Institut für Informatik
Universität Zürich

2003/2004

Lizentiatsarbeit der philosophischen Fakultät der Universität Zürich

Dr. Norbert E. Fuchs
Prof. Dr. M. Hess

AceLex – Lexikon für Ace

lexicon(adj, [logical_relation([accustomed]), positive([accustomed]), comparative(['more accustomed']), superlative(['most accustomed']), positive_aliases([]), comparative_aliases([]), superlative_aliases([]), complement([prepositional_phrase]), complementing_preposition([to]), comment([])]).
lexicon(adj, [logical_relation([accustomed]), positive([accustomed]), comparative(['more accustomed']), superlative(['most accustomed']), positive_aliases([]), comparative_aliases([]), superlative_aliases([]), complement([no_complement]), complementing_preposition([]), comment([])]).
lexicon(cn, [logical_relation([ace]), singular([ace]), plural([aces]), singular_aliases([]), plural_aliases([]), type([object]), gender([neuter]), collective_noun([no]), group([countable]), comment([])]).
lexicon(cn, [logical_relation(['AceLex']), singular(['AceLex']), plural(['AceLexes']), singular_aliases([['ACE–Lexicon']]), plural_aliases([['ACE–Lexica'], ['ACE–Lexicons']]), type([object]), gender([neuter]), collective_noun([no]), group([countable]), comment(['A Lexicon for Attempto Controlled English'])]).
lexicon(cn, [logical_relation([acerbity]), singular([acerbity]), plural([acerbities]), singular_aliases([]), plural_aliases([]), type([object]), gender([neuter]), collective_noun([no]), group([countable]), comment([])]).
lexicon(cn, [logical_relation([acerbity]), singular([acerbity]), plural([acerbities]), singular_aliases([]), plural_aliases([]), type([object]), gender([neuter]), collective_noun([no]), group([mass]), comment([])]).
lexicon(cn, [logical_relation([acetate]), singular([acetate]), plural([acetates]), singular_aliases([]), plural_aliases([]), type([object]), gender([neuter]), collective_noun([yes]), group([countable]), comment([])]).
lexicon(cn, [logical_relation([acetate]), singular([acetate]), plural([acetates]), singular_aliases([]), plural_aliases([]), type([object]), gender([neuter]), collective_noun([yes]), group([mass]), comment([])]).
lexicon(adj, [logical_relation([acetic]), positive([acetic]), comparative(['more acetic']), superlative(['most acetic']), positive_aliases([]), comparative_aliases([]), superlative_aliases([]), complement([no_complement]), complementing_preposition([]), comment([])]).

Alexandra Bünzli
Im Eichli 18
8162 Steinmaur

044 853 17 40
abuenzli@bluewin.ch

Abgabedatum: 21. November 2004

Inhaltsverzeichnis

1. Einleitung

Die vorliegende Lizentiatsarbeit ist im Rahmen eines Forschungsprojekts der Universität Zürich entstanden. Die Aufgabe besteht darin, ein Lexikon für das ATTEMPTO-Projekt der *Requirement Engineering Research Group* des Instituts für Informatik bereitzustellen. Ziel von ATTEMPTO ist es, eine neue Sprache zu entwickeln, die das Schreiben und Verarbeiten von Spezifikationen erleichtert. Für diese Sprache, die *Attempto Controlled English* (ACE) genannt wird, sollen in dieser Arbeit aus bestehenden elektronischen Lexika die für ACE benötigten Informationen extrahiert und in geeigneter Form für die Verwendung in ATTEMPTO aufbereitet werden.
Die Aufgabe umfasst drei Schwerpunkte:

- Die selbständige Suche nach geeigneten Quelllexika; Wahl eines Lexikons.
- Die logische Extraktion der benötigten Informationen für das neue ACE-Lexikon: Aus welchen Einträgen des Quelllexikons kann welche Information gewonnen werden?
- Der technische Programmierteil: Die Extraktion der Daten aus diesen Lexika und Herstellung des neuen Lexikons für die Sprache ACE – ACELEX.

Vorgegeben für die Arbeit waren lediglich die Informationen, die in ACELEX vorhanden sein müssen und die äusserliche Struktur von ACELEX, eine auf Listen basierte Struktur in der Sprache PROLOG.
Nach einer kurzen Einführung in ATTEMPTO und die Sprache ACE, werden danach die Anforderungen erläutert, die das neue Lexikon erfüllen muss. Ich werde

mich beim Aufbau dieses Hauptteils an den drei oben erwähnten Schwerpunkten der Aufgabenstellung orientieren:

- In Kapitel 3 werden die inhaltlichen Anforderungen an das neue Lexikon ACE-LEX erläutert und danach einige Lexika vorstellt, die als Quelllexika in Frage kommen. In einem Vergleich lege ich dar, welches Lexikon ich als Quelllexikon gewählt habe und begründe die Entscheidung.

- In Kapitel 4 erkläre ich, aus welchen Einträgen im Quelllexikon die für ACE-LEX relevanten Informationen herausgefiltert werden können. Dabei gehe ich bewusst auf Details ein, da dieses Kapitel als Referenz dienen soll. Die Logik der Informationsextraktion kann dort nachvollzogen werden.

- In Kapitel 5 wird der Programmierteil vorgestellt. Die einzelnen Komponenten der Applikation werden nach und nach eingeführt und erklärt. Auf Details der Programmierung wird nicht eingegangen, dafür verweise ich direkt auf den *Code* auf der beiliegenden CD-ROM im Anhang A. In Kapitel 6 werden die verwendeten Sprachen und Formate kurz vorgestellt, in denen die Applikation implementiert ist.

