation:* Saemtliche Funktionen dokumentiert, zusaetzlich genereller Ablauf in Diplomarbeit beschrieben.

14. *Verfügbarkeit:* Preis: Kopiergebuehr

15. *Wartbarkeit:* Saemtliche Funktionen sind mit einer funktionalen Spezifikation kommentiert, so dass das System leicht wartbar ist.

16. *Fremdinstallationen:* keine

17. *Benutzte Techniken und zugrundeliegende Theorien:* Spreading Activation

18. *Allgemeine Beschreibung:* siehe 2. Eine allgemeine Beschreibung ueber Spreading Activation findet sich beispielsweise in

    J. Diederich. *Spreading Activation and Connectionist Models for Natural Language Processing.* Technical Report TR-89-008, International Computer Science Institute, Berkeley, California, Februar 1989.

Literaturangaben:

Ralph Schäfer (1991): SPREADIAC - ein Spreading Activation-System auf KL-ONEartigen Wissensnetzen zur Verarbeitung impliziter Referenzen, Diplomarbeit am Lehrstuhl Prof. Wahlster, Fachbereich Informatik, Universität des Saarlandes

6 Generierung

6.1 KLEIST

1. *Name der Komponente:* KLEIST

2. *Zweck der Komponente:* (Text-) Generierung von Wegbeschreibungen in deutscher und japanischer Sprache

3. *Entwickelt und implementiert durch:* J. Meier, D. Metzing, K. Peters, T. Polzin, H. Rutz, M. Siegel

4. *Anschrift, Kontaktperson, e-mail:* D. Metzing, Fakultät für Linguistik und Literaturwissenschaft, Universität Bielefeld. e-mail: metzing@lili1.uni- bielefeld.de

5. *Implementierungssprache:* C-Prolog, CheOPS (objektorientierte Programmierumgebung)

6. *Zugrundeliegende Hardware / Betriebssystem(e):* SUN-4 Workstation, UNIX

7. *Beginn / Ende der Entwicklung:* 6/86 - 5/91

8. *Projekt:* Projekt Textgenerierung, Forschergruppe Kohärenz, Fakultät für Linguistik und Literaturwissenschaft, Universität Bielefeld

9. *Input / Eingabeschnittstellen:* Start- und Zielpunkt für den zu beschreibenden Weg

10. *Output / Ausgabeschnittstellen:* Deutscher oder japanischer Text

11. *Aktueller Zustand:* Produkt

13. *Dokumentation:* Kolibri 26, siehe *Literaturangaben:*

15. *Wartbarkeit:* Das Gesamtsystem ist erweiterbar (z.B. Lexikon/Grammatik, Wissen ueber Modellstadt)

17. *Benutzte Techniken und zugrundeliegende Theorien:* LFG, objektorientierte Programmierung

18. *Allgemeine Beschreibung:* 4 Phasen:

    1. Phase: Wegsuche (Input / Eingabeschnittstellen: Start- und Zielpunkt, Output / Ausgabeschnittstellen: Liste aus Wegsegmenten)

    2. Phase: Message-Planung (Input / Eingabeschnittstellen: Liste aus Wegsegmenten, Output / Ausgabeschnittstellen: Message)

    3. Phase: Planung der sprachlichen Form (Input / Eingabeschnittstellen: Message, Output / Ausgabeschnittstellen: Situationsschema)

    4. Phase: Oberflächengenerierung (Input / Eingabeschnittstellen: Situationsschema, Output / Ausgabeschnittstellen: Satz)

Die 1. Phase bezieht sich auf den gesamten Text, die anderen Phasen auf jeweils eine Äußerung.

19. *Entwicklungsumgebung zur Erstellung der Einträge:* Compiler, der LFG-Grammatik und Lexikon in Prolog ueberfuehrt

Literaturangaben:

Meier, J., Metzing, D., Polzin, T., Ruhrberg, P., Rutz, H. und Vollmer, M. (eds.) (1988), Generierung von Wegbeschreibungen. Kolibri Arbeitsberichte der Forschergruppe "Kohärenz", Nr.9. Bielefeld: Universität Bielefeld.

Metzing, D. (1989), Zur Modellierung von Kohärenzprozessen. In: W. Brauer und C. Freksa (eds.), Wissensbasierte Systeme. 3. internationaler GI-Kongreß, München; Proceedings (pp. 131-138). Berlin: Springer.

Metzing, D., Peters, K., Rutz, H., Siegel, M. (1991), Rekonstruktion von Verfahren der Textproduktion. In: Rickheit, G. (ed.), Kohärenzprozesse. Modellierung von Sprachverarbeitung in Texten und Diskursen (pp. 183-241). Opladen: Westdeutscher Verlag.

Peters, K., Rutz, H. und Siegel, M. (eds.) (1991), KLEIST - Textgenerierung in deutscher und japanischer Sprache. Kolibri Arbeitsberichte der Forschergruppe "Kohärenz", Nr.26. Bielefeld: Universität Bielefeld.

Ruhrberg, P. und Rutz, H. (1989), Räumliches Wissen und Semantik im Kontext der Generierung von Wegbeschreibungen. In: C. Freksa und C. Habel (eds.), Repräsentation und Verarbeitung räumlichen Wissens (pp. 235-250). Berlin: Springer.

Ruhrberg, P., Türling, H.-J. und Zimmermann, B. (eds.) (1990), Interaktion von Syntax und Semantik in der Textgenerierung. Kolibri Arbeitsberichte der Forschergruppe "Kohärenz", Nr.27. Bielefeld: Universität Bielefeld.

Rutz, H. (1990), Aspekte der Generierung von Wegbeschreibungen: Lokale und globale Kohärenz. In: Forschergruppe "Kohärenz"(ed.), Kohärenz. Kolibri Arbeitsberichte der Forschergruppe "Kohärenz", Nr.1 (pp.51-62). Bielefeld: Universität Bielefeld.

Rutz, H. (1990), Kohärenz konstituierende Prozesse in der Generierung von Wegbeschreibungen. In: W. Hoeppner (ed.), Räumliche Alltagsumgebungen des Menschen. Fachbericht Informatik (pp. 133-146). Koblenz: Universität Koblenz-Landau.

6.2 KOMET

1. *Name der Komponente:* KOMET
2. *Zweck der Komponente:* Multilinguale Textgenerierung und Textplanung.
3. *Entwickelt und implementiert durch:* Projekt KOMET, GMD/Institut fuer Intergrierte Publikations- und Informationssysteme.

   John Bateman, Elisabeth Maier, Elke Teich, Leo Wanner, Renate Henschel.
4. *Anschrift, Kontaktperson, e-mail:* John Bateman, GMD/IPSI, Dolivostr. 15, 6100 Darmstadt. *bateman@ipsi.darmstadt.gmd.de*
5. *Implementierungssprache:* Common LISP
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Symbolics Lisp machines, SUN (Sparcs, Common Lisp), TI Lisp Machines, MacIntosh.
7. *Beginn / Ende der Entwicklung:* Beginn: Oct 1989. Ende: wenn alles geloest ist.....:-)
8. *Projekt:* KOMET
9. *Input / Eingabeschnittstellen:* Abstrakte Spezifikationen von kommunikativen Zielen, Hintergrundwissen, ein Modell von Sprecher und Hoerer
10. *Output / Ausgabeschnittstellen:* text (strings).
11. *Aktueller Zustand:* Forschungs-Prototypen, die die Basis fuer weitere Entwicklungen darstellen.
12. *Abhängigkeiten / Anforderungen an andere Systeme:* Das Textgenerierungssystem basiert unter anderem auf der Repraesentation von Wissen in LOOM; das System kann aber aufgrund einer genau definierten Schnittstelle leicht an andere WR-Formalismen adaptiert werden. Systeme, die den Satzgenerator zur Erzeugung von natuerlichsprachigem Output verwenden wollen, muessen die Generatoreingabe in SPL (Sentence Plan Language) spezifizieren.
13. *Dokumentation:* teilsteils.
14. *Verfügbarkeit:* kostenlos, softwarelizenz erforderlich
15. *Wartbarkeit:* je nach Vertrautheitsgrad mit dem System....
16. *Fremdinstallationen:* Deutsche systemische Grammatik: Uni Hildesheim, FAW-Ulm, ISI (Los Angeles). Textplanungskomponente (ohne Lexikalisierung): ISI
17. *Benutzte Techniken und zugrundeliegende Theorien:* Zugundeliegende Theorie fuer das Gesamtsystem: Systemische-Funktionale Linguistik (SFG)

    fuer die Einzelkomponenten:

    * Grammatik: SFG * Textplanung: SFG, Rhetorical Structure Theory (RST) * Lexikalisierung: SFG, Meaning Text Model (MTM)

18. *Allgemeine Beschreibung:*

    Das KOMET System stellt ein integriertes System zur Generierung von monologischen Texten aus der Beschreibung von Textthemen und Kommunikativen Zielen dar. Die Einzelkomponenten des Textgenerierungssystems sind:

    * Textplanung / Textstrukturierung und Inhhaltsbestimmung Die Komponente besteht aus verschiedenen deklarativ repraesentierten Wissensquellen, die mit Hilfe eines zentralen Kontrollprozesses ge- steuert werden. Die Wissenquellen - kommunikative Ziele, Texttypen, rhetorische Relationen,..- und die Abhängigkeiten zwischen den Wissensquellen determinieren die verwendeten linguistischen Mittel. Die Ausgabe dieser Komponente besteht aus einer textuell strukturierten Kette von Konzepten. * Textplanung / Lexikalisierung Die Lexikalisierungskomponente bestimmt die jeweils fuer eine gegebene Situation geeignetsten Lexeme. Zu diesem Zweck wurden Entscheidungs- Netzwerke im Formalismus der SFG entwickelt, die fuer jedes Konzept nach einem Durchlauf ein Lexem waehlen. Die Komponente erzeugt als Output eine textuell strukturierte Kette von Lexemen, die auf eine SPL Eingabe fuer den Satzgenerator abgebildet wird. * Grammatik In Analogie zu der NIGEL Grammatik fuer das Englische wurde eine umfangreiche Grammatik fuer das Deutsche entwickelt.

19. *Entwicklungsumgebung zur Erstellung der Einträge:* ja.

20. *Anzahl und Art der Einträge:* Grammatische Oppositionen: 900 (grammatical systems) Semantische Schnittstelle Kategorien: 600 (inquiries) Semantische Schnittstelle Bedingungen: 600 (choosers) Semantische Ontologie: 300 Konzepte (upper model) Lexikalische Einträge: Domaenabhaengig.

Literaturangaben:

John A. Bateman and Maier, Elisabeth and Elke Teich and Leo Wanner (1991): Towards an Architecture for Situated Text Generation, in: International Conference on Current Issues in Computational Linguistics, Penang, Malaysia, Also available as technical report of GMD/Institut für Integrierte Publikations- und Informationssysteme, Darmstadt, West Germany

Eduard Hovy and Julia Lavid and Elisabeth Maier and Vibhu Mittal and Cécile Paris (1992): Employing Knowledge Resources in a New Text Planner Architecture, in: Proceedings of the 6th International Workshop on Natural Language Generation

Leo Wanner and Elisabeth A. Maier (1991): Lexical Choice as an Integrated Component of Situated Text Planning, in: Proceedings of the 3rd European Natural Language Generation Workshop, Innsbruck, March, 1991, Also available as technical report of GMD/Institut für Integrierte Publikations- und Informationssysteme, Darmstadt, West Germany

Elke Teich (1991): A systemic grammar of German for text generation, in: Papers from

the Seventeenth International Systemic Congress, University of Stirling, July 1990, Ed: Martin Davies and Louise Ravelli

6.3 Konnektionistisches Modell der Sprachproduktion

1. *Name der Komponente:* Konnektionistisches Modell der Sprachproduktion

2. *Zweck der Komponente:* Modellierung von Teilprozessen des kognitiven Prozesses der Sprachproduktion

3. *Entwickelt und implementiert durch:* Dr. Ulrich Schade Fakultaet fuer Linguistik und Literaturwissenschaft Universitaet Bielefeld Universitaetsstrasse 25 4800 Bielefeld 1

4. *Anschrift, Kontaktperson, e-mail:* siehe 3. e-mail: schade@lili2.uni-bielefeld.de

5. *Implementierungssprache:* CheOPS — objektorientierter PROLOG-Dialekt, entwickelt von: Dr. Hans-Juergen Eikmeyer, Adresse wie in 3. e-mail: eikmeyer@lili3.uni-bielefeld.de

6. *Zugrundeliegende Hardware / Betriebssystem(e):* hp workstation bzw. SUN 3 / UNIX

7. *Beginn / Ende der Entwicklung:* Sept. 1986 - Maerz 1990

8. *Projekt:* DFG-Forschergruppe Kohaerenz, Teilprojekt „Gesprochenes Deutsch“

   Literatur zu einigen Ergebnissen der Forschergruppe findet sich in Rickheit, G. (Hrsg.) (1991). "Kohaerenzprozesse - Modellierung und Sprachverarbeitung in Texten und Diskursen". Opladen: Westdeutscher Verlag.

9. *Input / Eingabeschnittstellen:* Aktivierung von Knoten des semantischen Teilnetzes Die Eingabe kann interaktiv oder durch das Einlesen einer Datei vorgenommen werden.

10. *Output / Ausgabeschnittstellen:* String von Graphemen bzw. von Phonemsymbolen Die Ausgabe erfolgt wahlweise auf den Bildschirm oder in eine Datei.

11. *Aktueller Zustand:* abgeschlossen; wird fuer Simulationen genutzt

12. *Abhängigkeiten / Anforderungen an andere Systeme:* C-PROLOG

13. *Dokumentation:* Schade, Ulrich (1992). "Konnektionismus - Zur Modellierung der Sprachproduktion". Opladen: Westdeutscher Verlag.

    Das Buch enthaelt einen etwa 30-seitigen Anhang, in welchem das zugrundeliegende Programm dokumentiert ist.

14. *Verfügbarkeit:* ?

15. *Wartbarkeit:* ?

16. *Fremdinstallationen:* wurde bislang noch nicht versucht

17. *Benutzte Techniken und zugrundeliegende Theorien:* Theorie der lokalen konnektionistischen Modelle (ßpreading activation")


UNIVERSITAS
SARAVIENSIS

18. *Allgemeine Beschreibung:* siehe 13.

19. *Entwicklungsumgebung zur Erstellung der Einträge:*

    Einträge fuer linguistische Einheiten aller Art werden durch Knoten des Netzwerkes repraesentiert. Fuer neue Einträge muss in der entsprechenden Netzwerkschicht ein Knoten definiert und dessen Verknuepfung mit anderen Knoten angegeben werden. Da ohnehin fuer unterschiedliche Simulationen unterschiedliche Netzwerke benutzt werden, wird das Netzwerk in zwei Dateien unabhaengig vom Restprogramm spezifiziert. Diese Dateien werden waehrend der Simulation vom Steuerungsprogramm konsultiert.

20. *Anzahl und Art der Einträge:* Bisherige Simulationen umfassten maximal 70 Wortknoten und damit etwa bis zu 250 Knoten fuer linguistische Einheiten.

Literaturangaben:

Vgl. die in 13. und 8. gemachten Angaben. Weitere Papiere sind in den genannten Buechern zitiert.

6.4 POPEL

1. *Name der Komponente:* POPEL
2. *Zweck der Komponente:* Generierungssystem fuer multimodale Dialogbeitraege
3. *Entwickelt und implementiert durch:* Dr. N. Reithinger, G. Neumann, W. Finkler, A. Kresse, R. Schaefer, J. Jung
4. *Anschrift, Kontaktperson, e-mail:* Dr. N. Reithinger SFB 314 - FB 14 Informatik IV Univ. des Saarlandes Saarbruecken e-mail: bert@cs.uni-sb.de
5. *Implementierungssprache:* Kern: Common Lisp mit CLOS Multimodale Ausgabe: derzeit GENERA 8.1 Graphik umstellung auf CLIM geplant
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Jede Plattform, auf der Common Lisp mit CLOS (und CLIM) ablauffaehig ist
7. *Beginn / Ende der Entwicklung:* 1985 - jetzt
8. *Projekt:* XTRA/PRACMA
9. *Input / Eingabeschnittstellen:* Intentionales Ziel, dass sich aus Elementen einer Wissensrepraesentation zusammensetzt
10. *Output / Ausgabeschnittstellen:* Text, Graphik
11. *Aktueller Zustand:* Prototyp
12. *Abhängigkeiten / Anforderungen an andere Systeme:* Wissensrepraesentationssystem SBLITTERS
13. *Dokumentation:* Dissertation und Diplomarbeiten
14. *Verfügbarkeit:* Copyright: SFB 314, Univ. des Saarlandes Preis: Kopiergebuehr
15. *Wartbarkeit:* Maessig
16. *Fremdinstallationen:* Univ. Bielefeld
17. *Benutzte Techniken und zugrundeliegende Theorien:*

    Rhetorcial Structure Theory, Planung, Wissensrepraesentation mit einer Termdefinitionssprache, unifikationsbasierte syntaktische Verarbeitung, Parallelverarbeitung
18. *Allgemeine Beschreibung:*

    POPEL ist ein leistungsfähiges und flexibles Generierungssystem für multimodale Dialogbeiträge.

    Der Schwerpunkt bei der Entwicklung war die Untersuchung des Architekturmodells und der Verarbeitungsstrategie eines Generierungssystems, damit es sowohl hinsichtlich der Flexibilität als auch der Geschwindigkeit eine wesentliche Verbesserung gegenüber bereits bestehenden Systemen verspricht. Das System wurde auf der Basis

einer gegebenen Anwendungssituation in einem natürlichsprachlichen Dialogsystem entwickelt.