Die Arbeit wird durch eine Zusammenfassung abgerundet, in der auch auf weiterführende Arbeiten hingewiesen wird.
Linguistische Begriffe, die dem Leser der Arbeit unbekannt sein könnten, sind in einem Glossar erklärt. Ich werde im Text nicht auf Definitionen im Glossar verweisen, sondern bitte den Leser, sich bei unbekannten Begriffen selbständig im Glossar zu informieren.
Im Anhang B findet sich eine Austellung über die Fehler und Eigenheiten, die in der lexikalischen Quelle während der Arbeit an diesem Projekt entdeckt worden sind.

2. Attempto

2.1. Was ist Attempto?

Attempto ist ein Forschungsprojekt der *Requirements Engineering Research Group* des Instituts für Informatik an der Universität Zürich[1].
Ziel von Attempto ist es, eine neue Spezifikations- und Wissensrepräsentationssprache zur Verfügung zu stellen, die auf einer kontrollierten Version der natürlichen Sprache basiert. Der traditionelle Ansatz des Schreibens in vollständiger natürlicher Sprache bringt gewisse Schwierigkeiten mit sich. Zwar lässt sich mit der natürlichen Sprache alles beschreiben, jedoch führt ihr unkontrollierter Gebrauch zu Ambiguität und unklaren Aussagen - bei Spezifikationen kann dies unter Umständen falsche Entscheide bei der Entwicklung des Produkts nach sich ziehen. Trotzdem sind auch heute noch die meisten Anforderungsspezifikationen in natürlicher Sprache geschrieben, denn die Vorteile dieses Ansatzes liegen auf der Hand: Es muss keine neue Sprache gelernt werden und die natürliche Sprache ist auch für andere menschliche Benutzer verständlich, ohne dass weitere Erklärungen nötig wären.
Ein anderer Ansatz versucht, mit eigens für das Schreiben von Spezifikationen entwickelten formalen Sprachen das Problem der Ambiguität zu lösen. Formale Sprachen haben den Vorteil, eine eindeutige Syntax und klare Semantik zu haben. Die oftmals logisch basierten Sprachen unterstützen die automatische Analyse der Spezifikatio-

[1] In diesem Kapitel folge ich weitgehend den Ausführungen auf der Attempto-Homepage (`www.ifi.unizh.ch/attempto/home/index.html`) und [Fuchs et al. 1999]

nen, es kann z. B. ihre Gültigkeit und Konsistenz überprüft werden. Die Nachteile dieses Ansatzes sind v.a. der anfängliche Lernaufwand und die Tatsache, dass formale Sprachen oftmals schwer zu verstehen sind. Das intuitive Verständnis und die Verbindung zur jeweiligen Domäne fehlt, was das Schreiben der Spezifikationen erschwert.

Die Sprache *Attempto Controlled English* (ACE) soll die Vorteile der beiden Ansätze - dem Schreiben der Spezifikation in vollständiger natürlicher Sprache und dem in formaler Sprache - vereinigen. Spezifikationen sollen in vertrauter natürlicher Sprache geschrieben werden können und trotzdem die automatische Verarbeitung erlauben. ACE ist eine Untermenge der englischen Sprache, die durch grammatikalische Einschränkungen und domänenspezifisches Vokabular gekennzeichnet ist. D.h. alle ACE-Sätze sind grammatisch korrekte englische Sätze, aber nicht alle möglichen korrekten englischen Sätze sind in ACE erlaubt. Die Einschränkungen machen es erst möglich, dass die in der natürlichen Sprache vorkommenden Ambiguitäten und die Unbestimmtheiten so reduziert werden, dass ACE für die Beschreibung von Spezifikationen geeignet ist. Im nachfolgenden Abschnitt wird auf die in natürlicher Sprache auftretenden Ambiguitäten eingegangen.

2.2. Ambiguität

Es gibt verschiedene Arten von Ambiguität. [HESS 2004a, S. 182ff.] unterscheidet folgende Typen von Ambiguitäten: Die lexikalische, die syntaktische und die logische Ambiguität. Die lexikalische Ambiguität wird auch *Homonymie*[2] genannt. Ein Wort hat zwei verschiedene Bedeutungen. Ein Beispiel hierfür ist das Wort „Tau“, das einmal ein dickes Seil und einmal das über Nacht an den Gegenständen kondensierte

[2]Manchmal wird auch noch zwischen *Homonymie* und *Polysemie* unterschieden. Homonymie bezeichnet Worte, die aus zwei historisch verschiedenen Bedeutungswurzeln entstanden sind, deren äusserliche Form zufällig gleich wurde. Polyseme Worte hingegen stammen historisch vom gleichen Begriff ab, haben sich aber im Laufe der Zeit auseinander entwickelt, bis sie als eigenständige Bedeutungen galten.

Wasser der Luftfeuchtigkeit bezeichnet.
Die syntaktische Ambiguität wird weiter unterteilt in morphologische, kategoriale und strukturelle Ambiguität. Nachfolgend ein Beispiel für die strukturelle Ambiguität, Details und Beispiele für andere Ambiguitäten sind in [Hess 2004a, S. 182-185] nachzulesen.

Ich sah den Mann im Park mit dem Teleskop

Dieser Satz hat mindestens vier Lesarten:

1. Ich sah den Mann, der im Park mit dem Teleskop war.
2. Ich sah den Mann, der im Park war, mit Hilfe des Teleskops.
3. Ich sah den Mann, als ich im Park mit dem Teleskop war.
4. Ich sah den Mann, als ich im Park war, mit Hilfe des Teleskops.

Die Ambiguität ensteht, da nicht klar ist, an welches Satzteil die beiden Präpositionalphrasen „im Park“ und „mit dem Teleskop“ angeschlossen werden sollen. Es gibt mehrere Möglichkeiten, diesen Anschluss zu realisieren. Wird mit „im Park“ der Ich-Erzähler oder der Mann beschrieben? Bezieht sich „mit dem Teleskop“ auf das Verb „sah“, den Ich-Erzähler oder den Mann[3]?
Als „logische“ Ambiguität bezeichnet [Hess 2004a, S. 186ff.] jene Ambiguitäten, die sich nicht an oberflächlichen Unterschieden der Syntaxstrukturen ausdrücken und nicht lexikalisch bedingt sind. Ein berühmtes Beispiel hierfür ist der Satz

Jeder Mann liebt eine Frau

Der Satz hat folgende Lesarten:

[3]Diese Ambiguität wird in Ace aufgelöst. Interpretationsregeln (vgl. Abschnitt 2.4.3) legen genau fest, welche Satzteile durch welche Strukturen modifiziert werden. Eine Präpositionalphrase modifiziert in Ace z.B. immer das Verb. Ist diese Lesart nicht die gewünschte, muss eine Umformulierung vorgenommen werden.

1. Für jeden Mann gibt es (mindestens) eine (möglicherweise andere) Frau, die er liebt.

2. Es gibt eine (und zwar ein und dieselbe) Frau, welche von allem Männern geliebt wird[4].

Diese und andere Ambiguitäten werden durch die Restriktionen in ACE verhindert. Nebst der Reduzierung der Ambiguität unterstützen die Restriktionen in ACE auch den Ersteller der Spezifikationen beim Schreiben, da er gezwungen ist, präzise und klare Formulierungen zu benutzen. So können die entstehenden Spezifikationen mit dem Computer verarbeitet und ohne Ambiguitäten in formale Spezifikationssprachen, speziell in Logik erster Stufe[5], übersetzt werden.

2.3. Vorgehen des Attempto-Systems

Das ATTEMPTO-*System* wurde entwickelt, um in ACE geschriebene Spezifikationstexte in sogenannte *discourse representation structures (DRS)*[6] zu übersetzen. Auf der Grundlage dieser DRS kann die Spezifikation validiert und getestet werden. DRS sind eine syntaktische Variante der Prädikatenlogik erster Ordnung[7]. Die ATTEMPTO *Parsing Engine* (APE) übersetzt die ACE-Spezifikation in die DRS und die Ergebnisse der Übersetzung werden in eine Datenbank aufgenommen. Mit Hilfe des ATTEMPTO-*Reasoner* (RACE) kann die Gültigkeit der Spezifikation und ihre Konsistenz geprüft werden. Zusätzlich kann RACE Anfragen in ACE über die Spezifikation beantworten. Die Datenbank kann ausserdem für Simulation und Prototyping ver-

[4] Auch diese Ambiguität wird in ACE durch die Anwendung von Interpretationsregeln verhindert (vgl. Abschnitt 2.4.3).

[5] Ein kurzer Überblick über die Logik und Literaturhinweise zu diesem Thema finden sich in [HESS 2004b].

[6] DRS sind Teil der *discourse representation theory extended by events and states (DRT-E)* siehe: H. Kamp, U. Reyle: From Discourse to Logic, Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory, Studies in Linguistics and Philosophy 42, Kluwer, 1993

[7] siehe Fussnote 5

wendet werden. Die Abbildung 2.1 illustriert das Vorgehen des ATTEMPTO-*Systems*.

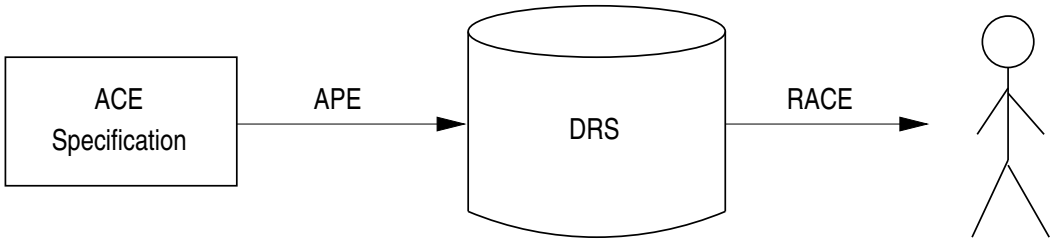

Abbildung 2.1.: Vorgehen des ATTEMPTO-*Systems*

Das kleine Beispiel 2.1[8] zeigt eine ACE-Spezifikation für einen simplen automatischen Geldautomaten (*simple automated teller*), der SimpleMat (SM) genannt wird. Die

> The customer enters a card and a numeric personal code. If it is not valid then SM rejects the card.

Beispiel 2.1: Eine kurze ACE-Spezifikation

Interpretation durch das ATTEMPTO-System liefert das in Beispiel 2.2 vorgestellte Ergebnis.

> The customer enters a card and [enters] a numeric personal code. If [the numeric personal code] is not valid then [SimpleMat] rejects [the card].