Die Evaluation mehrerer bereits implementierter Systeme und die Analyse der Anforderungen aufgrund der Integration in ein Dialogsystem führten zur Entwicklung einer neuen, performanzorientierten Architektur, indie sprachliche Verarbeitung in einer doppelten Verarbeitungskaskade abläuft, wobei parallel zur zweiten Kaskadenstufe die Generierung von Zeigegesten erfolgt.

Die Architektur des Systems POPEL erlaubt es, die Verarbeitung vollständig inkrementell durchzuführen. Inkrementell heißt, daß Teilergebnisse einer Komponente an die nachfolgenden Komponenten schon weitergegeben werden, bevor die Verarbeitung in der Ausgangskomponente vollständig abgeschlossen wurde.

Dadurch können zwischen den Komponenten Rückwirkungen erfolgen, sodaß eine Komponente Informationen, die für die Bearbeitung notwendig sind, aber aufgrund einer unterspezifizierten Eingabe fehlen, bei der Komponente anfordern kann, die diese Informationen liefert. Damit konnte die Trennung der beiden Hauptkomponenten zur Festlegung und Realisierung des Inhalts überwunden werden, ohne daß das aufgabenspezifische Wissen und die Methoden der Einzelkomponenten miteinander vermischt werden mußten.

19. *Entwicklungsumgebung zur Erstellung der Einträge:*

    Morpho-syntaktisches Lexikon: Wissensakquisitionsdialog bei unbekannten Woertern Semantisches Lexikon: Wissensakquisitionssystem FSS-WASTL

    Grammatik: PATR

20. *Anzahl und Art der Einträge:*

    Morpho-syntaktisches Lexikon: ca 7000

    Semantisches Lexikon: ca 100

    Grammatik: ID-Teil: ca 30, LP-Teil ca 20 (Aufgrund der verwendeten Theorie (HPSG-aehnlich) werden nur wenige Regeln benoetigt.

Literaturangaben:

N. Reithinger. Generating referring expressions and pointing gestures. In G. Kempen (Hrsg.), Natural Language Generation, Seiten 71–81. Nijhoff, Dordrecht, 1987.

N. Reithinger. Ein erster Blick auf POPEL: Wie wird was gesagt? In K. Morik (Hrsg.), 11. GWAI, Seiten 315–319, Berlin, 1987. Springer.

N. Reithinger. Dialogstrukturen und Dialogverarbeitung in XTRA. Memo Nr. 38, Lehrstuhl für Informatik IV, Universität des Saarlandes, Saarbrücken, 1989.

N. Reithinger. POPEL - an incremental and parallel natural language generation system. In C.L. Paris, W.R. Swartout und W.C. Mann (Hrsg.), Natural Language Generation in

Artificial Intelligence and Computational Linguistics, Kapitel 7, Seiten 179–200. Kluwer, Norwell, MA., 1991.

N. Reithinger. The Performance of an Incremental Generation Component for Multi-Modal Dialog Contributions. In Pdings of the 6th. International Workshop on Natural Language Generation, Springer, 1992.

N. Reithinger. Eine parallele Architektur zur inkrementellen Generierung multimodaler Dialogbeiträge. INFIX, St. Augustin, 1992. DISKI 1.

D. Schmauks und N. Reithinger. Generating multimodal output – conditions, advantages and problems. In 12. COLING, Seiten 584–588, Budapest, 1988.

J. Allgayer, R.M. Jansen-Winkeln, C. Reddig und N. Reithinger. Bidirectional use of knowledge in the multi-modal natural language access system XTRA. In 11. IJCAI, Seiten 1492–1497, Detroit, 1989.

J. Jung, A. Kresse, N. Reithinger und R. Schäfer. Das System ZORA – wissensbasierte Generierung von Zeigegesten. In D. Metzing (Hrsg.), 13. GWAI, Seiten 190–194, Berlin, 1989. Springer.

R.M. Jansen-Winkeln, A. Ndiaye und N. Reithinger. FSS-WASTL – interactive knowledge acquisition for a semantic lexicon. In Trends in Artificial Intelligence, 2nd AI*IA Congress, Springer, 1991.

W. Finkler. POPEL–HOW: Ein verteiltes, paralleles Modell zur inkrementellen Generierung natürlichsprachlicher Sätze aus konzeptuellen Einheiten, Teil 1. Diplomarbeit, Lehrstuhl für Informatik IV, Universität des Saarlandes, Saarbrücken, 1989.

G. Neumann and W. Finkler. A Head-Driven Approach to Incremental and Parallel Generation of Syntactic Structures. In: 13. COLING, pp. 288–293, Helsinki, 1990.

G. Neumann. POPEL–HOW: Parallele, inkrementelle Generierung natürlichsprachlicher Sätze aus konzeptuellen Einheiten, Teil 2. Diplomarbeit, Lehrstuhl für Informatik IV, Universität des Saarlandes, Saarbrücken, 1989.

6.5 VERBALIZE

1. *Name der Komponente:* VERBALIZE

2. *Zweck der Komponente:* Ueberfuehrung einer logischen Form in F-Strukturen; Besonderheit dabei ist die Moeglichkeit von erheblichen Umstrukturierungen durch inverse Lexeminterpretation

3. *Entwickelt und implementiert durch:* on: Helmut Horacek, Justus Meier, Martin Meister, Cornelia Peters

4. *Anschrift, Kontaktperson, e-mail:* In: Universitaet Bielefeld LILI-Fakultaet Postfach 8640 D-4800 Bielefeld 1 Tel.: (0521)-106-3678 (Horacek) E-mail: horacek@techfak.uni-bielefeld.de

5. *Implementierungssprache:* Commonlisp

6. *Zugrundeliegende Hardware / Betriebssystem(e):* SUN4, Unix

7. *Beginn / Ende der Entwicklung:* 1988-1992

8. *Projekt:* DIAMOD (und Vorarbeiten)

9. *Input / Eingabeschnittstellen:* Logische Form gemaess der im Projekt WISBER entwickelten Sprache IRS; 1- und 2- stellige Praedikate moeglich

10. *Output / Ausgabeschnittstellen:* F-Strukturen a la LFG

11. *Aktueller Zustand:* In Entwicklung

12. *Abhängigkeiten / Anforderungen an andere Systeme:* Zu Elementen der logischen Form passende Wissensrepraesentation; Konzept- und Rolleninformation, hierarchische Beziehungen, terminologisches und referentielles Benutzerwissen

17. *Benutzte Techniken und zugrundeliegende Theorien:* Unifikation, Depth-First Search, Pruning, Pattern Matching

18. *Allgemeine Beschreibung:* in den Artikeln

    a) H. Horacek, The Architecture of a Generation Component in a Complete Natural Language Dialog System, in Current Issues in Natural Language Generation, R. Dale, C. Mellish, M. Zock (eds.), S193-227, Academic Press, 1990.

    b) H. Horacek, Some Useful Search Tehniques for Natural Language Generation, in GWAI-90, H. Marburger (ed.), Geseke, Sept. 1990.

6.6 WIP-TAG-GEN

1. *Name der Komponente:* WIP-TAG-GEN
2. *Zweck der Komponente:* Syntaxkomponente zur inkrementellen Generierung natuerlicher Sprache
3. *Entwickelt und implementiert durch:* Anne Schauder, Wolfgang Finkler, Karin Harbusch und mehrere Wissenschaftliche Hilfskraefte, DFKI Saarbruecken
4. *Anschrift, Kontaktperson, e-mail:* Anne Schauder, DFKI, Stuhlsatzenhausweg 3, 6600 Saarbruecken, e-mail: schauder@dfki.uni-sb.de
5. *Implementierungssprache:* Symbolics Common LISP unter Release 8.01
6. *Zugrundeliegende Hardware / Betriebssystem(e):* MacIvory bzw. Symbolics Lispmaschine
7. *Beginn / Ende der Entwicklung:* Herbst 1989 – Maerz 1993
8. *Projekt:* WIP (Wissensbasierte Informationspraesentation)
9. *Input / Eingabeschnittstellen:* Die Eingabeschnittstelle verarbeitet Einheiten, die Inhaltswoerter enthalten mit Subkategorisierungsinformationen, syntaktischen/semantischen Angaben und funktionalen Beziehungen zwischen diesen Einheiten. Diese koennen ungeordnet und mit beliebigen Verzoegerungen (inkrementell) uebergeben werden. Eine Erweiterung des Systems um eine Wortwahlkomponente ist geplant, dabei wird sich die Eingabeschnittstelle entsprechend nach oben verschieben.
10. *Output / Ausgabeschnittstellen:* In eine globale Variable wird eine Liste von Strings inkrementell eingetragen, die von links nach rechts gelesen den produzierten Satz ergeben.
11. *Aktueller Zustand:* lauffaehig, noch in der Entwicklung
12. *Abhängigkeiten / Anforderungen an andere Systeme:* Wegen deklarativer Eingabespezifikation leicht als Ausgabe fuer andere Generierungssysteme geeignet; z.B. derzeit in ein multimodales Praesentationssystem integriert.
13. *Dokumentation:* Schauder 90, Schauder 92a, Schauder 92b, Harbusch et al. 91, Dokumentation des LISP-Codes in den Dateien
14. *Verfügbarkeit:* -
15. *Wartbarkeit:* -
16. *Fremdinstallationen:* keine

17. *Benutzte Techniken und zugrundeliegende Theorien:* - syntaktischer Repraesentationsformalismus: Lexikalisierte Local Dominance/Linear Precedence-TAGs mit Unifikation (siehe Harbusch et al. 91) - Beschreibungsgesteuerte Generierung: Die vorliegenden Eingaben und bereits repraesentierten syntaktischen Strukturen werden direkt als Instruktionen fuer den Generierer interpretiert. Die Eingabe bilden Lexeme, daraus ergibt sich eine lexikalische Steuerung. - Two-level Generierung: Zwischen Ein- und Ausgabe liegen zwei Ebenen syntaktischer Repraesentation, die hierarchische und die positionale Ebene (LD/LP-Trennung). - Parallele Bearbeitung verschiedener Teile des syntaktischen Baums - Inkrementelle Generierung, insbesondere Produktion inkrementeller Ausgabe, wobei Selbstkorrekturen noetig werden (vgl. Finkler und Schauder 92) - Implementierung: object-oriented concurrent programming

18. *Allgemeine Beschreibung:* Der hier vorgestellte syntaktische Generierer ist im Rahmen des Projekts WIP (Wissensbasierte Informationspraesentation, siehe [Wahlster et al. 88], [Wahlster et al. 91]) am DFKI Saarbruecken entstanden. Ziel des Projekts ist es, ausgehend von einer gemeinsamen Wissensquelle automatisch und dynamisch multimodale Praesentationen zu erzeugen. Die Modi sind z.Zt. Text und Graphik. Ausgehend von einer Eingabe, die das Praesentationsziel beschreibt, entwirft der Praesentationsplaner die zu produzierende Ausgabe und entscheidet dabei ueber Inhalt und Modus der zu erzeugenden Teile. Sie werden an die beiden entsprechenden Generierer uebergeben, den Text- und den Graphikgenerierer. Beide formen zusammen mit dem Planer je eine Kaskade aus einer Design- und einer Realisierungskomponente. Der dynamische Austausch von Information zwischen den beiden Generierern ermoeglicht z.B. die Erzeugung crossmodaler Referenzen.

    Der synaktische Generierer WIP-TAG-GEN bildet die unterste Komponente der Textgenerierungskaskade, die sogenannte Text- Realisierungs-Komponente. In der darueberliegenden Text-Design- Komponente - die z.Zt. in der Entwicklung ist - wird ueber die Satzform entschieden und die Wortwahl vollzogen. Die Eingabe in die Text- Realisierungskomponente besteht aus den gewaehlten Lexemen und spezifizierten syntaktischen Beziehungen zwischen ihnen in deklarativer Form.

    Dem Systems WIP-TAG-GEN liegt das Prinzip der Inkrementalitaet zugrunde, von deren Umsetzung wir uns entscheidende Effizienzvorteile versprachen. Es hat sich auch gezeigt, dass die Flexibilitaet des Systems erhoeht wird. Die inkrementelle Verarbeitung wird zusaetzlich durch die Realisierung des Generierers als verteiltes paralleles System unterstuetzt, indem die Verarbeitung der einzelnen Teile beschleunigt wird.

    Als zugrundeliegenden Repraesentationsformalismus fuer den syntaktischen Generierer waehlten wir die Tree Adjoining Grammars. Die Regeln dieser Grammatik werden durch Baeume repraesentiert, deren - gegenueber kontextfreien Grammatiken - erweiterte Domaene der Lokalitaet eine kompakte Darstellung syntaktischer Strukturen ermoeglicht. Durch die Kombination mit dem Unifikationsformalismus (UTAG, siehe Schauder 92b) koennen komplexe syntaktische Merkmale kodiert und ihre Vererbung durch die Baeume ermoeglicht werden. Die Aufteilung des Generierers in eine hierarchische und eine positionale Ebene (s.u.) wird durch die Verwendung

von LD/LP (Local Dominance/Linear Precedence) TAGs erleichtert. Bei dieser Erweiterung von TAGs werden die Regeln (die Baeume) aufgespalten in ein Mobile, das ausschliesslich die hierarchische Struktur der entsprechenden Konstruktion beschreibt, und eine Menge von LP-Regeln zu jedem Mobile, die die Reihenfolgen der Geschwisterknoten einschraenken. Um eine inkrementelle und parallele Verarbeitung im syntaktischen Generierer zu ermoeglichen, muessen die Strukturen, die auf die Objekte des parallelen Systems verteilt werden, moeglichst unabhaengig sein. Die Baeume Lexikalisierter TAGs beschreiben syntaktische Constraints jeweils eines lexikalen Items und repraesentieren etwaige Ergaenzungen durch Platzhalter im Baum (sogenannte Substitutionsknoten). Jedes Objekt des verteilten parallelen Systems ist verantwortlich dafuer, eines der eingegebenen Lexeme in die syntaktische Gesamtstruktur zu integrieren. Als ersten Schritt waehlt es einen entsprechenden Baum der lexikalisierten TAG, im weiteren Verlauf der Generierung versuchen die Objekte, ihre lokal verwalteten Baeume durch Nachrichtenaustausch mit den Baeumen der anderen Objekte zu verknuepfen. Dabei werden die zwei Verknuepfungsoperationen von TAGs - Adjunktion und Substitution - entsprechend im verteilten parallelen System umgesetzt. Die Objekte bilden eine Objekthierarchie bez. der Dependenzrelationen zwischen den von ihnen verwalteten Lexemen.

Die Architektur des syntaktischen Generierers umfasst in der Hauptsache drei Komponenten. Im Interface wird die oben beschriebene Baumauswahl realisiert. Dabei spielen nicht nur die syntaktischen Constraints des ausgewaehlten Lexems eine Rolle, sondern auch die bisherige Entwicklung der Praesentation sowie die Werte verschiedener Parameter - wie z.B. Stil - die im Gesamtsystem gesetzt werden koennen. Im Phrase Formulator versuchen die Objekte, ihre lokale syntaktische Struktur per Nachrichtenaustausch mit den Strukturen der anderen Objekte zu einem syntaktischen Satzbaum zu kombinieren, der allerdings immer noch ueber die Objekte verteilt bleibt. Die beiden Verknuepfungsoperationen Adjunktion und Substitution werden 'virtuell' ausgefuehrt, damit das lokale Wissen der Objekte nicht vermischt wird und die Parallelitaet aufrechterhalten werden kann. Statt die Baeume strukturell zu verbinden, werden zwischen den Objekten Kopien von Merkmalsstrukturen hin- und hergeschickt, die das Wissen der entsprechenden Teilbaeume repraesentieren. Hat ein Objekt seinen Baum in die globale Struktur eingebunden und hat es von den Nachbarobjekten genuegend Information zur Flexion der Terminale geerbt, so wechselt es in die Linearization Component. Dort durchlaeuft es seinen Teilbaum gemaess den assoziierten LP-Regeln. Die Frage nach der Ausgabeerlaubnis beantwortet der jeweilige Regent in der Objekthierarchie. Substitutionsknoten unterbrechen die Ausgabe, da sie Platzhalter fuer die Strukturen anderer Objekte sind. In diesem Fall kann das Objekt die Ausgabeaktivitaet an den jeweiligen Dependenten uebergeben. Vor der Ausgabe eines Wortes wird dieses mithilfe des Flexionsmoduls MORPHIX flektiert.

Bei der Erzeugung inkrementeller Ausgabe muessen auch Selbstkorrekturen moeglich sein, z.B. wenn ein optionales Element des aktuellen Satzes zu einem Zeitpunkt eingegeben wird, zu dem es nicht mehr syntaktisch korrekt an den bisherigen Ausgabesatz angehaengt werden kann. Zur Zeit werden einige dieser Repair-Faelle verteilt

im Objektnetz geloest; eine zentrale Monitorkomponente ist in Entwicklung, die flexibler auf Problemfaelle reagieren kann. Diese Reaktion muss sich nicht nur auf Umorganisationen auf der syntaktischen Ebene beziehen, sie kann auch eine Neuplanung auf einer der hoeheren Ebenen ausloesen.