Beispiel 2.2: Paraphrase einer ACE-Spezifikation

Was hat das ATTEMPTO-*System* gemacht? Das weggelassene Verb „enters“ wurde eingefügt, das anaphorische „it“[9] wurde durch die Nominalphrase „the numeric personal code“ ersetzt und die Abkürzung „SM“ durch „SimpleMat“. Ausserdem wurde erkannt, dass die Nominalphrase „the card“ sich auf die im vorherigen Satz

[8] Das Beispiel findet sich auf `www.ifi.unizh.ch/attempto/description/index.html`
[9] vgl. Abschnitt 2.4.2.4

stehende Nominalphrase „a card“ bezieht. Diese Änderungen oder Interpretationen sind durch die eckigen Klammern gekennzeichnet.
Die Übersetzung der Spezifikation durch die ATTEMPTO *Parsing Engine* (APE) in die *discourse representation structure* (DRS) ist im Beispiel 2.3 zu sehen[10].

```
[A, B, C, D, E, F]
customer(A)
card(B)
event(C,enter(A,B))
numeric(D)
personal_code(D)
event(E,enter(A,D))
named(F,simplemat)
IF:
   [ ]
   NOT:
        [G]
           state(G,valid(D))
THEN:
   [H]
   event(H,reject(F,B))
```

Beispiel 2.3: DRS

A, B, ..., H in der Struktur sind *discourse referents* - existentiell quantifizierte Variablen[11] - die für die Objekte des *discourse* – der betrachteten „Mini-Welt“ – stehen. Der Rest der DRS stellt die Konditionen für diese *discourse referents* dar.
Der Benutzer kann nun mit Hilfe von RACE Fragen an das System stellen:

Who enters a card?

RACE beantwortet die Frage durch Schlussfolgerungen aufgrund der Spezifikation:

[10] Das Beispiel ist veraltet. Die aktuelle Version von ATTEMPTO arbeitet mit erweiterten *discourse representation structures*. Um das Beispiel möglichst verständlich zu halten, wird in dieser Arbeit jedoch die alte Version vorgestellt. Weitere Informationen sind auf der ATTEMPTO-Homepage `www.ifi.unizh.ch/attempto/home/index.html` zu finden und speziell in [FUCHS et al. 2004].
[11] [HESS 2004b, S. 74ff.]

[The customer] enters a card.

In ähnlicher Weise kann die Spezifikation „ausgeführt“ werden, d.h. die logischen Konsequenzen der Spezifikation werden – auf der Ebene von ACE – sichtbar gemacht. So können z.B. Hypothesen getestet werden: Durch logische Schlussfolgerungen kann überprüft werden, was passiert, wenn der *numeric personal code* nicht gültig ist. RACE antwortet, indem er die logischen Konsequenzen aufzählt:

SimpleMat rejects the card.

Es kann auch überprüft werden, welche Konditionen zu einem bestimmten Ereignis oder Zustand führen. RACE kann mit einer Aufzählung der möglichen Ursachen beantworten, was nach Spezifikation der Grund dafür sein kann, dass SimpleMat eine Karte zurückweist:

The numeric personal code is not valid.

Es ist ausserdem möglich, die logische und temporale Struktur der Spezifikation Schritt für Schritt sichtbar zu machen. So kann der Spezifikationstext

The customer enters a card and a numeric personal code.

durch folgende Einzelschritte dargestellt werden:

event: A enters B
A customer
B card
event: A enters D
A customer
D personal code

Auf diese Weise kann die logische und temporale Struktur der Spezifikation vom Benutzer einfach nachvollzogen und überprüft werden, da das domänenspezifische Vokabular die Orientierung erleichtert.

2.4. Ace

Um die Spezifikationen verarbeiten zu können, müssen sie in der Sprache Ace geschrieben sein. Ace besteht im Wesentlichen aus folgenden drei Komponenten: dem Vokabular, den Konstruktionsregeln und den Interpretationsregeln. Diese Regeln und Restriktionen müssen gelernt werden, sind jedoch einfach anzuwenden. In den nächsten Abschnitten stelle ich diese drei Komponenten kurz vor.

2.4.1. Vokabular

Das Vokabular umfasst folgende Wortgruppen:

Funktionswörter Funktionswörter wie Artikel („a“, „the“), Konjunktionen („and“, „or“) und Präpositionen („in“, „to“) sind vordefiniert.

Inhaltswörter Inhaltswörter sind benutzerdefinierte, domänenspezifische Wörter wie Verben („enter“), Nomen („train“), Adjektive („red“) und Adverben („immediately“). Der Benutzer kann diese Wörter mit Hilfe eines lexikalischen Editors definieren. Der lexikalische Editor ist in Prolog geschrieben. Er ist textbasiert, dem Benutzer werden Fragen gestellt, die entweder mit einer Auswahl oder durch eine individuelle Tastatureingabe beantwortet werden. Als erstes muss sich der Benutzer entscheiden, was er tun will: Einen neuen Eintrag einfügen, einen vorhandenen Eintrag editieren, einen Eintrag löschen oder das Programm verlassen. In der Abbildung 2.2 ist ein *Screenshot* des Editors zu sehen. Der Benutzer muss keine fortgeschrittenen linguistischen Kenntnisse besitzen, um Wörter mit dem Editor einzufügen, es genügen grammatikalische Grundkenntnisse. Auf diese Weise kann er genau diese Wörter einfügen, die er braucht. Ein vollständiges Lexikon auf diese Weise zu erstellen, ist nicht praktikabel: Das in dieser Arbeit generierte Lexikon soll als Basislexikon dienen und mit Attempto mitgeliefert werden.

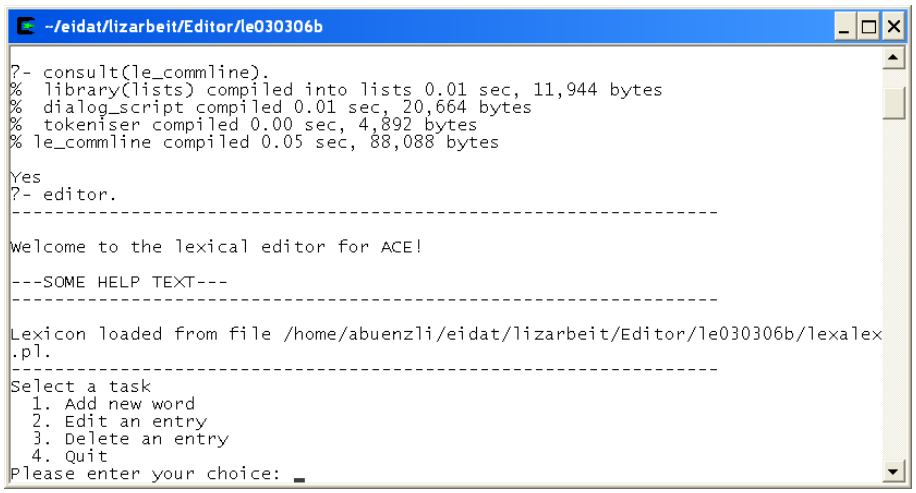

Abbildung 2.2.: Der lexikalische Editor

2.4.2. Konstruktionsregeln

Die Konstruktionsregeln definieren die Form der ACE-Sätze und -Texte. Diese Regeln definieren Restriktionen auf der Wort- und Satzebene. Dadurch sollen Ungenauigkeiten der natürlichen Sprache von Anfang an verhindert werden.
Eine Spezifikation in ACE besteht aus Sätzen. Es gibt *simple sentences*, *composite sentences* und *query sentences*. In den folgenden Abschnitten werde ich zuerst einen Überblick über diese drei Satztypen geben und darlegen, wie sie aufgebaut werden dürfen. Danach gehe ich auf die Einschränkungen ein, denen diese Sätze zusätzlich unterliegen.

2.4.2.1. Simple Sentences

Simple sentences haben die Form

subject + verb + complements + adjuncts

Dabei sind die Komplemente (*complements*) nötig für transitive und ditransitive Verben. Ihr Vorhandensein und ihre Anzahl hängen von den entsprechenden Verben ab: Sie werden von ihnen verlangt. Adjunkte (*adjuncts*) hingegen sind optional.

Oftmals handelt es sich dabei um Angaben über Ort, Zeit und Art. Das Beispiel 2.4 zeigt einen *simple sentence*.

$$\underbrace{\text{The driver}}_{subject} \quad \underbrace{\text{stops}}_{verb} \quad \underbrace{\text{the train}}_{complement} \quad \underbrace{\text{at the station}}_{adjunct}$$

Beispiel 2.4: Simple Sentence aus ACE

2.4.2.2. Composite Sentences

Composite sentences werden aus *simple sentences* gebildet. Dazu werden folgende Konstruktionsmuster zur Verfügung gestellt:

- Koordination mit *and* und *or*
  Bsp: „A passenger presses an alarm signal button and the driver stops the train.“ (vgl. Abschnitt 2.4.2.5)
- Subordination durch Konditionalsätze mit *if...then...*
  Bsp: „If a passenger presses the alarm signal button then the driver stops the train“
- Subordination durch Relativsätze, die Subjekt oder Objekt modifizieren (*who, which, that*)
  „The driver stops the train that has a defect brake.“
- Negation der Verbalphrase (*does not, is not*)
  Bsp: „The train does not stop.“
- Negation der Nominalphrase (*no*)
  Bsp: „No train stops at this station.“
- Quantifikation (Quantoren: *a, there is a, every, for every*; vgl. Abschnitt 2.4.3, Seite 22)

Bsp: „Every train has a driver.“ Dieser Satz hat zwei textuelle Quantoren: „every“ und „a“. Der Satz ist in vollständiger natürlicher Sprache logisch ambig (vgl. Abschnitt 2.2, Seite 9) und hat folgende zwei Lesarten:

> Alle Züge haben einen (von den anderen eventuell verschiedenen) Fahrer[12].

oder

> Es gibt genau einen Fahrer, für den gilt, dass alle Züge ihn als Fahrer haben[13].

In Ace wird aufgrund der in den Interpretationsregeln vorgegebene Schachtelung der Quantoren (vgl. Abschnitt 2.4.3, Seite 22) die zweite Lesart verhindert.

2.4.2.3. Query Sentences

In Ace sind *yes/no*-Fragen und *wh*-Fragen erlaubt.

- *yes/no*-Fragen werden durch Invertierung von Subjekt und Verb aus *simple sentences* gebildet. D.h. Verb und Subjekt tauschen die Plätze: Ist das Verb des umzustellenden Satzes „be“, wird es dem Subjekt vorangestellt („Is the train in the station?“), ist es ein anderes Verb, wird vor das Subjekt „do“ oder „does“ eingefügt („Does the driver stop the train?“).
- *wh*-Fragen beginnen mit einem sogenannten *wh*-Wort: „who“, „what“, „when“, „where“, „how“[14] etc. und werden mit „do“ oder „does“ gebildet („Where does the driver stop?“). Wird jedoch nach dem Subjekt des Satzes gefragt, entfällt die Einfügung von „do“ bzw. „does“ („Who stops the train?“).

[12] Dargestellt in der Prädikatenlogik erster Stufe: $\forall$ T: train(T) $\rightarrow \exists$ D: driver(D) $\wedge$ has(T,D)
[13] Dargestellt in der Prädikatenlogik erster Stufe: $\exists$ D: driver(D) $\wedge \forall$ T: train(T) $\rightarrow$ has(T,D)
[14] „how“ wird ebenfalls zu den *wh*-Fragen gezählt

2.4.2.4. Anaphora

Eine Anapher oder auch Anaphora ist eine sprachliche Einheit, die zu einer anderen sprachlichen Einheit im vorangehenden Text, dem Antezedens, in einer sogenannten anaphorischen Beziehung steht: D.h. man kann das Referenzobjekt der Anapher nur durch die Beziehung auf das vorangestellte Antezedens bestimmen. In ACE sind Personalpronomen und Nominalphrasen mit bestimmtem Artikel als Anaphora zugelassen. Zur Illustration hierzu das Beispiel 2.5. In diesem Beispiel hat es zwei

A passenger of a train alerts the driver. He stops the train.