Eine Hilfe bei der Entdeckung syntaktischer Mehrdeutigkeiten und auch bei der Durchfuehrung von Selbstkorrekturen ist die Anticipation Feedback Loop. Ein Earley-basierter, direkter TAG-Parser, der auf der gleichen Grammatik wie der Generierer arbeitet, analysiert inkrementell die entstehenden Satzteile und liefert das Ergebnis an die Monitorkomponente. Die Anticipation Feedback Loop ist konzeptuell entwickelt und wird in Kuerze in das System integriert.

19. *Entwicklungsumgebung zur Erstellung der Einträge:* Die Baeume der Grammatik werden in einer Datei in einer hierarchischen Listennotation definiert. Diese Datei wird in Baumstrukturen uebersetzt, die in einer eigens dafuer entwickelten graphischen Oberflaeche untersucht und testweise verknuepft werden koennen. Die Lexikoneintraege werden ebenfalls entsprechend einer definierten Syntax in einer Datei beschrieben und in interne Strukturen uebersetzt. Diese Strukturen und insbesondere die assoziierten TAG-Baeume koennen in einer weiteren graphischen Oberflaeche untersucht und die lexikalische Insertion eines Blattes in einem Baum simuliert werden.

20. *Anzahl und Art der Einträge:* Eine deutsche Grammatik und eine deutsches Lexikon befinden sich im Aufbau. Augenblicklich umfasst die Grammatik etwa 150 Baeume (Satzbaeume, Baeume fuer Ergaenzungen, Adjunkte, Nebensaetze, Infinitivphrasen, ...).

Literaturangaben:

[Finkler und Neumann 88] W. Finkler and G. Neumann. MORPHIX: A Fast Realization of a Classification-Based Approach to Morphology. In: Proceedings of the Workshop 'Wissensbasierte Sprachverarbeitung', Berlin, Germany, 1988, Springer-Verlag.

[Finkler und Schauder 92] W. Finkler and A. Schauder. Effects of Incremental Output on Incremental Natural Language Generation. Proceedings of the 10th European Conference on Artificial Intelligence (ECAI 92), 1992. to appear.

[Harbusch et al. 91] K. Harbusch, W. Finkler, A. Schauder. Incremental Syntax Generation with Tree Adjoinng Grammars. Proceedings 4. Int. GI- Kongress Wissensbasierte Systeme, 23./24. Okt. 1991, Muenchen, pp. 363- 374, Springer-Verlag, 1991.

[Schauder 90] A. Schauder. Inkrementelle syntaktische Generierung natuerlicher Sprache mit Tree Adjoining Grammars, Diplomarbeit, Fachbereich Informatik, Universitaet des Saarlandes, Saarbruecken, 1990

[Schauder 92a] A. Schauder. Incremental Syntactic Generation of Natural Language with

Tree Adjoining Grammars. Technical memo, German Research Center for Artificial Intelligence (DFKI), 1992. to appear.

[Schauder 92b] A. Schauder. Realization of Tree Adjoining Grammar with Unification, Technical memo, German Research Center for Artificial Intelligence (DFKI), 1992. to appear.

[Wahlster et al. 88] W. Wahlster, E. Andre, M. Hecking, T. Rist: WIP: Wissensbasierte Informationspraesentation, Projektbeschreibung, DFKI Saarbruecken, 1988.

[Wahlster et al. 91] W. Wahlster, E. Andre, W. Graf, T. Rist: Designing Illustrated Texts: How Language Production is influenced by Graphics Generation. Proceedings of the 5th Conference of the European Chapter of the ACL (EACL 91), pp. 8-14, Berlin, 1991.

6.7 NUGGET

1. *Name der Komponente:* NUGGET (eingetr.Warenzeichen der SNI AG)

2. *Zweck der Komponente:* Generierung natürlichsprachlicher, dynamischer Erklärungen und Fakten, sowie Regeln im Rahmen von Expertensystemanwendungen der Shell TWAICE (R)

4. *Anschrift, Kontaktperson, e-mail:* Konrad Jablonski, SNI AG - AP 333, Anwendersoftware und Projekte, KI-Zentrum Paderborn, Riemekestr. 160, 4790 Paderborn jablonski.pad@sni.de

5. *Implementierungssprache:* Prolog, Zusatzmodul Verbmorphologie in C

6. *Zugrundeliegende Hardware / Betriebssystem(e):* UNIX, DOS (andere Portierungen im Rahmen von TWAICE zu erfragen) SNI-SINIX-Maschinen (TARGON, MX, RM), DOS-PC, UNIX-PC

7. *Beginn / Ende der Entwicklung:* bis 31.12.86: Prototypentwicklung ab 1.1.87: Produktentwicklung bis Anf. 1990 / Release 4.0 Wartung gewdhrleistet

8. *Projekt:* BMFT-Projekt WISBER ESPRIT-Projekt 311/ ADKMS

9. *Input / Eingabeschnittstellen:* semantisch-pragmatische Repraesentation in der propositionalen Sprache ARPS (proprietaer); PROLOG-Notation Blackboard-orientierte Verknuepfung mit Applikationen (TWAICE-unabhaengig)

10. *Output / Ausgabeschnittstellen:* formatierter ASCII-Text

11. *Aktueller Zustand:* freigegebenes, modulares Teilsystem des offiziellen TWAICE-Produktumfangs

12. *Abhängigkeiten / Anforderungen an andere Systeme:* Einhaltung der Schnittstellen-Sprache

13. *Dokumentation:* zahlreiche Fachliteratur/ TBL-Buch s. Literaturliste

14. *Verfügbarkeit:* zusammen mit TWAICE, Gesamt-Preis HW-abhaengig auf Anfrage Copyright SNI, Warenzeichenschutz

    Einzellizenz auf Anfrage

15. *Wartbarkeit:* durch SNI gegen Kostenerstattung Kauf des Quellcodes zu einzeln zu vereinbarenden Konditionen

16. *Fremdinstallationen:* im Rahmen von TWAICE bei zahlreichen Kunden Installation bei der Forschung/Entwicklung von TA Triumph Adler AG

17. *Benutzte Techniken und zugrundeliegende Theorien:* Definite Clause Grammar, semantisch gesteuerte Restrictionen, sprechakt-basiert

18. *Allgemeine Beschreibung:* s. Literatur

19. *Entwicklungsumgebung zur Erstellung der Einträge:* modulares Teilsystem ALEXIS mit Desktop-Oberflaeche Erfragung morphologischer Merkmale Bearbeitung auch durch linguistische Laienmoeglich

20. *Anzahl und Art der Einträge:* nur morphologische Mindestangaben zur korrekten Flexion Zuordnung von Items zu Wissensbank zu Lexemen

Literaturangaben:

(sortiert in chronologischer Reihenfolge)

K.Jablonski, A.Rau, H.Rvsner: COLING 86 - Bonn (Tagungsbericht); Rundbrief d. Fachausschusses KI der GI; Heft 45 / 1987; Oldenbourg München

K.Jablonski, A.Rau, J.Ritzke: "Konzeption und Architektur des taktischen Textgenerierungssystems NUGGET"WISBER-Memo 12; Saarbrücken Mai 1987

K.Jablonski, A.Rau, J.Ritzke: Syntax- und Morphologieverarbeitung im Textgenerierungssystem NUGGET. in: Abstracts der Jahrestagung der Deutschen Gesellschaft für Sprachwissenschaft, Wuppertal Mdrz 1988; auch in: WISBER-Arbeitsunterlage U 21

K.Jablonski: Bericht über den 11th German Workshop on Artificial Intelligence (Sprachorientierte KI); in: LDV-Forum (5) 1988, Nummer 4, S. 48 - 52

K.Jablonski, A.Rau, J.Ritzke: Anwendung einer logischen Grammatik zur Generierung deutscher Texte. in: H.Trost (Hrsg.): 4.Vsterreichische Artificial-Intelligence-Tagung, Wien, August 88, Proceedings; Berlin etc.: Springer 1988; S. 83-93; auch als WISBER-Bericht 25, Juni 1988

K.Jablonski, A.Rau, J.Ritzke: NUGGET - Ein DCG-basiertes Textgenerierungssystem. WISBER-BERICHT 27, August 1988

K.Jablonski, A.Rau, J.Ritzke: Benutzerfreundlichkeit durch natürliche Sprache: NUGGET - ein Textgenerierungssystem. in: S.Savory (Hrsg.): Expertensysteme: Nutzen für Ihr Unternehmen; München-Wien: Oldenbourg, 2. überarbeitete u. erweiterte Aufl. 1989, S. 365 - 392

K.Lehner: Eine Lehrkomponente für Expertensysteme: eine Implementierung mit TWAICE. in: unix/mail (7) 1989; Heft 2; S. 52 - 56

H.U.Block, B.Frederking, M.Gehrke, H.Haugeneder, R.Hunze, K.Jablonski, A.Rau, J.Ritzke, S.Schachtl: Sprachanalyse und Textgenerierung im natürlich-sprachlichen Beratungssystem WISBER; in: W.Brauer/C.Freksa: Wissensbasierte Systeme, 3. Internationaler GI-Kongreß München 1989, Berlin, Heidelberg etc.: Springer 1989; S. 275 - 285; erscheint demndchst als Langfassung in: Informatik - Forschung und Entwicklung, Springer

U.Bauernfeind: Natürlich-sprachliche Textgenerierung macht Expertensystem- Schlußfolgerungen durchschaubarer. in: unix/mail (7) 1989; Heft 4; S. 38-42

K.Jablonski, A.Rau, J.Ritzke: Wissensbasierte Textgenerierung - Linguistische Grundlagen und softwaretechnische Realisierung; Tübingen: Narr 1990

K.Lehner: Wissensbasierte Lehrsysteme; München: Oldenbourg 1990

H.-H.Pardey: Die "künstliche Intelligenzßpricht ganz normales Deutsch. in: Frankfurter Allgemeine Zeitung; Technik und Motor; 10.4.1990; S. T6

K.Jablonski, J.Samuel: Natural Language Information Access System; in: ESPRIT - Information Processing Systems; Brüssel 1990

6.8 Antwortgenerierungskomponente von NAUDA

1. *Name der Komponente:* Antwortgenerierungskomponente von NAUDA
2. *Zweck der Komponente:* Erweiterung einer NL-DB-Schnittstelle (LanguageAccess) um natuerlichsprachl. Antworten bei: - geeignetem DB-Output (Tabellen mit 1 numerischen Wert) - Zurueckweisung/Korrektur von Praesuppositionsverletzungen - Ueberbeantwortung bei Fragen nach numerischen Werten
3. *Entwickelt und implementiert durch:* Maria Strobel
4. *Anschrift, Kontaktperson, e-mail:* Maria Strobel FAW, Postf. 2060, 7900 Ulm STROBEL at DHDIBM1 STROBEL at DULFAW1A

   bzw. Dr. Dietmar Roesner (Tel: o731 / 501 - 530) ROESNER at DULFAW1A
5. *Implementierungssprache:* Prolog(VM-Prolog)
6. *Zugrundeliegende Hardware / Betriebssystem(e):* IBM Mainframe / VM
7. *Beginn / Ende der Entwicklung:* 04-91 / 04-92
8. *Projekt:* AUDA (Natuerlichsprachlicher Zugang zu Umweltdatenbanken)
9. *Input / Eingabeschnittstellen:* logische Form von LanguageAccess, bzw. Output der Datenbank
10. *Output / Ausgabeschnittstellen:* NL (deutsch)
11. *Aktueller Zustand:* fertiger Prototyp
17. *Benutzte Techniken und zugrundeliegende Theorien:* Erweiterung der Paraphrasengenerierung von LanguageAccess
18. *Allgemeine Beschreibung:* s.2

7 Architektur

7.1 SUNSTAR - Systemarchitektur fuer Sprachverarbeitende Systeme

1. *Name der Komponente:* SUNSTAR - Systemarchitektur fuer Sprachverarbeitende Systeme

2. *Zweck der Komponente:* Eine Systemarchitektur zur einfachen und schnellen Generierung sprachgesteuerter Anwendungen

3. *Entwickelt und implementiert durch:* ESPRIT Projekt SUNSTAR

4. *Anschrift, Kontaktperson, e-mail:* Fraunhofer-Institut fuer Arbeitswirtschaft und Organisation Nobelstrasse 12 7000 Stuttgart 80 Dipl.-Ing. T. Renner, Dipl.-Ing. J. Brettschneider renner@iao.fhg.de, brettsch@iao.fhg.de

5. *Implementierungssprache:* C, C++

6. *Zugrundeliegende Hardware / Betriebssystem(e):* SUN SPARC / SUN4-OS4 PC / VENIX386

7. *Beginn / Ende der Entwicklung:* 12/89 - 06/92

8. *Projekt:* ESPRIT 2094 - SUNSTAR

9. *Input / Eingabeschnittstellen:* Offenes System, Anbindung von Moduln (Spracheingabe, Sprachausgabe, Applikation, etc.) ueber Treibersoftware

10. *Output / Ausgabeschnittstellen:* s. 9.

11. *Aktueller Zustand:* Systemfunktionalitaet anhand 5 verschiedener Anwendungen im Telekommunikations- und im Buerobereich demonstriert. Bewertung der Anwendungen, Systemverbesserungen in der Durchfuehrung.

12. *Abhängigkeiten / Anforderungen an andere Systeme:* Keine

13. *Dokumentation:* zu beziehen bei o.a. Kontaktpersonen

15. *Wartbarkeit:* gewaehrleistet

16. *Fremdinstallationen:* Installationen bei vier verschiedenen Projekt- partnern

18. *Allgemeine Beschreibung:* Das SUNSTAR Projekt wird von acht europaeischen Firmen, Forschungs- instituten und Universitaeten getragen. Im Projekt wurde ein sehr flexibles sprachverarbeitendes System mit integrierter Dialogkontrolle entwickelt. Das System basiert auf einer generalisierten Architektur, die aus den Komponenten Dialogbeschreibung, Dialogmanager, Kommunika- tionssteuerung und Geraetetreibern besteht. (Sprach-)Ein-/Ausgabegeraete und sprachgesteuerte Applikation

werden ueber Treibersoftware an das System angebunden. Dadurch koennen technologische Entwicklungen im Bereich Spracherkennung und Sprachsynthese beruecksichtigt und einfach in das bestehende System integriert werden. Im Folgenden sind die einzelnen Komponenten des Systems kurz beschrieben: (Komponente 3R Software und COSIMA: siehe unter Spracherkennung, DDL Tool: siehe unter Tools, DECtalk: siehe unter Sprachsynthese, SCM - SUNSTAR Communication Manager und ICM - Interpretation and Control Module folgen.)

7.2 SCM - SUNSTAR Communication Manager

1. *Name der Komponente:* SCM - SUNSTAR Communication Manager

2. *Zweck der Komponente:* Zentraler Bestandteil der offenen Systemarchitektur. Verbindet die Software Module des Systems, die auf unterschiedlichen Rechnern (Workstations, PCs) unter unterschiedlichen UNIX-Implementationen laufen koennen. Aufgaben: Management der Kommunikation zwischen Prozessen und Rechnern, Handhabung der Systemkonfiguration (Belegung von Geraeten und Applikationen, Initialisierung und Kontrolle der Systemkomponenten, Systemueberwachung und Fehlerbehandlung.

3. *Entwickelt und implementiert durch:* Projekt SUNSTAR

4. *Anschrift, Kontaktperson, e-mail:* Fraunhofer-Institut fuer Arbeitswirtschaft und Organisation Nobelstrasse 12 7000 Stuttgart 80 Dipl.-Ing. T. Renner, Dipl.-Ing. J. Brettschneider renner@iao.fhg.de, brettsch@iao.fhg.de

5. *Implementierungssprache:* C

6. *Zugrundeliegende Hardware / Betriebssystem(e):* Workstations, PC's unter UNIX, Kommunikation zwischen Rechnern ueber TCP/IP.

7.3 ICM - Interpretation and Control Module

1. *Name der Komponente:* ICM - Interpretation and Control Module

2. *Zweck der Komponente:* Der ICM handhabt den Dialog des Systems mit dem Benutzer gemaess einer zugrundeliegenden Dialogbeschreibung. Der ICM ist dabei Applikationsunabhaengig. Mehrere ICMs koennen im System gleichzeitig aktiv, d.h. mehrere (unterschiedliche) Applikationen und Dialoge koennen vom System gleichzeitig gehand- habt werden. Aufgaben des ICM: Parsing der Dialogbeschreibung, Interpretation und Kontrolle des Dialogs zwischen Benutzer und Applikation, Kontrolle von Ein-/Ausgabegeraeten und Applikationen, Fehlerbehandlung.

3. *Entwickelt und implementiert durch:* SUNSTAR Projekt

4. *Anschrift, Kontaktperson, e-mail:* Fraunhofer-Institut fuer Arbeitswirtschaft und Organisation Nobelstrasse 12 7000 Stuttgart 80 Dipl.-Ing. T. Renner, Dipl.-Ing. J. Brettschneider renner@iao.fhg.de, brettsch@iao.fhg.de

5. *Implementierungssprache:* C

6. *Zugrundeliegende Hardware / Betriebssystem(e):* Workstations, PCs unter UNIX

7.4 ASL-Nord-Architektur

1. *Name der Komponente:* ASL-Nord-Architektur

2. *Zweck der Komponente:* Kommunikationsorientierte Architektur fuer das ASL-Nord-System. Die Komponente bildet einen Rahmen fuer die Verarbeitungs- komponenten eines Speech-Language-Systems. Dieser Rahmen stellt den Verarbeitungskomponenten ein Kommunikationssystem zu Ver- fuegung, mit dessen Hilfe der Austausch von partiellen Analyse- resultaten ermoeglicht wird. Zusaetzlich koennen die Komponenten einen interaktiven Dialog fuehren mit dem Ziel der Aufloesung von lokalen Ambiguitaeten durch Ermittlung nicht-lokaler Information. Die Architektur orientiert sich an verteilten Systemen und an Ergebnissen aus der Verteilten KI (DAI).