Beispiel 2.5: Anaphora in ACE

Anaphora. Die eine ist das Personalpronomen „he“, das sich auf die vorangehende Nominalphrase „the driver“ bezieht. Das Personalpronomen „he“ kann sich in einem korrekten ACE-Text nur auf „the driver“ beziehen, da sich die Anapher gemäss den Interpretationsregeln in Abschnitt 2.4.3 auf die am nächsten links stehende Nominalphrase beziehen muss. Die andere Anapher ist die Nominalphrase „the train“ mit dem bestimmten Artikel, die die vorangehende Nominalphrase „a train“ mit unbestimmten Artikel referenziert.

2.4.2.5. Koordination

Wie in Abschnitt 2.4.2.2 erwähnt, ist in ACE die Koordination mit „and“ und „or“ erlaubt. In diesem Abschnitt werden die verschiedenen Arten der Koordination ausgeführt, die in ACE zugelassen sind. In ACE ist Koordination zwischen Sätzen und zwischen Phrasen des gleichen syntaktischen Typs möglich. Die Sätze in Beispiel 2.6 sollen dies verdeutlichen. Im ersten Beispielsatz werden zwei Sätze mit „and“ zusammengehängt. Der zweite Beispielsatz zeigt die Verknüpfung von zwei Verbalphrasen, im dritten werden zwei Adjunkte koordiniert. Falls zwei Verbalphrasen koordiniert werden, die mit dem gleichen Verb gebildet werden („A passenger presses a red but-

1. A passenger presses an alarm signal button and the driver stops the train.
2. A passenger presses an alarm signal button and alerts the driver.
3. A driver stops a train immediately or at the next station.

Beispiel 2.6: Koordination in ACE

ton or presses a green button.“) kann das Verb im zweiten Teil weggelassen werden („A passenger presses a red button or a green button.“).

2.4.2.6. Lexikalische Einschränkungen

Auch das Vokabular ist gewissen Einschränkungen unterworfen. In der folgenden Aufzählung sind einige dieser lexikalischen Restriktionen aufgeführt:

- Verben sind nur in der *simple present tense* (Gegenwart) in aktiver Verwendung im Indikativ zugelassen. Ausserdem erlaubt ACE nur die Benutzung der dritten Person Singular und Plural („he presses“, „they press“).
- Es werden keine modalen Verben („may“, „can“, „must“ etc.) oder intensionale Verben („hope“, „know“, „believe“ etc.) zugelassen.
- Ebenfalls nicht erlaubt sind modale Adverben („possibly“, „probably“).

ACE erlaubt dem Benutzer Abkürzungen („ASB“ für „alarm signal button“ oder alternative Wörter („alarm button“ statt „alarm signal button“) zu definieren, die als Synonyme verwendet werden sollen. Dazu kann der Benutzer den bereits in Abschnitt 2.4.1 erwähnten lexikalischen Editor benutzen. Wie man bei „alarm signal button“ sieht, sind in ACE zusammengestzte Inhaltswörter erlaubt. Ebenfalls möglich wäre die Form „alarm-signal-button“. Natürlich sind auch einfache, d.h. nicht zusammengesetzte Inhaltswörter wie z.B. „train“ erlaubt.

2.4.2.7. Phrasale Einschränkungen

Auch auf der Phrasenebene sind Einschränkungen zu beachten. Nachfolgend einige Beispiele zur Veranschaulichung:

- Es werden nur Nominalphrasen und Präpositionalphrasen als Komplemente für Verben zugelassen. Sätze wie „He reads the sign“ (Nominalphrase als Komplement) oder „He drives from one station to the other station“ (Präpositionalphrase als Komplement) sind erlaubt. Hingegen sind keine Sätze wie „He reads that the train stops at every station“ erlaubt, da keine Satz-Komplemente zugelassen sind.

- Adjunkte können nur in der Form von Präpositionalphrasen („at the station“) und als Adverben („immediately“) realisiert werden.

- Nur *of*-Konstruktionen („the part of the train“) sind als postnominale Erweiterung [15] mit einer Präposition zugelassen.

2.4.3. Interpretationsregeln

Die Interpretationsregeln kontrollieren die semantische Analyse grammatikalisch[16] korrekter ACE-Sätze. Z.B. werden durch diese Regeln Ambiguitäten, die durch die Konstruktionsregeln (vgl. Abschnitt 2.4.2) nicht verhindert werden konnten, aufgelöst. Das Resultat der Analyse durch die Interpretationsregeln kommt in einer Paraphrase zum Ausdruck, d.h. der Spezifikationstext wird wie in Beispiel 2.2 mit den vom System vorgenommenen Änderungen, die er aufgrund der Interpretationsregeln erfahren hat, in der Sprache ACE ausgegeben.
Einige wichtige Interpretationsregeln sind:

[15] Die postnominale Erweiterung „of the train“ steht rechts vom Kopf („part“) der von ihm modifizierten Nominalphrase „the part“.
[16] D.h. die Sätze erfüllen die ACE-Anforderungen bezüglich Vokabular und Konstruktionsregeln.