3. *Entwickelt und implementiert durch:* Claudius Pyka im Rahmen des Projekts ASL-Nord.

4. *Anschrift, Kontaktperson, e-mail:* Universitaet Hamburg Fachbereich Informatik Arbeitsbereich Natuerlichsprachliche Systeme Bodenstedtstr. 16 2000 Hamburg 50 C. Pyka: claude@nats1.informatik.uni-hamburg.de

5. *Implementierungssprache:* C

   Anmerkung: Die Verarbeitungskomponenten, die von der Architektur integriert werden, sind in den Sprachen Lisp, Prolog und C++ implementiert. Entsprechende Schnittstellen zwischen den Programmiersprachen stellt ebenfalls die Architekturkomponente zur Verfuegung.

6. *Zugrundeliegende Hardware / Betriebssystem(e):* Sun SPARC / SunOS4.1.1

7. *Beginn / Ende der Entwicklung:* Beginn: 01.91 Ende: voraussichtlich 12.94

8. *Projekt:* ASL-Nord

11. *Aktueller Zustand:* in Entwicklung

12. *Abhängigkeiten / Anforderungen an andere Systeme:* siehe Punkte 1. und 5.

14. *Verfügbarkeit:* (noch) nicht verfuegbar

17. *Benutzte Techniken und zugrundeliegende Theorien:* Verteilte KI, Verteilte Systeme; Psycholinguistik und kognitive Psychologie; Modelle der menschlichen Sprach- verarbeitung

18. *Allgemeine Beschreibung:* siehe Literatur

Literaturangaben:

1. C. Pyka: Architektur von ASL-Nord. (ASL-Nord Technischer Bericht ASL-TR-4-91/UHH, Juni 1991)

2. C. Pyka: Deterministische, inkrementelle und zeitsynchrone Verarbeitung und die Architektur von ASL-Nord. (ASL-Nord Technischer Bericht ASL-TR-22-91/UHH, Dezember 1991)

3. C. Pyka: Management of Hypotheses in an Integrated Speech-Language Architecture. (erscheint in den Proceedings der ECAI 92)

8 Tools

8.1 TDL2LATEX

1. *Name der Komponente:* TDL2LATEX
2. *Zweck der Komponente:* Uebersetzung von interner Darstellung von TDL in LATEX Code
3. *Entwickelt und implementiert durch:* Stephan Diehl
4. *Anschrift, Kontaktperson, e-mail:* diehl@dfki.uni-sb.de
5. *Implementierungssprache:* CommmonLisp
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Unix
7. *Beginn / Ende der Entwicklung:* ab Januar 1992
8. *Projekt:* DISCO

11. *Aktueller Zustand:* noch in der Testphase
12. *Abhängigkeiten / Anforderungen an andere Systeme:* nur zusammen mit dem Unifikator oder TDL verwendbar
13. *Dokumentation:* Es existiert ein User Manual (8 Seiten) als DVI-Datei.
14. *Verfügbarkeit:* Freie Software, Copyright Hinweis soll beibehalten werden zu Ruhm und Ehren des Autors
15. *Wartbarkeit:* Kenntnisse in LATEX und vom Aufbau der internen Struktur der Merkmalsstrukturen vorausgesetzt ist das Programm leicht zu warten.

8.2 DDL Tool - Dialogue Description Language and Dialogue Design Tool

1. *Name der Komponente:* DDL Tool - Dialogue Description Language and Dialogue Design Tool

2. *Zweck der Komponente:* Der Benutzerdialog wird auf einer Workstation mit Hilfe des Dialogue Design Tools entworfen. Das DDL Tool ist ein maechtiges grafisches Werkzeug, das drei Schichten umfasst. Es er- moeglicht die Spezifizierung, Implementierung und Verifikation des Dialogs, sowie einige Systemtests. Der erste Schritt des Dialogentwurfs besteht in der grafischen Festlegung der Dialogelemente und des Dialogverlaufs. Die grafischen Elemente basieren auf SDL Symbolen, die durch CCITT genormt sind. Die Symbolmenge wurde entsprechend der speziellen Anforderungen von Sprachdialogen erweitert. In einem zweiten und dritten Schritt werden die einzelnen Symbole in Lexika spezifiziert. Der Dialog wird dann in einer speziellen Dialogbeschreibungssprache compiliert. Der resultierende Code wird vom ICM zur Dialogsteuerung verwendet. Das Dialog Tool beinhaltet eine Syntaxueberpruefung auf allen Ebenen. Es kann auch zur Verifikation und zum Testen des Systems verwendet werden. Bei laufendem System kann der Dialogverlauf in der grafischen Schicht des Dialog Tools dargestellt werden.

3. *Entwickelt und implementiert durch:* Projekt SUNSTAR

4. *Anschrift, Kontaktperson, e-mail:* Fraunhofer-Institut fuer Arbeitswirtschaft und Organisation Nobelstrasse 12 7000 Stuttgart 80 Dipl.-Ing. T. Renner, Dipl.-Ing. J. Brettschneider renner@iao.fhg.de, brettsch@iao.fhg.de

5. *Implementierungssprache:* C++

6. *Zugrundeliegende Hardware / Betriebssystem(e):* Workstations, UNIX

8.3 CHART-DISPLAY

1. *Name der Komponente:* CHART-DISPLAY
2. *Zweck der Komponente:* Darstellung der Chart waehren des Parsing
3. *Entwickelt und implementiert durch:* Stephan Diehl, Bernd Kiefer
4. *Anschrift, Kontaktperson, e-mail:* diehl@dfki.uni-sb.de
5. *Implementierungssprache:* CommonLisp
6. *Zugrundeliegende Hardware / Betriebssystem(e):* UNIX
7. *Beginn / Ende der Entwicklung:* ab Juni 1991
8. *Projekt:* DISCO
9. *Input / Eingabeschnittstellen:* Der Benutzer definiert Schnittstellenfunktionen auf die Items in der Chart
10. *Output / Ausgabeschnittstellen:* Grafische Darstellung der Chart in einem Fenster
11. *Aktueller Zustand:* noch in der Entwicklung
12. *Abhängigkeiten / Anforderungen an andere Systeme:* derzeit wird das XTOOLKIT von Andreas Becker (DFKI) verwendet neuere Versionen werden CLIM als X-Grafikinterface benutzen.
13. *Dokumentation:* keine
14. *Verfügbarkeit:* als Testversion
18. *Allgemeine Beschreibung:* Beim Parsen werden rechte Seiten einer Regel durch die linke ersetzt. Den aktuellen Menge solcher Reduzierungen lassen sich grafisch darstellen.

8.4 Feature Editor fuer Macintosh

1. *Name der Komponente:* Feature Editor fuer Macintosh
2. *Zweck der Komponente:* Vollstaendig interaktive (GUI) Anzeigen und Veraendern von Featurestrukturen von (fast) beliebigen Merkmalsformalismen
3. *Entwickelt und implementiert durch:* Bernd Kiefer
4. *Anschrift, Kontaktperson, e-mail:* Bernd Kiefer, Deutsches Forschungszentrum fuer Kuenstliche Intelligenz, Stuhlsatzenhausweg 3, 6600 Saarbruecken, email: kiefer@ dfki.uni-sb.de
5. *Implementierungssprache:* C
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Getestet auf Mac II mit System 6.0.4 & 6.0.7
7. *Beginn / Ende der Entwicklung:* 1. Jan. 1989 - 1. Nov. 1991
8. *Projekt:* DISCO
9. *Input / Eingabeschnittstellen:* Textfiles in denen die Merkmalstrukturen in einer speziellen Syntax dargestellt sind.
10. *Output / Ausgabeschnittstellen:* siehe Input.
11. *Aktueller Zustand:* Programm fertiggestellt, Dokumentation noch in Entwicklung
12. *Abhängigkeiten / Anforderungen an andere Systeme:* keine
13. *Dokumentation:* wird noch erstellt
14. *Verfügbarkeit:* Weitergabe zu wiss. Zwecken
15. *Wartbarkeit:* Kann als Applikation nur vom Entwickler gewartet werden.
16. *Fremdinstallationen:* Bob Kasper, Ohio State University Kimmo Koskenniemi, University of Helsinki
18. *Allgemeine Beschreibung:* Interaktive Betrachtung und Aenderung von Merkmalstrukturen mit graphischer Oberflaeche wird durch dieses Tool ermoeglicht. Als eigene Applikation mit einem einfachen Interface ueber Textfiles kann es von allen anderen Programmiersystemen aus benutzt werden. Groesstmoegliche Flexibilitaet war ein wichtiges Kriterium bei der Entwicklung dieses Tools. Die Dokumentation sollte Ende April '92 verfueg- bar sein

8.5 Feature Editor fuer X-Windows

1. *Name der Komponente:* Feature Editor fuer X-Windows

2. *Zweck der Komponente:* Vollstaendig interaktive (GUI) Anzeigen und Veraendern (noch in Entwicklung) von Featurestrukturen von (fast) beliebigen Merkmalsformalismen

3. *Entwickelt und implementiert durch:* Bernd Kiefer, Thomas Fettig

4. *Anschrift, Kontaktperson, e-mail:* Bernd Kiefer, Deutsches Forschungszentrum fuer Kuenstliche Intelligenz, Stuhlsatzenhausweg 3, 6600 Saarbruecken, email: kiefer@ dfki.uni-sb.de

5. *Implementierungssprache:* C

6. *Zugrundeliegende Hardware / Betriebssystem(e):* Unix und X11R4 oder X11R5

7. *Beginn / Ende der Entwicklung:* 1. Apr. 1990 - 1. Jul. 1992 (vorraussichtlich)

8. *Projekt:* DISCO

9. *Input / Eingabeschnittstellen:* Textfiles in denen die Merkmalstrukturen in einer speziellen Syntax dargestellt sind. (koennen auch ueber pipe eingelesen werden)

10. *Output / Ausgabeschnittstellen:* siehe Input.

11. *Aktueller Zustand:* Programm noch in Entwicklung

12. *Abhängigkeiten / Anforderungen an andere Systeme:* keine

13. *Dokumentation:* wird noch erstellt

14. *Verfügbarkeit:* Weitergabe zu wiss. Zwecken

15. *Wartbarkeit:* Kann als Applikation nur vom Entwickler gewartet werden.

18. *Allgemeine Beschreibung:* Interaktive Betrachtung und Aenderung von Merkmalstrukturen mit graphischer Oberflaeche wird durch dieses Tool ermoeglicht. Als eigene Applikation mit einem einfachen Interface ueber Textfiles kann es von allen anderen Programmiersystemen aus benutzt werden. Groesstmoegliche Flexibilitaet war ein wichtiges Kriterium bei der Entwicklung dieses Tools. Die Dokumentation sollte Ende April '92 verfueg- bar sein

8.6 Interaktiver Grapher

1. *Name der Komponente:* Interaktiver Grapher
2. *Zweck der Komponente:* Visualisierungs- und Inspektionstool für Typhierarchien (a la HPSG), die mit TDL ExtraLight erstellt wurden
3. *Entwickelt und implementiert durch:* Ursprünglich in Public Domain; Anpassung, Erweiterungen und Verbesserungen bei DISCO durch Abdel Kader Diagne
4. *Anschrift, Kontaktperson, e-mail:* DFKI/Saarbrücken, Hans-Ulrich Krieger, krieger@dfki.uni-sb.de
5. *Implementierungssprache:* Allegro Commen Lisp auf Macintosh
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Apple Macintosh / System 6.0.5 und 6.0.7
7. *Beginn / Ende der Entwicklung:* Entwicklung abgeschlossen. Offen für Erweiterungen. Zur Zeit wird an einer Reimplementierung des Graphers für TDL (s. Fragebogen unter Allegro CommonLISP auf den Sun-Maschinen (SPArc) gearbeitet—als Window- Standard wird dort CLIM eingesetzt
8. *Projekt:* DISCO/DFKI
9. *Input / Eingabeschnittstellen:* beliebige Datenstrukturen.
10. *Output / Ausgabeschnittstellen:* Graph bzw. versch. Views
11. *Aktueller Zustand:* fertiges Modul.
13. *Dokumentation:* teilweise vorhanden.
16. *Fremdinstallationen:* siehe Fragebogen zu TDL bzw. TDL ExtraLight
17. *Benutzte Techniken und zugrundeliegende Theorien:* Fokussierung und multiple Views.
18. *Allgemeine Beschreibung:* Visualisierung von Hierarchien und andere Beziehungen zwischen Objekten. Erzeugung und Verwaltung mehrerer Views eines Graphen. Hierarchisierung von Views. Definition von Mouse-Methods, diese sind Lisp-Prozeduren, die an Knoten gebunden werden und maus-gesteuert aktivierbar sind. Maussensitive Knoten: manuell positionierbar. Interaktives Update von View Parametern. Dynamisches Adaptieren des Layouts (Knoten des Graphen können mit der Mouse bewegt werden, etc.)

8.7 Testtool

1. *Name der Komponente:* Testtool
2. *Zweck der Komponente:* Integritaetstest des Unifikators
3. *Entwickelt und implementiert durch:* Christoph Weyers
4. *Anschrift, Kontaktperson, e-mail:* weyers@dfki.uni-sb.de
5. *Implementierungssprache:* Common Lisp
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Sparc-Station / SunOS 4.1
7. *Beginn / Ende der Entwicklung:* 01/92 - 03/92
8. *Projekt:* DISCO
9. *Input / Eingabeschnittstellen:* file, standard I/O
10. *Output / Ausgabeschnittstellen:* file, standard I/O
11. *Aktueller Zustand:* final release
12. *Abhängigkeiten / Anforderungen an andere Systeme:* keine
13. *Dokumentation:* REPORT verfuegbar
14. *Verfügbarkeit:* Weitergabe zu wiss. Zwecken
15. *Wartbarkeit:* kein warten
16. *Fremdinstallationen:* z. Zeit nicht verfuegbar
18. *Allgemeine Beschreibung:*

    Das Testtool unterstuetzt das Anlegen und Verwalten von Beispielstrukturen, mit deren Hilfe jederzeit die Konsistenz des UDINE-Unifikators getestet werden kann.

8.8 DiTo (Diagnostic Tool for German Syntax)

1. *Name der Komponente:* DiTo (Diagnostic Tool for German Syntax)
2. *Zweck der Komponente:* Fehlerdiagnose innerhalb der Syntaxkomponente von Systemen, die natuerliche Sprache verarbeiten; Kontrolle der Systemperformanz und Unterstuetzung der Konsistenzerhaltung bei der Verarbeitung syntaktischer Phaenomene; Entwicklung von Syntaxfragmenten
3. *Entwickelt und implementiert durch:* J.Nerbonne, K.Netter, A.K.Diagne, L.Dickmann, J.Klein
4. *Anschrift, Kontaktperson, e-mail:* DFKI/Saarbrücken, Judith Klein, klein@dfki.uni-sb.de
5. *Implementierungssprache:* AWK, YACC
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Unix auf SPARCstation, Sun OS Release 4.1
7. *Beginn / Ende der Entwicklung:* Februar 1991 / Februar 1992 (Kern)
8. *Projekt:* DISCO (DFKI)
9. *Input / Eingabeschnittstellen:* Anfragen in string-Form oder als ASCII-Datei
10. *Output / Ausgabeschnittstellen:* Ausgabe in string-Form oder als ASCII-Datei
11. *Aktueller Zustand:* konzeptuelles Kernschema fertig; einfache SQL-aehnliche Anfragesprache (AQL)fertig; Utility-Tools in Arbeit (Konsistenzpruefungen; Anfragenoptimierung steht noch aus); Portierung auf DOS-Version steht noch aus
12. *Abhängigkeiten / Anforderungen an andere Systeme:*
13. *Dokumentation:*

    J.Nerbonne, K.Netter, K.Diagne, L.Dickmann and J.Klein: A Diagnostic Tool for German Syntax, in J.Neal and S.Walter "Natural Language Processing Systems Evaluation Workshop"Rome Laboratory, Griffiss Air Force Base, RL-TR-91-362

    A.K. Diagne: DiTo - A Diagnostic Tool for German Syntax. Data Base and User's Manual. erscheint als: DFKI Technical Document, DFKI, Saarbruecken 1992

    J.Klein and L.Dickmann: DiTo - A Diagnostic Tool for German Syntax. Daten-Dokumentation: Verbrektion und Koordination. erscheint als: DFKI Technical Document, DFKI, Saarbruecken, 1992
14. *Verfügbarkeit:* public domain
15. *Wartbarkeit:* Hinzufuegen neuer Relationen fuer weitere syntaktische Bereiche in die DB fast voellig unabhaengig von bestehenden; Einfuegen neuer Daten ebenso unabhaengig

16. *Fremdinstallationen:* IAI in SB, CL-Lehrstuhl in SB, in Verhandlung: Koblenz, Institut fuer Computerlinguistik

17. *Benutzte Techniken und zugrundeliegende Theorien:* Theorie relationaler Datenbanken

18. *Allgemeine Beschreibung:* DiTo besteht aus: - einer relationalen Datenbank mit sql-aehnlicher Anfragesprache (AQL), die in AWK programmiert wurde (besondere String-Behandlung durch Matching- Funktionen moeglich) - Files mit linguistischen Daten (Beispielsaetze und syntaktische Annotierungen) aus den Bereichen Verbrektion und Koordination (in Arbeit: Funktionsverbgefuege, als naechstes geplant: NP-Syntax)

    Spezielle Fragen fuer Lexikon und Grammatik:

19. *Entwicklungsumgebung zur Erstellung der Einträge:* bisher: Emacs-Files fuer die linguistischen Daten; in Arbeit: Eintragungen ueber die Anfragesprache AQL

20. *Anzahl und Art der Einträge:* - bereits ca. 1000 deutsche Saetze zu Verbrektion und Koordination - syntaktische Merkmale zu diesen Syntaxbereichen und allgemeine Satzinformationen

Literaturangaben: vgl. Dokumentation

8.9 G_LOG

1. *Name der Komponente:* G_LOG

2. *Zweck der Komponente:* Erweitertes Prologsystem: Featureterme + Featureunifikation, einfache 'constraint resolution', Modulkonzept: Wissenspakete, Datenbankanschluß in Vorbereitung.

3. *Entwickelt und implementiert durch:* Dr. Helmut Gust

4. *Anschrift, Kontaktperson, e-mail:* Dr. Helmut Gust, Universität Osnabrück, Fachbereich Sprach- und Literaturwissenschaft, Arbeitsbereich Computerlinguistik und Künstliche Intelligenz, Postfach 4469, 4500 Osnabrück, Tel: 0541 - 969 2578, bitnet: lilogwls@dosunil, Fax: 0541 - 969 2500

5. *Implementierungssprache:* C

6. *Zugrundeliegende Hardware / Betriebssystem(e):* MSDOS, ATARI-TOS, Portierung auf AIX in Vorbereitung

7. *Beginn / Ende der Entwicklung:* 1986 ...

11. *Aktueller Zustand:* Stabile einsetzbare Version

13. *Dokumentation:* fast fertig

14. *Verfügbarkeit:* auf Anfrage

15. *Wartbarkeit:* auf Anfrage

16. *Fremdinstallationen:* Das System wird an der Universität Osnabrück im Arbeitsbereich CL & KI in Forschung un Lehre eingesetzt. Außerdem ist der Einsatz dieses Systems mit integrierter Datenbank in einem voraussichtlich im September anlaufenden Projekt, das sich mit der Integration unterschiedlicher lexikalischer Ressourcen in einer lexikalischen Wissensbank beschäftigt, geplant.

8.10 EGG - Editor für Generalisierte Phrasenstruktur-Grammatiken (GPSG)

1. *Name der Komponente:* EGG - Editor für Generalisierte Phrasenstruktur-Grammatiken (GPSG)
2. *Zweck der Komponente:* Verfassen von GPS-Grammatiken zur Verwendung in MUe-Systemen
3. *Entwickelt und implementiert durch:* Erich Ziegler, Tim Eckhart und Wilhelm Weisweber
4. *Anschrift, Kontaktperson, e-mail:* Erich Ziegler, TU Berlin, Projekt KIT-FAST, Sekr. 5-12, Franklinstr. 28/29, W-1000 Berlin 10, e-Mail: ez@cs.tu-berlin.de, Tel: (030) 314-73604/-27778
5. *Implementierungssprache:* Arity Prolog
6. *Zugrundeliegende Hardware / Betriebssystem(e):* AT/XT kompatibeler PC / MS-DOS 3.31
7. *Beginn / Ende der Entwicklung:* 1989 bis Febr. 1992
8. *Projekt:* Anapherninterpretation in der maschinellen Uebersetzung (KIT-FAST II, EUROTRA-D Begleitforschung Berlin)
9. *Input / Eingabeschnittstellen:* keine, bzw. vorhandene GPS-Grammatiken zum editieren.
10. *Output / Ausgabeschnittstellen:* GPSG in fuenf Teilen:
    - Merkmalsdefinition;
    - Aliase;
    - ID-Regeln;
    - LP-Aussagen;
    - FCRs.
11. *Aktueller Zustand:* einsatzfaehig.
12. *Abhängigkeiten / Anforderungen an andere Systeme:* keine.
13. *Dokumentation:* keine
14. *Verfügbarkeit:* kostenlos, gegen Rueckmeldung
15. *Wartbarkeit:* maessig
16. *Fremdinstallationen:* Universitaet Bochum, Slawistik

17. *Benutzte Techniken und zugrundeliegende Theorien:* konstruktive Version der GPSG

18. *Allgemeine Beschreibung:* Der Benutzer kann mit EGG eine GPS- Grammatik schreiben, bzw. editieren, aus der ein GPSG-Parser dann die syntaktischen Strukturen fuer Saetze erzeugen kann.

Literaturangaben:

Christa Hauenschild, Stephan Busemann: A constructive Version of GPSG for machine translation, in: E. Steiner, P. Schmidt, C. Zellinsky-Wibbelt: From Syntax to Semantics - Insights from Machine Translation, Frances Pinter, London 1988, S. 216-238

9 Schnittstellen und Systeme

9.1 NLI-AIDOS

1. *Name der Komponente:* NLI-AIDOS
2. *Zweck der Komponente:* Natürlichsprachliches Interface fuer das Informationssystem AIDOS
3. *Entwickelt und implementiert durch:* H. Helbig, H. Böttger, F. Ficker, F. Zämker, P.String (RPD und SRS Dresden)
4. *Anschrift, Kontaktperson, e-mail:* Dr. habil. Hermann Helbig, Software- und Systemhaus SRS, O-8072 Dresden, PSF 412, Tel: 4811-257, Fax 4811,202
5. *Implementierungssprache:* LISP
6. *Zugrundeliegende Hardware / Betriebssystem(e):* PC AT ab 2 MByte / MS-DOS
7. *Beginn / Ende der Entwicklung:* 1985 - 1991
8. *Projekt:* NLI-AIDOS
9. *Input / Eingabeschnittstellen:* Natürlichsprachliche Fragesätze, Aufforderungen
10. *Output / Ausgabeschnittstellen:* Semantische Zwischensprache (SZS)
11. *Aktueller Zustand:* Prototyp; Weiterentwicklung vorgesehen, im Augenblick aus finanziellen Gründen suspendiert.
12. *Abhängigkeiten / Anforderungen an andere Systeme:* luaffähiges AIDOS mit Dialogmodul auf gleichem Rechner
13. *Dokumentation:* Sammlung vo Themenberichten
14. *Verfügbarkeit:* nicht ausgeliefert
15. *Wartbarkeit:* wird noch gewartet
16. *Fremdinstallationen:* -
17. *Benutzte Techniken und zugrundeliegende Theorien:* Wortklassen - Experten; WCFA = Wortklassen gesteuerte funktionelle Analyse
18. *Allgemeine Beschreibung:* Das NLI gestattet eine natürlichsprachige Abfrage von Dokumenten - bzw. Faktenbeständen des IRS- AIDOS
19. *Entwicklungsumgebung zur Erstellung der Einträge:* Werkbank für den Lexikographen
20. *Anzahl und Art der Einträge:* ca. 2000

Literaturangaben:

Helbig H. "Künstliche Intelligenz und automatische Wissensverarbeitung". Verlag "Technik", Berlin 1991, Kapitel 7–8.

Helbig H. "Syntactic-semantic Analysis of Natural Language by a New Word-class Controlled Functional Analysis". Computers and AI, Bratislava 1986, Bd 5.

9.2 ADKMS-Datenbankinterface

1. *Name der Komponente:* ADKMS-Datenbankinterface
2. *Zweck der Komponente:* natuerlichsprachlicher Dialog mit einem erweiterten relationalen Datenbanksystem ORACLE (R)
3. *Entwickelt und implementiert durch:* SNI, Uni Hildesheim, Olivetti
4. *Anschrift, Kontaktperson, e-mail:* Konrad Jablonski, SNI AG - AP 333, Anwendersoftware und Projekte, KI-Zentrum Paderborn, Riemekestr. 160, 4790 Paderborn jablonski.pad@sni.de
5. *Implementierungssprache:* Prolog, Zusatzmodul Verbmorphologie in C rekursive Erweiterungen von SQL in MODULA-2
6. *Zugrundeliegende Hardware / Betriebssystem(e):* UNIX, SNI-SINIX-Maschinen (TARGON, MX, RM), UNIX-PC, SUN 3
7. *Beginn / Ende der Entwicklung:* ab 1.1.89 bis 28.2.1991
8. *Projekt:* ESPRIT-Projekt 311/ ADKMS Gesamtprojektleiter: Konrad Jablonski (ab 1.1.1990) weitere Partner: TA Triumph Adler AG, Olivetti, TU Berlin, Datamont
9. *Input / Eingabeschnittstellen:* natuerlichsprachliche Anfragen an das Informationssystem
10. *Output / Ausgabeschnittstellen:* formatierter dynamisch generierter NL-Antworttext
11. *Aktueller Zustand:* Projekt-Prototyp lauffaehig fuer eingeschraenkte Domaene
12. *Abhängigkeiten / Anforderungen an andere Systeme:* separate Einzelmodule (Parser, Datenbankexperte, SQL-Erweiterung, planbasierte Generierung, Textgenerierung) koennen bei Einhaltung der definierten, entkoppelten Schnittstellen portiert werden
13. *Dokumentation:* zahlreiche Fachliteratur/ TBL-Buch s. Literaturliste
14. *Verfügbarkeit:* Copyright SNI Einzellizenz auf Anfrage
15. *Wartbarkeit:* im Augenblick keine Wartung; nur im Rahmen eines Entwicklungsauftrags
17. *Benutzte Techniken und zugrundeliegende Theorien:* Definite Clause Grammar, semantisch gesteuerte Restrictionen, sprechakt-basiert, Nachempfindung deduktiver Datenbanken, SQL-Standard, Schema-/Planorientierung, Konzepthierarchien.
18. *Allgemeine Beschreibung:* Unterstuetzung von Multilingualitaet (Deutsch/Italienisch impl.) automatische Zuordnung Lexeme - DB-Konzepte regelbasierte Herleitung zusaetzlicher Informationen automatische Verallgemeinerung/Spezialisierung in der Konzepthierarchie gleiche semantische Reprdsentation als Output Parser und Input Generierung „echte"Generierung der Antworten Ueberbeantwortung und Infos bei unzureichenden Daten

19. *Entwicklungsumgebung zur Erstellung der Einträge:* modulares Teilsystem ALEXIS mit Desktop-Oberflaeche Erfragung morphologischer Merkmale Bearbeitung auch durch linguistische Laienmoeglich

20. *Anzahl und Art der Einträge:* nur morphologische Mindestangaben zur korrekten Flexion Zuordnung von Items der Datenbank zu Lexemen

Literaturangaben: (gesamte NL-Publikationen SNI)

(sortiert in chronologischer Reihenfolge)

K.Jablonski, A.Rau, H.Rvsner: COLING 86 - Bonn (Tagungsbericht); Rundbrief d. Fachausschusses KI der GI; Heft 45 / 1987; Oldenbourg München

K.Jablonski, A.Rau, J.Ritzke: Konzeption und Architektur des taktischen Textgenerierungssystems NUGGET, WISBER-Memo 12; Saarbrücken Mai 1987

K.Jablonski, A.Rau, J.Ritzke: Syntax- und Morphologieverarbeitung im Textgenerierungssystem NUGGET. in: Abstracts der Jahrestagung der Deutschen Gesellschaft für Sprachwissenschaft, Wuppertal Mdrz 1988; auch in: WISBER-Arbeitsunterlage U 21

K.Jablonski: Bericht über den 11th German Workshop on Artificial Intelligence (Sprachorientierte KI); in: LDV-Forum (5) 1988, Nummer 4, S. 48 - 52

K.Jablonski, A.Rau, J.Ritzke: Anwendung einer logischen Grammatik zur Generierung deutscher Texte. in: H.Trost (Hrsg.): 4.Österreichische Artificial-Intelligence-Tagung, Wien, August 88, Proceedings; Berlin etc.: Springer 1988; S. 83-93; auch als WISBER-Bericht 25, Juni 1988

K.Jablonski, A.Rau, J.Ritzke: NUGGET - Ein DCG-basiertes Textgenerierungssystem. WISBER-BERICHT 27, August 1988

K.Jablonski, A.Rau, J.Ritzke: Benutzerfreundlichkeit durch natürliche Sprache: NUGGET - ein Textgenerierungssystem. in: S.Savory (Hrsg.): Expertensysteme: Nutzen für Ihr Unternehmen; München-Wien: Oldenbourg, 2. überarbeitete u. erweiterte Aufl. 1989, S. 365 - 392

K.Lehner: Eine Lehrkomponente für Expertensysteme: eine Implementierung mit TWAICE. in: unix/mail (7) 1989; Heft 2; S. 52 - 56

H.U.Block, B.Frederking, M.Gehrke, H.Haugeneder, R.Hunze, K.Jablonski, A.Rau, J.Ritzke, S.Schachtl: Sprachanalyse und Textgenerierung im natürlich-sprachlichen Beratungssystem WISBER; in: W.Brauer/C.Freksa: Wissensbasierte Systeme, 3. Internationaler GI-Kongreß München 1989, Berlin, Heidelberg etc.: Springer 1989; S. 275 - 285; erscheint demndchst als Langfassung in: Informatik - Forschung und Entwicklung, Springer

U.Bauernfeind: Natürlich-sprachliche Textgenerierung macht Expertensys tem- Schlußfol-

gerungen durchschaubarer. in: unix/mail (7) 1989; Heft 4; S. 38-42

K.Jablonski, A.Rau, J.Ritzke: Wissensbasierte Textgenerierung - Linguistische Grundlagen und softwaretechnische Realisierung; Tübingen: Narr 1990

K.Lehner: Wissensbasierte Lehrsysteme; München: Oldenbourg 1990

H.-H.Pardey: Die „künstliche Intelligenz" spricht ganz normales Deutsch. in: Frankfurter Allgemeine Zeitung; Technik und Motor; 10.4.1990; S. T6

K.Jablonski, J.Samuel: Natural Language Information Access System; in: ESPRIT - Information Processing Systems; Brüssel 1990

9.3 METEXA

1. *Name der Komponente:* METEXA

2. *Zweck der Komponente:* Wissensbasierte Analyse radiologischer Befundungstexte und Generierung von Erwartungen zur Unterstuetzung von Systemen zur Erkennung kontinuierlich gesprochener Sprache

3. *Entwickelt und implementiert durch:* Martin Schroeder

4. *Anschrift, Kontaktperson, e-mail:* Martin Schroeder Universitaet Hamburg Fachbereich Informatik Arbeitsbereich "Natuerlichsprachliche Systeme"Bodenstedtstr. 16 W-2000 Hamburg 50 email: martin@nats4.informatik.uni-hamburg.de Tel: 040 / 4123 6146 Fax: 040 / 4123 6530

5. *Implementierungssprache:* Prolog (ProLog by BIM)

6. *Zugrundeliegende Hardware / Betriebssystem(e):* SUN SparcStation (SUN-OS 4.0x)

7. *Beginn / Ende der Entwicklung:* Jan 90 - Okt 92

8. *Projekt:* (METEXA)

9. *Input / Eingabeschnittstellen:* natuerlichsprachliche Texte

10. *Output / Ausgabeschnittstellen:* semantische Repraesentation

11. *Aktueller Zustand:* in Entwicklung

12. *Abhängigkeiten / Anforderungen an andere Systeme:* -

13. *Dokumentation:* Arbeitspapiere, gut dokumentierte Dateien

14. *Verfügbarkeit:* auf Anfrage

17. *Benutzte Techniken und zugrundeliegende Theorien:* - Bottom-Up Parser (BUP) in Prolog mit Graphunifikation - Conceptual Graphs (Sowa 1984) als Repraesentationssprache

18. *Allgemeine Beschreibung:* Wissensbasierte Analyse: - Radiologische Befundungstexte werden syntaktisch, semantisch, und unter Ausnutzung einer medizinischen Wissensbasis analysiert. Ergebnis ist eine semantische Repraesentation einer Aeusserung, die auf eine Datenbank oder Expertensystem etc. abgebildet werden kann. Die korrekte Interpretation wird ueberwiegend durch eine semantische Ueberpruefung auf der Wissensbasis ermittelt. Der Parser ermittelt nur Konstituentengrenzen und modifizierende Konstituenten. Keine morphosyntaktischen Merkmale, nur syntaktische Kategorien. Generierung von Erwartungen: - Auf Grund von Domaenenwissen ueber typische Formulierungsstrategien von Radiologen werden Erwartungen fuer die naechste Aeusserung erzeugt. Grundlegende Struktur ist die "Checkliste". Die Erwartungen werden nach ihrer Erzeugung auf semantischer Ebene und in die

lexikalische Ebene umgesetzt. Diese Erwartungsmengen dienen dazu, den Suchraum bei der Auswahl von Worthypothesen bei der Erkennung kontinuierlich gesprochener Sprache zu beschraenken. Dadurch wird eine ßemantisch-pragmatische Sensibilisierungërreicht.

19. *Entwicklungsumgebung zur Erstellung der Einträge:* keine (Editor)

20. *Anzahl und Art der Einträge:* ca. 1000 Vollformen

Literaturangaben:

Schroeder, M.: Knowledge-based Processing of Medical Language: A Language Engineering Approach, (submitted to GWAI-92).