- Verben bezeichnen entweder Zustände (*states*) oder Ereignisse (*events*). Zustände dauern eine gewisse Zeitspanne an, bis sie explizit beendet werden, Ereignisse finden im Moment statt und sind dann vorbei. Ein Beispiel für ein Zustandsverb ist „be“, während „arrive“ ein Ereignisverb ist. Die Einteilung ist nicht trivial. Bei vielen Verben fällt es schwer, sie eindeutig einer dieser beiden Kategorien zuzuordnen. Besonders schwierig ist die Einteilung von Prozessen, wie z.B. „run“, oder „drive“. Einerseits dauern sie eine gewisse Zeit an und haben keinen natürlichen Endpunkt, können also als Zustände angesehen werden. Andererseits ist ein Prozess wie „run“ oder „drive“ etwas, das getan wird, eine Eigenschaft, die den Ereignissen zugesprochen wird[17]. Jedes Verb jedoch muss eindeutig einer Kategorie zugeteilt sein. Im Lexikon ist diese Einteilung festgehalten.

- Die Reihenfolge des Auftretens der Verben im Text entspricht der zeitlichen Abfolge der damit bezeichneten Ereignisse oder Zustände. In den Sätzen

  > A passenger alerts a driver. The driver stops a train. The train is in a station.

  ist das erste Ereignis das Alarmieren des Fahrers, zeitlich gefolgt vom Ereignis des Stoppen des Zuges, worauf der Zustand beginnt, dass sich der Zug nun in der Station befindet.

- Präpositionalphrasen in Adjunkt-Positionen modifizieren immer das Verb. Im Satz „The driver stops the train in a station“ modifiziert das Adjunkt „in a station“ das Ereignis „stops the train“, indem es den Ort des Ereignisses angibt.

- Anaphorische Beziehungen sind möglich durch Personalpronomen oder Nominalphrasen mit bestimmten Artikeln (vgl. Abschnitt 2.4.2.4). Das Referenzob-

[17] Diskussion über dieses Problem in [Parsons 1994]

jekt (Antezedens) muss die am nächsten links der Anapher stehende Nominalphrase sein, die in Numerus und Genus übereinstimmt. Damit wird ein Fall der strukturellen Ambiguität verhindert: Im Satz aus [Hess 2004b, S. 185]

> „John drank the wine on the table and it was good."

ist ohne Restriktionen nicht klar, worauf sich das Personalpronomen „it" bezieht. Möglich sind (theoretisch): „the wine" oder „the table". Für Menschen ist die richtige Auflösung („the wine") kein Problem, für ein Computer-Programm jedoch schon. Durch die Regel, dass die Anapher immer auf das am nächsten mögliche Antezedens verweist, lassen sich solche Fälle lösen. Wäre der Beispielsatz ein Ace-Satz, würde die falsche Beziehung angenommen: „the table was good". Der Satz müsste umformuliert werden.

- Die Koordination von Nominalphrasen innerhalb von Verbalphrasen wird gleich interpretiert wie eine Koordination von Verbalphrasen. Das ausgelassene Verb wird beiden Nominalphrasen vorangestellt („A passenger presses a red button and [presses] a green button", vgl. 2.4.2.5).

- Tritt ein Quantor (*a, there is a, every, for every*; vgl. Abschnitt 2.4.2.2) im Text auf, öffnet er einen Skopus (Gültigkeitsbereich), der bis zum Ende des Satzes gilt. Jeder danach im Satz auftretende Quantor ist dann innerhalb des Skopus des vorangehenden Quantors. Im bereits in Abschnitt 2.4.2.2, Seite 17 vorgestellten Beispiel wird durch diese Regel verhindert, dass der Satz ambig ist:

  > Every train has a driver.

  Die folgenden zwei Lesarten wären für diesen Satz eigentlich möglich:

  1. Alle Züge haben einen (von den anderen eventuell verschiedenen) Fahrer.
     $\forall$ T: train(T) $\rightarrow \exists$ D: driver(D) $\wedge$ has(T,D).

2. Es gibt genau einen Fahrer, für den gilt, dass alle Züge ihn als Fahrer haben.
   $\exists$ D: driver(D) $\wedge$ $\forall$ T: train(T) $\rightarrow$ has(T,D).

Nachdem durch das Auftreten von „every“ ein Skopus geöffnet wurde, muss der danach durch „a“ geöffnete Skopus innerhalb des ersteren liegen. Das ist in der ersten Lesart der Fall. In der zweiten Lesart würde jedoch der Skopus von „every“ innerhalb des Skopus von „a“ liegen. Die zweite Lesart wird in Ace also nicht zugelassen.

3. Lexical Sources

3.1. Inhaltliche Anforderungen

Um entscheiden zu können, welches Lexikon als Basis für AceLex in Frage kommt, muss zuerst aufgezeigt werden, welche Informationen AceLex für Ace zur Verfügung stellen muss. Ich werde die Anforderungen nach Wortart gegliedert vorstellen und zuerst die Nomen, dann die Verben, danach die Adjektive und schliesslich die Adverben behandeln. Da es sich bei Ace um eine kontrollierte Version der englischen Sprache handelt, werde ich bei Bezeichnungen, die keine geeignete Übersetzung im Deutschen besitzen, die englischen Begriffe der Grammatik verwenden. Die Gross- bzw. Kleinschreibung dieser Worte richtet sich nach den Regeln im Englischen, d.h. sie werden nur am Satzanfang grossgeschrieben.