9.4 Intercity-Zugauskunft

1. *Name der Komponente:* -

2. *Zweck der Komponente:* Fuehren eines Auskunftsdialogs ueber ein begrenztes Anwendungsgebiet (Intercity-Zugauskunft). Die akustische Eingabe ist kontinuierlich gesprochene Sprache.

3. *Entwickelt und implementiert durch:* Entwickelt und implementiert durch: AG Angewandte Informatik, Universitaet Bielefeld (Prof. Dr. Sagerer, Dr. Kummert) Informatik 5 (Mustererkennung), Universitaet Erlangen (Prof. Dr. Niemann, Dipl.-Inf Mast) im Rahmen des EVAR-Projekts

4. *Anschrift, Kontaktperson, e-mail:* Prof. Dr. G. Sagerer, AG. Angewandte Informatik, Technische Fakultaet, Postfach 10 01 31, W-4800 Bielefeld 1 e-mail: sagerer@techfak.uni-bielefeld.de

5. *Implementierungssprache:* C

6. *Zugrundeliegende Hardware / Betriebssystem(e):* Hardwareunabhaengig, Unix

7. *Beginn / Ende der Entwicklung:* 1981

9. *Input / Eingabeschnittstellen:* Wordlattice, Wortgraphen, n-beste Wortketten

10. Ouput / Ausgabeschnittstellen: Systemreaktion mittels synthetisierter Sprache

11. *Aktueller Zustand:* dialogfaehige Version in beinahe Echtzeit vorhanden. Verbesserungen in Richtung Echtzeit und erhoehte Dialogfaehigkeit werden vorgenommen.

13. *Dokumentation:* Theoretische Beschreibung ueber Veroeffentlichungen

14. *Verfügbarkeit:* nach Absprache

17. *Benutzte Techniken und zugrundeliegende Theorien:* Konstituentengrammatik, Tiefenkasustheorie, Dialogakte

18. *Allgemeine Beschreibung:*

    Das obige System soll einen natuerlichsprachlichen Auskunftsdialog ueber die Intercity-Zuege der Deutschen Bundesbahn fuehren und dabei kontinuierlich gesprochene Sprache in Telefonqualitaet sprecherunabhaengig verarbeiten. Die linguistische Analyse orientiert sich an dem geschichteten linguistischen Modell, bei dem das Wissen der verschiedenen Verarbeitungsebenen separat modelliert ist. Zwar erfolgt die Darstellung des syntaktischen, semantischen und pragmatischen Wissens in einem semantischen Netz, aber durch die Modellierung in verschiedenen Abstraktionsebenen erfolgt eine klare Trennung der Inhalte. Durch diese modulare Strukturierung wird die Wissensbasis aenderungsfreundlich, leicht erweiterbar und uebersichtlich gehalten. Waehrend der Analyse jedoch erfolgt keine sequentielle Abarbeitung der einzelnen Ebenen, sondern es werden fruehzeitig Vorhersagen aus "hoeheren" Ebenen der

linguistischen Wissensbasis erzeugt, um den Verarbeitungsaufwand zu begrenzen. Die aktuelle Realisierung dieser Wissensbasis umfasst eine Schnittstellenebene zur Worterkennung (Hypothesenebene), eine Ebene zur Repraesentation syntaktischer Konstituenten und spezieller Zeitangaben (Syntaxebene), eine Ebene zur Modellierung der allgemeinen Bedeutung von Verben und Nomina auf der Basis der Tiefenkasus Theorie (Semantikebene) und eine Ebene, in der anwendungsabhaengige Begriffe dargestellt sind (Pragmatikebene). Die Verbesserung des Dialogmodells ist in Bearbeitung.

Literaturangaben:

F. Kummert: Flexibele Steuerung eines sprachverstehenden Systems mit homogener Wissensbasis, Dissertation, Universitaet Erlangen-Nuernberg, 1991

G. Sagerer: Automatisches Verstehen gesprochener Sprache, Reihe Informatik, Band 74, Wissenschaftsverlag Mannheim/Wien/Zuerich

9.5 EVAR

1. *Name der Komponente:* EVAR
2. *Zweck der Komponente:* Ein sprachverstehendes System zum Führen eines informationsabfragenden Dialogs; Pilotanwendung InterCity-Zugauskunft
3. *Entwickelt und implementiert durch:* Lehrstuhl fuer Informatik 5 (Mustererkennung)
4. *Anschrift, Kontaktperson, e-mail:* Prof. H. Niemann Lehrstuhl fuer Informatik 5 (Mustererkennung) Martensstr. 3 8520 Erlangen

   Tel.: 09131/85 7774 Fax.: 09131/303811

   e-mail: niemann@informatik.uni-erlangen.de
5. *Implementierungssprache:* in C unter ERNEST
6. *Zugrundeliegende Hardware / Betriebssystem(e):* DEC-Rechner unter ULTRIX
7. *Beginn / Ende der Entwicklung:* 1979 / laufend
8. *Projekt:* Univ. Erlangen, BMFT, DFG
9. *Input / Eingabeschnittstellen:* Abtastwerte ueber AD/DA-Geraet der Firma Gradient Technology
10. *Output / Ausgabeschnittstellen:* Gewuenschte Zugauskunft als Vollsynthese
11. *Aktueller Zustand:* laufendes Software-Produkt
12. *Abhängigkeiten / Anforderungen an andere Systeme:* AD/DA- Geraet, Zugdatenbank der Deutschen Bundesbahn (Firma Hacon) Synthesegeraet der Daimler-Benz AG
13. *Dokumentation:* [NSE92]
14. *Verfügbarkeit:* ja, eingeschraenkt (siehe Punkt 12), auf Anfrage
16. *Fremdinstallationen:* ja
17. *Benutzte Techniken und zugrundeliegende Theorien:* siehe [NSE92]
18. *Allgemeine Beschreibung:* siehe [NSE92]

Literaturangaben:

[NSE92] E. Niemann, G. Sagerer, U. Ehrlich, E. G. Schukat-Talmazzini, F. Kummert The Interaction of Word Recognition and Linguistic Processing in Speech Understanding in P. Laface, R. De Mori Speech Recognition and Understanding, Springer, 1991.

9.6 Semantikbasiertes maschinelles Übersetzungssystem

1. *Name der Komponente:* -

2. *Zweck der Komponente:* Semantikbasiertes maschinelles Uebersetzungssystem. Grammatiktyp LFG, Semantiktyp DRT.

3. *Entwickelt und implementiert durch:* Top-down-Parser und Tomita-Parser: Andreas Eisele, Jochen Doerre Erweiterungen zum Top-down-Parser: Dieter Kohl Generator: Dieter Kohl Semantikkomponente: Walter Kasper, Kurt Eberle Erweiterung fuer LFG-Grammatiken zur Semantikkonstruktion: Walter Kasper

   Grammatiken, Parser und Schnittstellensoftware stammen aus anderen Projekten des IMS. Parser: Andreas Eisele, Jochen Doerre; Grammatiken (Deutsch/Französisch): Klaus Netter, Ursula Kaercher; Schnittstellensoftware: Dieter Kohl

4. *Anschrift, Kontaktperson, e-mail:* Anschrift: Institut fuer maschinelle Sprachverarbeitung - Computerlinguistik - Universitaet Stuttgart Azenbergstr. 12 D-7000 Stuttgart 1 Kontaktperson: Dieter Kohl e-mail: dieter@adler.philosophie.uni-stuttgart.de

5. *Implementierungssprache:* Prolog, Edinburgh Syntax. Bekanntermassen kompatibel mit Quintus Prolog 2.4, Quintus Prolog 3.1.1, Sicstus Prolog, CProlog.

6. *Zugrundeliegende Hardware / Betriebssystem(e):* Keine besonderen Hardwarevoraussetzungen, Betriebssystem UNIX (SUN-OS).

7. *Beginn / Ende der Entwicklung:* Beginn 1989

8. *Projekt:* SFB 340 'Sprachtheoretische Grundlagen fuer die Computerlinguistik', Teilprojekt B3 Rohrer/Wedekind.

9. *Input / Eingabeschnittstellen:* Durch '.' terminierte Saetze ueber Tastatureingabe oder ueber Textdateien mittels Eingabeumleitung innerhalb des Interpreters. Die Eingabe wird entweder als ein Kommando interpretiert oder als Satz der Sprache, fuer die eine Grammatik geladen ist.

10. *Output / Ausgabeschnittstellen:* Die Ausgabe erfolgt entweder auf den Bildschirm oder in eine Datei mittels Ausgabeumleitung innerhalb des Interpreters. Es lassen sich die Strukturen anzeigen, die zwischen den einzelnen Komponenten weitergereicht werden. Zwischenstrukturen werden als Prolog-Terme oder speziell dargestellte Feature-Strukturen (zum Teil auch als LaTex-Code in eine Datei) ausgegeben. Die Ausgabe der Generierung ist ein Satz der Zielsprache in normaler Textform.

11. *Aktueller Zustand:* Das Produkt befindet sich noch in der Entwicklung.

12. *Abhängigkeiten / Anforderungen an andere Systeme:* Verwendet die Schnittstellensoftware des Charon-Systems.

13. *Dokumentation:* Eine Zusammenstellung der Dokumentationen der einzelne Komponenten ist geplant.

14. *Verfügbarkeit:* Copyright Institut fuer maschinelle Sprachverarbeitung, wird im akademischen Bereich kostenlos sein.

15. *Wartbarkeit:* Variiert innerhalb der Teilkomponenten. Durch modularen Aufbau mit wenigen Aufrufpraedikaten im allgemeinen gut.

16. *Fremdinstallationen:* keine

17. *Benutzte Techniken und zugrundeliegende Theorien:* Generator: Basiert auf dem Algorithmus von Dieter Kohl (1991). Verwendet eine modifizierte top-down breadth-first Strategie, die mit einer Agenda und partiellen Loesungsmengen arbeitet. Der Generator ist universeller einsetzbar als der des Charon-Systems und soll nicht nur auf die LFG beschraenkt bleiben.

    Schnittstelle zum Benutzer: Kommandointerpreter, der eine Eingabe als Satz interpretiert, wenn sie nicht als Kommando erkannt wird.

18. *Allgemeine Beschreibung:*

    Der Generator besteht aus einem Compiler und einem Generatorkern, der die durch den Compiler aufgebauten Informationen interprtetiert. Der Generator eignet sich zur Generierung aus Merkmalsstrukturen, die in Bezug auf die Zielgrammatik sowohl unter- als auch ueberspezifizierten sein koennen.

    Allgemeine Verarbeitung: Ein Satz wird eingelesen und durch den Parser analysiert. Die LFG-Grammatik des Parsers beschreibt zu der syntaktischen funktionalen Struktur zusaetzlich eine semantische Struktur, sowie eine syntaktische Transferstruktur. Die semantische Struktur wird durch Resolutionskomponenten zur Anaphernresolution sowie der temporalen Resolution weiterbearbeitet und gegebenenfalls modifiziert. Die modifizierte semantische Struktur wird zusammen mit der syntaktischen Transferstruktur zu einer f-Struktur kombiniert, die dem Generator als Eingabe dient. Der Generator erzeugt aus dieser Eingabe einen Oberflaechensatz der Zielsprache.

19. *Entwicklungsumgebung zur Erstellung der Einträge:* Erstellen von LFG-Regeln und Lexikoneintraegen mittels Emacs-Editor in Form von Text-Dateien. Die Lexikoneintraege enthalten Templates die mit Hilfe eines UNIX-Shellscripts expandiert werden. Die Templates werden in einer separaten Textdatei vorher definiert, und sind Abkuerzungen fuer komplexe Merkmalsbeschreibungen.

20. *Anzahl und Art der Einträge:* (Sematikbasierter Transfer Deutsch-Franzoesisch)

    Deutsches Grammatikfragment: C-Struktur-Regeln: 29 Lexikoneintraege (Vollformen): 811 Templates: 329

    Franzoesisches Grammatikfragment: C-Struktur-Regeln: 38 Lexikoneintraege (Vollformen): 1027 Templates: 337

Literaturangaben:

Ronald M. Kaplan, Klaus Netter, Jürgen Wedekind, Annie Zaenen (1989): Translation by Structural Correspondences, in: proc. of the 4th eacl, Manchester

Dieter Kohl (1991): Generierung aus unter- und überspezifizierten Merkmalsstrukturen in LFG, Institut für maschinelle Sprachverarbeitung, Universität Stuttgart, Arbeitspapiere des SFB 340 *S*prachtheoretische Grundlagen für die Computerlinguistik Bericht Nr. 9

Dieter Kohl (1992): Generation from under- and overspecified Structures, in: proc. of the 14th coling, Ed: Zampolli, A., to appear, Université de Nantes

Klaus Netter, Jürgen Wedekind (1986): An LFG-based approach to machine translation, in Proceedings of IAI-MT 86, Saarbrücken, West Germany

Jürgen Wedekind (1990): Unifikationsgrammatiken und ihre Logik, Dissertation an der Universität Stuttgart

9.7 Charon-System

1. *Name der Komponente:* Charon-System

2. *Zweck der Komponente:* System zur transferbasierten maschinellen Uebersetzung mittels LFG-Grammatiken. Durch den modularen Aufbau eignet sich das System auch zum einzelsprachlichen Testen von Grammatiken und laesst sich als Testumgebung fuer den Einsatz von Parsern, Generatoren und andere NL-Komponenten, wie z.B. Semantikkomponenten und Morphologiekomponenten verwenden. Die Minimalkonfiguration der Entwicklungsumgebung besteht aus einem Kommandointerpreter, einem Parser und einem Generator.

3. *Entwickelt und implementiert durch:* Schnittstellensoftware: Dieter Kohl Top-down-Parser: Andreas Eisele, Jochen Doerre Erweiterungen zum Top-down-Parser: Dieter Kohl Top-down breadth-first Generator: Juergen Wedekind, Stefan Momma/Jochen Doerre, Dieter Kohl Unix-Tools zur Template-Expansion: Doug Jones Makefiles zur einfachen Installation und zum Erzeugen von Grammatikdirectories: Dieter Kohl

   LFG-Grammatiken fuer die Uebersetzung Deutsch/Franzoesisch

   Grammatiken: Klaus Netter, Sybille Scheub, Veerle van Geenhoven, Ursula Kaercher, Anette Frank, Thilo Tappe, Judith Meier. Transfer: Klaus Netter, Juergen Wedekind, Thilo Tappe, Anette Frank, Judith Meier

4. *Anschrift, Kontaktperson, e-mail:* Anschrift: Institut fuer maschinelle Sprachverarbeitung - Computerlinguistik - Universitaet Stuttgart Azenbergstr. 12 D-7000 Stuttgart 1 Kontaktperson: Dieter Kohl e-mail: dieter@adler.philosophie.uni-stuttgart.de

5. *Implementierungssprache:* Prolog, Edinburgh Syntax. Bekanntermassen kompatibel mit Quintus Prolog 2.4, Quintus Prolog 3.1.1, Sicstus Prolog, CProlog.

   Das Einrichten von Subdirectories zur Erstellung von Grammatikfragmenten ist ueber Makefiles automatisiert.

6. *Zugrundeliegende Hardware / Betriebssystem(e):* Keine besonderen Hardwarevoraussetzungen, Betriebssystem UNIX (SUN-OS).

7. *Beginn / Ende der Entwicklung:* 1985 / 1992

8. *Projekt:* Stuttgarter Teilprojekt der EUROTRA-D Begleitforschung

9. *Input / Eingabeschnittstellen:* Durch '.' terminierte Saetze ueber Tastatureingabe oder ueber Textdateien mittels Eingabeumleitung innerhalb des Interpreters. Die Eingabe wird entweder als ein Kommando interpretiert oder als Satz der Sprache, fuer die eine Grammatik geladen ist.

10. *Output / Ausgabeschnittstellen:* Die Ausgabe erfolgt entweder auf den Bildschirm oder in eine Datei mittels Ausgabeumleitung innerhalb des Interpreters. Es lassen sich die Strukturen anzeigen, die zwischen den einzelnen Komponenten weitergereicht werden. Zwischenstrukturen werden als Prolog-Terme oder speziell dargestellte Feature-Strukturen (zum Teil auch als LaTex-Code in eine Datei) ausgegeben. Die Ausgabe der Generierung ist ein Satz der Zielsprache in normaler Textform.

11. *Aktueller Zustand:*

    Die Entwicklungsumgebung ist am IMS im Einsatz, Teile davon werden jedoch weiterentwickelt. Innerhalb des Projekts wird mit Grammatiken fuer das Sprachpaar Deutsch/Franzoesisch gearbeitet.

    Die Schnittstellensoftware wird gelegentlich an zusaetzliche Anforderungen, die durch Aenderungen an den Teilkomponenten entstehen, angepasst, oder der Kommandointerpreter wird an Benutzerwuensche angepasst.

12. *Abhängigkeiten / Anforderungen an andere Systeme:*

    Parser und Generator lassen sich unabhaengig vom gesamten Entwicklungssystem verwenden. Beide verwenden jedoch das gleiche Modul zum Einlesen von LFG-Grammatiken.

13. *Dokumentation:*

    Fuer die Bedienung der Entwicklungsumgebung existiert innerhalb des Sys tems ein einfaches Hilfesystem. Eine Benutzungsanleitung fuer den Einstieg ist in Arbeit.

    Die Installation des Systems auf UNIX-Rechnern ist ausfuehrlich in entsprechenden Readme- und Installations-Dateien dokumentiert.

    Die einzelnen Softwarekomponenten sind unterschiedlich gut dokumentiert.

    Eine Zusammenstellung der Dokumentationen der einzelne Komponenten ist geplant.

    Die Grammatiken werden derzeit dokumentiert.

14. *Verfügbarkeit:* Copyright Institut fuer maschinelle Sprachverarbeitung, kostenlos im akademischen Bereich.

15. *Wartbarkeit:* Variiert innerhalb der Teilkomponenten. Durch modularen Aufbau mit wenigen Aufrufpraedikaten im allgemeinen gut.

16. *Fremdinstallationen:* ca. 3 Kopien

17. *Benutzte Techniken und zugrundeliegende Theorien:*

    Top-down-Parser: Aus dem Format einer LFG-Grammatik wird ein Prolog- Programm aehnlich wie ein DCG-Parser erzeugt, das einen Top-down-Parser implementiert, in dem parallel zur C-Struktur die F-Struktur erzeugt wird. In der Erweiterung werden Lazy-Evaluation-Techniken eingesetzt, um Listenkonkatenationen und variable Pfadangaben (z.B. fuer functional uncertainty) zu verarbeiten.

Generator: Basiert auf dem Algorithmus von Juergen Wedekind (1986, 1988) zur Generierung durch Ableitung. Eine LFG-Grammatik wird in eine prologlesbare Form uebersetzt und es werden eine Reihe von Informationen aus der Grammatik extrahiert, die fuer die Generierung gebraucht werden. Die Grammatik liegt in Form von Prologfakten vor, und wird bei der Generierung durch eine top-down breadth-first Verarbeitungsstrategie abgearbeteitet, die implizit head-driven ist, d.h. der relevante Kopf wird durch die Verarbeitungsstrategie automatisch gefunden, und z.B. das Verb eines Satzes zuerst generiert.

Schnittstelle zum Benutzer: Kommandointerpreter, der eine Eingabe als Satz interpretiert, wenn sie nicht als Kommando erkannt wird.

Der Transfer basiert auf dem in Kaplan/Netter/Wedekind/Zaenen89 beschriebenen Ansatz.

18. *Allgemeine Beschreibung:*

Zu den Teilmodulen siehe 17.

Das Charon-System als Entwicklungsumgebung fuer einzelsprachliche Grammatiken und Transfergrammatiken: Auf Betriebssystemsebene ist das Charon-System auf eine Reihe von Subdirectories verteilt. So existieren das Subdirectory, das die Prolog-Sourcen fuer das Charon-System enthaelt, Subdirectories, in denen UNIX-Kommandos und saved states fuer das Charon-System lokalisiert sind, sowie ein Subdirectory, unterhalb von dem einzelne Grammatiken oder Grammatikpaare fuer die maschinelle Uebersetzung verwaltet werden. Fuer den Grammatikentwickler ist nur das letztgenannte Subdirectory 'grammars' relevant. Im Subdirectory 'grammars' kann jeder Benutzer mit der entsprechenden Gruppenzugehoerigkeit sich ein Subdirectory mit Hilfe eines Make-Kommandos einrichten. Dabei gibt der Benutzer an, ob er sein Subdirectory fuer eine einzelsprachliche Anwendung benoetigt oder fuer eine Transferbasierte Uebersetzung. Entsprechend diesen Angaben wird sein Subdirectory eingerichtet, wobei darin wiederum Subdirectories fuer die Grammatiken, Dateien zum Laden seines Systems in den Prolog-Teil des Charon-Systems, sowie ein Makefile und Readme-File generiert werden. Der Benutzer erstellt seine LFG-Grammatiken in Form von UNIX Textdateien mit dem fuer ihn gewohnten Editor. Fuer Merkmalsbeschreibungen innerhalb von Lexikoneintraegen koennen parametrisierte Templates verwendet werden, die der Benutzer in einem separaten File definiert. Derzeit muessen die Templates noch vor dem Einlesen in Prolog expandiert werden, was mittels eines UNIX-Kommandos, das im Charon-System definiert ist, geschieht, wobei aus einer Datei, die Template-Aufrufe enthaelt, eine neue Datei erzeugt wird, die keine Template-Aufrufe mehr enthaelt. Nachdem die Grammatikdateien erstellt und expandiert worden sind, kann der Benutzer in seinem Grammatiksubdirectory charon_raw starten, einen Prolog-saved-state, der den Prologteil des Charon-Systems enthaelt. Nach Aufruf des Kommandointerpreters genuegt das Einlesen einer Definitionsdatei, die Kommandos des Interpreters enthaelt, um den Grammatiken fuer die Analyse und Generierung (die auch die gleichen sein koennen, aber nicht sein muessen) Namen zu geben und fuer das Charon-System zu spezifizieren, sowie die Compilierung der Grammatiken zu starten. Bei Aenderungen an einer

Grammatik sind diese im Texteditor vorzunehmen, und die Grammatik neu zu compilieren. Dabei koennen bei den Parsern auch nur die Files neu compiliert werden, die geaendert wurden, beim Generator ist dies derzeit noch nicht moeglich. Es lassen sich grundsaetzlich vom Kommandointerpreter alle UNIX-Kommandos ansprechen, wobei in der Regel ein einleitendes Prefixkommando zu verwenden ist. Ebenso lassen sich die wichtigsten Prolog-Praedikate zum Debuggen von Prolog-Programmen im Kommandointerpreter aktivieren, und Prolog-Dateien koennen direkt geladen werden. Zum Testen einer Grammatik gibt der Benutzer Saetze ein, die analysiert und entsprechend von Flageinstellungen, die der Benutzer beliebig aendern kann, weiterverarbeitet werden. Im Transferfall sieht ein Durchlauf z.B. folgendermassen aus: Satzeingabe -¿ Analyse -¿ Extraktion der Transferstruktur -¿ Satzgenerierung aus Transferstruktur Ueber Backtracking lassen sich saemtliche Generierungen und saemtliche Analysen erhalten, wobei ueber ein Flag gesteuert werden kann, ob nach der jeweils ersten Loesung abgebrochen werden soll, oder das Backtracking sonst beschraenkt wird. Saemtliche Zwischenstrukturen koennen bei Bedarf ausgegeben werden, und ueber Ein-Ausgabeumleitung lassen sich z.B. Testlaeufe und ihre Ergebnisse im Batchbetrieb behandeln. Zu allen Kommandos und Flags, sowie den allgemeinen Moeglichkeiten des Kommandointerpreters, steht dem Benutzer innerhalb des Interpreters Hilfsinformation zur Verfuegung.

19. *Entwicklungsumgebung zur Erstellung der Einträge:*

    Erstellen von LFG-Regeln und Lexikoneintraegen mittels Emacs-Editor in Form von Text-Dateien. Die Lexikoneintraege enthalten Templates die mit Hilfe eines UNIX-Shellscripts expandiert werden. Die Templates werden in einer separaten Textdatei vorher definiert, und sind Abkuerzungen fuer komplexe Merkmalsbeschreibungen.

20. *Anzahl und Art der Einträge:*

    (Transfer Deutsch-Franzoesisch)

    Deutsches Grammatikfragment: C-Struktur-Regeln: 53 Lexikoneintraege (Vollformen): 1076 Templates: 267

    Franzoesisches Grammatikfragment: C-Struktur-Regeln: 58 Lexikoneintraege (Vollformen): 1651 Templates: 459

Literaturangaben:

Andreas Eisele, Jochen Dörre (1986): A Lexical Functional Grammar System in Prolog, in: Proceedings of the 11th International Conference on Computational Linguistics, 551–553, University of Bonn, West Germany

Jochen Dörre, Stefan Momma (1987): Generierung aus f-Strukturen als strukturgesteuerte Ableitung, in: Proceedings of GWAI-87, Geseke, West Germany

van Geenhoven, Veerle (1990): Zur Syntax der Pronominaladverbien und extraponierten Komplementsätzen in LFG, Institut für maschinelle Sprachverarbeitung, Universität Stuttgart, Manuskript

Ullrich Heid, Klaus Netter, Jürgen Wedekind (1988): Zur Funktionsweise des Transfers auf f-Strukturen, EUROTRA-D Working Papers 6, Saarbrücken

Kaercher (1988): Französische Partizipialkonstruktionen in LFG, Institut für maschinelle Sprachverarbeitung, Universität Stuttgart, Manuskript

Ronald M. Kaplan, Klaus Netter, Jürgen Wedekind, Annie Zaenen (1989): Translation by Structural Correspondences, in: Proceedings of the 4th Conference of the European Chapter of the Association for Computational Linguistics, Manchester

Dieter Kohl, Stefan Momma (1992): LFG based Generation in ACORD, in: The Construction of a Natural Language and Graphic Interface – Results and perspectives from the ACORD project, Ed: Bes, Gabriel, Part Generation in ACORD, Chapter 5. Springer, to appear

Stefan Momma, Jochen Dörre (1987): Generation from f-Structure s, in: Categories, Polymorphism and Unification, Ed: Klein, v. Benthem, Cognitive Science Centre, University of Edinburgh and Institute for Language, Logic and Information, University of Amsterdam

Klaus Netter, Jürgen Wedekind (1986): An LFG-based approach to machine translation, in: Proceedings of IAI-MT 86, Saarbrücken, West Germany

Scheub, Sybille (1986): Eine Grammatik des Französischen in LFG, Institut für maschinelle Sprachverarbeitung, Universität Stuttgart, Manuskript, Dokumentation des aktuellen Standes

Jürgen Wedekind (1986): A Concept of Derivation for LFG, in: Proceedings of the 11th International Conference on Computational Linguistics, 486–489, Institut für Kommunikationsforschung und Phonetik, University of Bonn, West Germany

Jürgen Wedekind (1988): Generation as Structure Driven Derivation, in: Proceedings of the 12th International Conference on Computational Linguistics, 732–737, Budapest, Hungary

Jürgen Wedekind (1990): Unifikationsgrammatiken und ihre Logik, Dissertation an der Universität Stuttgart

9.8 Interpreter und Entwicklungsumgebung fuer nicht konfluente Termersetzungssysteme in der maschinellen Sprachuebersetzung

1. *Name der Komponente:* Interpreter und Entwicklungsumgebung fuer nicht konfluente Termersetzungssysteme in der maschinellen Sprachuebersetzung
2. *Zweck der Komponente:* Parsing, semantische Analyse, konzeptuelle Analyse, Transfer und Generierung
3. *Entwickelt und implementiert durch:* Wilhelm Weisweber
4. *Anschrift, Kontaktperson, e-mail:* Wilhelm Weisweber TU Berlin Projekt KIT-FAST, Sekr. FR 5-12 Franklinstr. 28/29 W-1000 Berlin 10 ww@kit.cs.tu-berlin.de
5. *Implementierungssprache:* Quintus Prolog / Arity Prolog
6. *Zugrundeliegende Hardware / Betriebssystem(e):* Sun / Unix oder AT/XT kompatibler PC / MS-DOS 3.31
7. *Beginn / Ende der Entwicklung:* 1987 - heute
8. *Projekt:* Anapherninterpretation in der maschinellen Uebersetzung (KIT-FAST II, EUROTRA-D Begleitforschung Berlin)
9. *Input / Eingabeschnittstellen:* Ausgangsstrukturen (DAGs), z.B. - syntaktische Strukturen (GPSG), - Funktor-Argument-Strukturen (FAS). Syntax der Zielstrukturen, Lexikon der Zielstruktur, Termersetzungssystem,
10. *Output / Ausgabeschnittstellen:* Zielstrukturen (DAGs), z.B. - syntaktische Strukturen (GPSG) - Funktor-Argument-Strukturen (FAS) - konzeptuelle Strukturen (ABox)
11. *Aktueller Zustand:* z.T. noch in der Entwicklung
12. *Abhängigkeiten / Anforderungen an andere Systeme:* keine
13. *Dokumentation:* keine explizite Systemdokumentation
14. *Verfügbarkeit:* kostenlos, gegen Rueckmeldung
15. *Wartbarkeit:* im Moment nur durch Entwickler
16. *Fremdinstallationen:* DFKI Saarbruecken Universitaet Koblenz-Landau
17. *Benutzte Techniken und zugrundeliegende Theorien:* Termersetzung (Deduktionssysteme)

18. *Allgemeine Beschreibung:* Das System wird als Kern eines experimentellen maschinellen Uebersetzungssystems eingesetzt und fuehrt alle erforderlichen Strukturuebergange aus. Durch Vorverarbeitung der Regelsysteme ist eine sehr effiziente Interpretation der Termersetzungsregeln moeglich.

19. *Entwicklungsumgebung zur Erstellung der Einträge:* Die Entwicklungsumgebung kann auch fuer lexikalische Regeln verwendet werden.

20. *Anzahl und Art der Einträge:* ca. 220 lexikalische Abbildungsregeln aller Strukturuebergaenge

Literaturangaben:

Wilhelm Weisweber: Transfer in Machine Translation by Non-Confluent Term-Rewrite Systems, Procs. der GWAI-89, Eringerfeld 1989, S. 264-269

Wilhelm Weisweber, Christa Hauenschild: A Model of Multi-Level Transfer for Machine Translation and Its Partial Realization, KIT-Report 77, TU Berlin 1990 und erscheint in den Procs. des Seminars "Computers & Translation '89", Tiflis 1989

Wilhelm Weisweber: Term-Rewriting as a Basis for a Uniform Architecture in Machine Translation, erscheint in den Procs. der Coling-92, Nantes 1992

9.9 CAT2

1. *Name der Komponente:* CAT2
2. *Zweck der Komponente:* Multilinguale MUe
3. *Entwickelt und implementiert durch:* Randall Sharp und andere
4. *Anschrift, Kontaktperson, e-mail:* Randall Sharp , IAI Martin-LUther-Strasse 14 6600 Saarbruecken 3 West Germany e-mail: randy@iai.uni-sb.de
5. *Implementierungssprache:* SICStus PROLOG
6. *Zugrundeliegende Hardware / Betriebssystem(e):* In der Praxis: Sunworkstations UNIX In der Theorie: Alle Maschinen und Betriebssysteme, unter denen SICStus laeuft.
7. *Beginn / Ende der Entwicklung:* 1985/ kein Ende abzusehen
8. *Projekt:* Eurotra und andere
9. *Input / Eingabeschnittstellen:* ASCII Text
10. *Output / Ausgabeschnittstellen:* ASCII Text
11. *Aktueller Zustand:* In Entwicklung
12. *Abhängigkeiten / Anforderungen an andere Systeme:* s.o.
13. *Dokumentation:* Randall Sharp, CAT2 Reference Manual Version 2.1, IAI Saarbruecken.
14. *Verfügbarkeit:* Copyright IAI Saarbruecken, Preis ????
16. *Fremdinstallationen:* IAI Martin-Luther-Strasse 14 Saarbruecken, Germany

    Prof. Pflueger Universitaet Bremen Bremen, Germany

    Prof. Batori Universitaet Koblenz Koblenz, Germany

    Prof.Dr. Koenitz Universitaet Leipzig Leipzig, Germany

    Dr. Mounira Loughraieb Universite de Nancy Nancy, France

    Eurotra-France Maison de l'homme Boulevard Raspail Paris, France

    Eurotra-England University of Essex Colchester, England

    Eurotra-Espana Barcelona, Spain

    Eurotra-Portugal Lisbon, Portugal

    Dr. Carl Brown SICS P.O.Box 1263 S-164 28 Kista, Sweden

    Dr. Sean O'Nuallain School of Computer Applications Dublin City University Dublin 9, Ireland

    Prof. Harvey Abramson Institute of Industrial Science University of Tokyo Tokyo, Japan

17. *Benutzte Techniken und zugrundeliegende Theorien:* Unifikations- und Constraint-basierter Formalismus Mischung aus HPSG, GB und Eurotra-Linguistik

18. *Allgemeine Beschreibung:* s. Literatur

19. *Entwicklungsumgebung zur Erstellung der Einträge:* Lextool, kann mit verschiedenen Definitionen fuer Lexikonstrukturen ausgestattet werden.

20. *Anzahl und Art der Einträge:* Von Sprache zu Sprache variierend, bis zu 1300 Woertern.

    Grosse Grammatiken Deutsch, Englisch, Franzoesisch Kleine Grammatiken Italienisch, Spanisch, Portugiesisch, Russisch, Niederlaendisch, Japanisch

Literaturangaben:

ALSHAWI, H., D.J.Arnold, R.Backofen, D.M.Carter, J.Lindop, K.Netter, S.G.Pulman, J.Tsujii, H.Uszkoreit (1991): "ET6/1 Formalism Study - Final Report", CEC, DG XIII, Luxembourg.

ARNOLD, D., L.Jaspaert, R.Johnson, S.Krauwer, M.Rosner, L.des Tombe, G.B.Varile und S.Warwick (1985): "A MU1 View of the <C,A>,T Framework in EUROTRA", Proceedings of the Conference on Theoretical und Methodological Issues in Machine Translation of Natural Languages, Hamilton, NY, pp. 1-14.

ARNOLD, D., S.Krauwer, M.Rosner, L.des Tombe und G.B.Varile (1986): "The ¡C,A¿,T Framework in EUROTRA: A Theoretically Committed Notation for MT", Proceedings of COLING'86, Bonn, pp. 297-303.

Haller, J. 1989. EUROTRA - Das Forschungs- und Entwicklungsprojekt der EG zur Maschinellen Uebersetzung: Franzoesisch-deutsche Uebersetzung mit der Seitenlinie CAT2. In: Rolshoven und Seelbach (eds.), "Romanistische Computerlinguistik - Theorien und Implementation", Max Niemeyer Verlag, Tuebingen.

Haller, J.1991. EUROTRA - O projeto de pesquisa e desenvolvimento em traduao automatica da Commissao Europeia. Exemplos de Traducao Portugues-Alemao. In: Revista "Letras de Hoje", PUC Porto Alegre, 1991.

Mesli, N. 1990. Analyse et traduction automatique de constructions à verbe support dans le formalisme CAT2. Article présenté au 9ème Colloque Européen sur la Grammaire et le Lexique Comparés des Langues Romanes, La Londe-Les-Maures.

MESLI, N. (1991): Analyse et Traduction Automatique de Constructions à Verbe Support dans le Formalisme CAT2, EUROTRA-D Working Paper 19b, IAI, Saarbruecken; also as Funktionsverbgefuege in der maschinellen Analyse und Uebersetzung: linguistische Beschreibung und Implementierung im CAT2-Formalismus, EUROTRA-D Working Paper 19.

Schuetz, J. und Sharp, R. 1989. CAT2 - Komplexitaet eines Formalismus fuer multilinguale MUe. In: J.Schuetz (ed.), Workshop Semantik und Transfer. EUROTRA-D Working Paper No.6, IAI, Saarbruecken.

Sharp, R. 1988. CAT2 - Implementing a Formalism for Multi-Lingual MT. Proceedings of the 2nd International Conference on Theoretical und Methodological Issues in Machine Translation of Natural Languages, Pittsburgh, PA.

Sharp, R. 1989. CAT2 - A Formalism for Multilingual Machine Translation. Proceedings of the International Seminar on Machine Translation, Tbilisi, Georgia, USSR.

Sharp, R. 1990. Modelling GB in the CAT2 Machine Translation System. Workshop on GB-Parsing, June 15-16, University of Geneva.

Sharp, R. 1990. The CAT2 Machine Translation System. Proceedings of COLING'90, Helsinki.

SHARP, R. (1991): "CAT2: An Experimental Eurotra Alternative", Machine Translation, 6-3, pp. 215-228.

SHARP,R. und STREITER,O. 1992. Simplifying the Complexity of Machine Translation. To appear in: META - Specail issue on Machine Translation.

9.10 IBM SAA LanguageAccess

1. *Name der Komponente:* IBM SAA LanguageAccess
2. *Zweck der Komponente:* Datenabfrage in natürlicher Sprache (Engl Deutsch)
3. *Entwickelt und implementiert durch:* IBM Nordic Laboratories Liding: Schweden in Zusammenarbeit mit dem Wissenschaftlichen Zentrum der IBM Deutschland und IBM Research Yorktown Heights
4. *Anschrift, Kontaktperson, e-mail:* M.Zoeppritz ZOE@GYSVMHD1
5. *Implementierungssprache:* Prolog
6. *Zugrundeliegende Hardware / Betriebssystem(e):*MVS VMXA OS/2 DB/2 SQL/DS
7. *Beginn / Ende der Entwicklung:* ab 1985
9. *Input / Eingabeschnittstellen:* Query Interface oder über Appl Progr I (API)
10. *Output / Ausgabeschnittstellen:* QMF oder AS
11. *Aktueller Zustand:* Produkt Rel 1.1
12. *Abhängigkeiten / Anforderungen an andere Systeme:*s.o.
13. *Dokumentation:* Überblick, Benutzerhandbücher
14. *Verfügbarkeit:* C IBM, Preis hängt von der Maschinengröße ab
15. *Wartbarkeit:*durch IBM
17. *Benutzte Techniken und zugrundeliegende Theorien:* verschiedene
18. *Allgemeine Beschreibung:* Komponenten: Engine: Sprach-und-Anwendungsunabhängige Architektur Sprachabhängige Syntax interagiert mit einer kompositionellen Semantik. Diese verwendet ein sprachabhängiges Basismodell erweitert um ein Modell der Begriffe der jeweiligen Anwendung zur Erstellung einer semantischen Repräsentation der Eingabe (in CLF conceptual Logical Form). Diese wird mithilfe der im Anwendungsmodell angegebenen SQL-Fragmente in eine andere Form (DBLF data base logical form) überführt und dann rekursiv in SQL überführt und optimiert. QI Query Interface CT Customization Tool zur Anpassung an eine gegebene Datenbank - Erstellung eines Modells der Anwendung (Vokabular, Bezüge der Begriffe zur Datenbank, Bezüge der Begriffe untereinander)
19. *Entwicklungsumgebung zur Erstellung der Einträge:* CT (Customization Tool) auf PS/2 unter OS/2 Presentation Manager
20. *Anzahl und Art der Einträge:* Basismodell ca 500, Benutzermodell je nach Anwendung Lexikoneinträge für die W:rter mit Morphologie und Komplementen/Adjunkten Abbildung zwischen W:rtern und Begriffen, Begriffe mit Beziehungen zur Basishierarchie und zu anderen Begriffen u.a. zur Filterung (Selektionsrestriktionen)

Komponentenindex

Deutsches
Forschungszentrum
für Künstliche
Intelligenz GmbH

DFKI
-Bibliothek-
PF 2080
D-6750 Kaiserslautern
FRG

DFKI Publikationen

Die folgenden DFKI Veröffentlichungen sowie die aktuelle Liste von allen bisher erschienenen Publikationen können von der oben angegebenen Adresse bezogen werden.
Die Berichte werden, wenn nicht anders gekennzeichnet, kostenlos abgegeben.

DFKI Publications

The following DFKI publications or the list of all publisched papers so far can be ordered from the above address.
The reports are distributed free of charge except if otherwise indicated.

DFKI Research Reports

RR-91-14
Peter Breuer, Jürgen Müller: A Two Level Representation for Spatial Relations, Part I
27 pages

RR-91-15
Bernhard Nebel, Gert Smolka:
Attributive Description Formalisms ... and the Rest of the World
20 pages

RR-91-16
Stephan Busemann: Using Pattern-Action Rules for the Generation of GPSG Structures from Separate Semantic Representations
18 pages

RR-91-17
Andreas Dengel, Nelson M. Mattos:
The Use of Abstraction Concepts for Representing and Structuring Documents
17 pages

RR-91-18
John Nerbonne, Klaus Netter, Abdel Kader Diagne, Ludwig Dickmann, Judith Klein:
A Diagnostic Tool for German Syntax
20 pages

RR-91-19
Munindar P. Singh: On the Commitments and Precommitments of Limited Agents
15 pages

RR-91-20
Christoph Klauck, Ansgar Bernardi, Ralf Legleitner
FEAT-Rep: Representing Features in CAD/CAM
48 pages

RR-91-21
Klaus Netter: Clause Union and Verb Raising Phenomena in German
38 pages

RR-91-22
Andreas Dengel: Self-Adapting Structuring and Representation of Space
27 pages

RR-91-23
Michael Richter, Ansgar Bernardi, Christoph Klauck, Ralf Legleitner: Akquisition und Repräsentation von technischem Wissen für Planungsaufgaben im Bereich der Fertigungstechnik
24 Seiten

RR-91-24
Jochen Heinsohn: A Hybrid Approach for Modeling Uncertainty in Terminological Logics
22 pages

RR-91-25
Karin Harbusch, Wolfgang Finkler, Anne Schauder:
Incremental Syntax Generation with Tree Adjoining Grammars
16 pages

RR-91-26
M. Bauer, S. Biundo, D. Dengler, M. Hecking, J. Koehler, G. Merziger:
Integrated Plan Generation and Recognition
- A Logic-Based Approach -
17 pages

RR-91-27
A. Bernardi, H. Boley, Ph. Hanschke, K. Hinkelmann, Ch. Klauck, O. Kühn, R. Legleitner, M. Meyer, M. M. Richter, F. Schmalhofer, G. Schmidt, W. Sommer:
ARC-TEC: Acquisition, Representation and Compilation of Technical Knowledge
18 pages

RR-91-28
Rolf Backofen, Harald Trost, Hans Uszkoreit:
Linking Typed Feature Formalisms and Terminological Knowledge Representation Languages in Natural Language Front-Ends
11 pages

RR-91-29
Hans Uszkoreit: Strategies for Adding Control Information to Declarative Grammars
17 pages

RR-91-30
Dan Flickinger, John Nerbonne:
Inheritance and Complementation: A Case Study of *Easy* Adjectives and Related Nouns
39 pages

RR-91-31
H.-U. Krieger, J. Nerbonne:
Feature-Based Inheritance Networks for Computational Lexicons
11 pages

RR-91-32
Rolf Backofen, Lutz Euler, Günther Görz:
Towards the Integration of Functions, Relations and Types in an AI Programming Language
14 pages

RR-91-33
Franz Baader, Klaus Schulz:
Unification in the Union of Disjoint Equational Theories: Combining Decision Procedures
33 pages

RR-91-34
Bernhard Nebel, Christer Bäckström:
On the Computational Complexity of Temporal Projection and some related Problems
35 pages

RR-91-35
Winfried Graf, Wolfgang Maaß: Constraint-basierte Verarbeitung graphischen Wissens
14 Seiten

RR-92-01
Werner Nutt: Unification in Monoidal Theories is Solving Linear Equations over Semirings
57 pages

RR-92-02
Andreas Dengel, Rainer Bleisinger, Rainer Hoch, Frank Hönes, Frank Fein, Michael Malburg:
Π_{ODA}: The Paper Interface to ODA
53 pages

RR-92-03
Harold Boley:
Extended Logic-plus-Functional Programming
28 pages

RR-92-04
John Nerbonne: Feature-Based Lexicons: An Example and a Comparison to DATR
15 pages

RR-92-05
Ansgar Bernardi, Christoph Klauck, Ralf Legleitner, Michael Schulte, Rainer Stark:
Feature based Integration of CAD and CAPP
19 pages

RR-92-06
Achim Schupetea: Main Topics of DAI: A Review
38 pages

RR-92-07
Michael Beetz:
Decision-theoretic Transformational Planning
22 pages

RR-92-08
Gabriele Merziger: Approaches to Abductive Reasoning - An Overview -
46 pages

RR-92-09
Winfried Graf, Markus A. Thies:
Perspektiven zur Kombination von automatischem Animationsdesign und planbasierter Hilfe
15 Seiten

RR-92-10
M. Bauer: An Interval-based Temporal Logic in a Multivalued Setting
15 pages

RR-92-11
Susane Biundo, Dietmar Dengler, Jana Koehler:
Deductive Planning and Plan Reuse in a Command Language Environment
13 pages

RR-92-13
Markus A. Thies, Frank Berger:
Planbasierte graphische Hilfe in objektorientierten Benutzungsoberflächen
13 Seiten

RR-92-14
Intelligent User Support in Graphical User Interfaces:
1. InCome: A System to Navigate through Interactions and Plans
   Thomas Fehrle, Markus A. Thies
2. Plan-Based Graphical Help in Object-Oriented User Interfaces
   Markus A. Thies, Frank Berger

22 pages

RR-92-15
Winfried Graf: Constraint-Based Graphical Layout of Multimodal Presentations
23 pages

RR-92-16
Jochen Heinsohn, Daniel Kudenko, Berhard Nebel, Hans-Jürgen Profitlich: An Empirical Analysis of Terminological Representation Systems
38 pages

RR-92-17
Hassan Aït-Kaci, Andreas Podelski, Gert Smolka: A Feature-based Constraint System for Logic Programming with Entailment
23 pages

RR-92-18
John Nerbonne: Constraint-Based Semantics
21 pages

RR-92-19
Ralf Legleitner, Ansgar Bernardi, Christoph Klauck PIM: Planning In Manufacturing using Skeletal Plans and Features
17 pages

RR-92-20
John Nerbonne: Representing Grammar, Meaning and Knowledge
18 pages

RR-92-21
Jörg-Peter Mohren, Jürgen Müller
Representing Spatial Relations (Part II) -The Geometrical Approach
25 pages

RR-92-22
Jörg Würtz: Unifying Cycles
24 pages

RR-92-24
Gabriele Schmidt: Knowledge Acquisition from Text in a Complex Domain
20 pages

RR-92-25
Franz Schmalhofer, Ralf Bergmann, Otto Kühn, Gabriele Schmidt: Using integrated knowledge acquisition to prepare sophisticated expert plans for their re-use in novel situations
12 pages

RR-92-26
Franz Schmalhofer, Thomas Reinartz, Bidjan Tschaitschian: Intelligent documentation as a catalyst for developing cooperative knowledge-based systems
16 pages

RR-92-27
Franz Schmalhofer, Jörg Thoben: The model-based construction of a case-oriented expert system
18 pages

DFKI Technical Memos

TM-91-11
Peter Wazinski: Generating Spatial Descriptions for Cross-modal References
21 pages

TM-91-12
Klaus Becker, Christoph Klauck, Johannes Schwagereit: FEAT-PATR: Eine Erweiterung des D-PATR zur Feature-Erkennung in CAD/CAM
33 Seiten

TM-91-13
Knut Hinkelmann:
Forward Logic Evaluation: Developing a Compiler from a Partially Evaluated Meta Interpreter
16 pages

TM-91-14
Rainer Bleisinger, Rainer Hoch, Andreas Dengel:
ODA-based modeling for document analysis
14 pages

TM-91-15
Stefan Bussmann: Prototypical Concept Formation
An Alternative Approach to Knowledge Representation
28 pages

TM-92-01
Lijuan Zhang:
Entwurf und Implementierung eines Compilers zur Transformation von Werkstückrepräsentationen
34 Seiten

TM-92-02
Achim Schupeta: Organizing Communication and Introspection in a Multi-Agent Blocksworld
32 pages

TM-92-03
Mona Singh
A Cognitiv Analysis of Event Structure
21 pages

TM-92-04
Jürgen Müller, Jörg Müller, Markus Pischel, Ralf Scheidhauer:
On the Representation of Temporal Knowledge
61 pages

TM-92-05
Franz Schmalhofer, Christoph Globig, Jörg Thoben
The refitting of plans by a human expert
10 pages

TM-92-06
Otto Kühn, Franz Schmalhofer: Hierarchical skeletal plan refinement: Task- and inference structures
14 pages

DFKI Documents

D-91-14
Erich Achilles, Bernhard Hollunder, Armin Laux, Jörg-Peter Mohren: KRIS : Knowledge Representation and Inference System
- Benutzerhandbuch -
28 Seiten

D-91-15
Harold Boley, Philipp Hanschke, Martin Harm, Knut Hinkelmann, Thomas Labisch, Manfred Meyer, Jörg Müller, Thomas Oltzen, Michael Sintek, Werner Stein, Frank Steinle:
µCAD2NC: A Declarative Lathe-Worplanning Model Transforming CAD-like Geometries into Abstract NC Programs
100 pages

D-91-16
Jörg Thoben, Franz Schmalhofer, Thomas Reinartz:
Wiederholungs-, Varianten- und Neuplanung bei der Fertigung rotationssymmetrischer Drehteile
134 Seiten

D-91-17
Andreas Becker:
Analyse der Planungsverfahren der KI im Hinblick auf ihre Eignung für die Abeitsplanung
86 Seiten

D-91-18
Thomas Reinartz: Definition von Problemklassen im Maschinenbau als eine Begriffsbildungsaufgabe
107 Seiten

D-91-19
Peter Wazinski: Objektlokalisation in graphischen Darstellungen
110 Seiten

D-92-01
Stefan Bussmann: Simulation Environment for Multi-Agent Worlds - Benutzeranleitung
50 Seiten

D-92-02
Wolfgang Maaß: Constraint-basierte Plazierung in multimodalen Dokumenten am Beispiel des Layout-Managers in WIP
111 Seiten

D-92-03
Wolfgan Maaß, Thomas Schiffmann, Dudung Soetopo, Winfried Graf: LAYLAB: Ein System zur automatischen Plazierung von Text-Bild-Kombinationen in multimodalen Dokumenten
41 Seiten

D-92-06
Hans Werner Höper: Systematik zur Beschreibung von Werkstücken in der Terminologie der Featuresprache
392 Seiten

D-92-07
Susanne Biundo, Franz Schmalhofer (Eds.):
Proceedings of the DFKI Workshop on Planning
65 pages

D-92-08
Jochen Heinsohn, Bernhard Hollunder (Eds.):
DFKI Workshop on Taxonomic Reasoning
Proceedings
56 pages

D-92-09
Gernod P. Laufkötter: Implementierungsmöglichkeiten der integrativen Wissensakquisitionsmethode des ARC-TEC-Projektes
86 Seiten

D-92-10
Jakob Mauss: Ein heuristisch gesteuerter Chart-Parser für attributierte Graph-Grammatiken
87 Seiten

D-92-12
Otto Kühn, Franz Schmalhofer, Gabriele Schmidt:
Integrated Knowledge Acquisition for Lathe Production Planning: a Picture Gallery (Integrierte Wissensakquisition zur Fertigungsplanung für Drehteile: eine Bildergalerie)
27 pages

D-92-13
Holger Peine: An Investigation of the Applicability of Terminological Reasoning to Application-Independent Software-Analysis
55 pages

D-92-15
DFKI Wissenschaftlich-Technischer Jahresbericht 1991
130 Seiten

D-92-16
Judith Engelkamp (Hrsg.)
Verzeichnis von Softwarekomponenten für natürlichsprachliche Systeme
189 Seiten

D-92-21
Anne Schauder: Incremental Syntactic Generation of Natural Language with Tree Adjoining Grammars
57 pages

Verzeichnis von Softwarekomponenten für natürlichsprachliche Systeme

Judith Engelkamp (Hrsg.)

D-92-16
Document