3.1.1. Nomen

Für Nomen müssen folgende Informationen in AceLex vorhanden sein:

- Singular und Plural-Formen, wobei für unzählbare Nomen (*mass nouns*) keine Pluralform existiert.
- Grammatisches Geschlecht (*masculine, feminine, neuter*); im Englischen ist dies nur bei Personen zu beachten.
- Zählbarkeits-Domäne: Es soll eine Antwort auf die Frage geliefert werden, ob das Wort ein zählbares Nomen(*countable noun*), ein unzählbares Nomen

(*mass noun*) oder ein Nomen ist, das auf beide Arten verwendet werden kann (*countable-mass noun*) Ein *countable noun* steht immer mit einem Artikel, sei es der bestimmte oder unbestimmte (*the* bzw. *a, an*). Ein *mass noun* jedoch kann nie mit einem unbestimmten Artikel stehen, sondern nur mit einem bestimmten oder gar keinem Artikel. Nomen, die auf beide Arten verwendet werden können (z.B. „fish“ - „a fish swam downstream.“ - *countable* / „fish can be cooked in many ways.“ - *mass*) sollen zwei Einträge im Lexikon erhalten, einmal als *countable noun* und einmal als *mass noun*.

- Objekt Typ: Zeigt an, ob das Nomen eine Person, Zeit oder ein Objekt repräsentiert.

- *Collective noun*: Es soll bestimmt werden, ob das Nomen ein *collective noun* ist, d.h. ob es in seiner Singularform als Subjekt von im Singular stehenden Verben, wie auch als Subjekt von im Plural stehenden Verben fungieren kann.

  Collective nouns[1] sind singuläre Nomen, die sich auf eine Gruppe von Einheiten beziehen, die entweder als einzelne Entitäten oder als grössere Einheiten angesehen werden können: „a minority is in favor of the action“, „A minority are in favor of the action“. Meistens beziehen sich die *collective nouns* auf eine Gruppe von Lebewesen[2].

[1] Die folgenden Informationen und Beispiele sind hauptsächlich vom *LearnEnglish Archive* des *British Council*: `www.learnenglish.org.uk/grammar/archive/collective_nouns.html`

[2] Im Gebrauch von *collective nouns* sind einige Unterschiede im Amerikanischen und Britischen Englisch zu beachten:

- Im Amerikanischen Englisch nimmt das *collective noun* ein Verb im Singular, wenn es sich auf die Gruppe als Ganzes bezieht: „The family was united on the question.“ Wenn es sich hingegen auf die einzelnen Entitäten der Gruppe bezieht, sollte ein Verb im Plural benutzt werden: „My family are always fighting among themselves“.
- Im Britischen Englisch ist der Gebrauch eines Verbs im Plural weiter verbreitet als im Amerikanischen Englisch. Es gibt keine expliziten Regeln, wann ein Verb im Plural und wann eines im Singular eingesetzt werden soll. Singuläre Verben sind jedoch häufiger, wenn von einer unpersönlichen Einheit gesprochen wird, wie z.B. in „The average British family has 3.6 members. It is smaller and richer than 50 years ago“ im Gegensatz zur persönlicheren Aussage „My family have decided to move to Nottingham. They think it's a better place to live“.

Die Nomen sollen in drei Kategorien unterteilt werden: Die normalen Nomen (*common nouns*), die Eigennamen (*proper nouns*) und Nomen, die Einheiten ausdrücken (*measurement nouns*), wie z.B. „kilogramm“, „watt“ etc.. Um diese *measurement nouns* bestimmen zu können, muss das Lexikon Masseinheiten (*standard units*) enthalten und diese auch als Einheiten kennzeichnen. Falls möglich, soll die dazugehörige Dimension (*standard dimension*), d.h. das Mass, welches die jeweilige Einheit beschreibt, angegeben werden. Ein Beispiel: „kilogramm“ ist eine Einheit der Dimension „Gewicht“ (*weight*).

3.1.2. Verben

Für die Einträge von Verben verlangt ATTEMPTO folgende Informationen:

- 3. Person Singular und Plural: In ACE wird eine kontrollierte Sprache verwendet, die nur Aussagen in der 3. Person im Präsens (*simple present tense*) zulässt.

- *phrasal verbs*: Verbale Partikel von *phrasal verbs* (zusammengesetzten Verben, wie z.B. „look up“) sollen erfasst werden. Dabei darf keine Durchmischung mit Präpositionen von Präpositionalphrasen-Komplementen geschehen, d.h. es muss unterschieden werden zwischen verbalen Partikeln und Präpositionen. In der traditionellen Grammatik werden *phrasal verbs* (Verben mit verbalen Partikeln) und *prepositional verbs* (Verben, die immer mit bestimmter Präposition stehen) unterschieden. Eine Präposition muss immer vor der Nominalphrase stehen, auf die sie Bezug nimmt: „He looks after his mother“, unmöglich jedoch ist „*He looks his mother after“[3]. Ein verbaler Partikel jedoch gehört zum Verb dazu, ist also nicht als eigenständiges Wort anzusehen. Der verbale Partikel kann sowohl vor der Nominalphrase, wie auch nach der Nominalphrase stehen: „He looked up the entry in the lexicon“ oder „He looked the entry

[3]Der * bedeutet, dass der Satz ungrammatisch ist

up in the lexicon“. Ist die Nominalphrase jedoch ein Pronominalobjekt, muss der Partikel hinter dem Objekt stehen: „He looked it up“, „*He looked up it“. Die Unterscheidung kann mit diesem Umstellungstest etwas vereinfacht werden, bleibt aber für Nicht-Muttersprachler schwer, da sie sich nicht völlig auf ihr Sprachgefühl verlassen können[4]. Heutzutage werden „echte“ *phrasal verbs* („look something up“) und *prepositional verbs* („look after somebody“) normalerweise beide zu den *phrasal verbs* gezählt. Sie müssen dann mit *separable* oder *inseparable* unterschieden werden, um die bei der Zusammenlegung verlorengegangene Eigenschaft, ob der Partikel auch hinter der Nominalphrase stehen kann, zu kompensieren. In AceLex soll die Einteilung nach dem traditionellen Ansatz vorgenommen werden, d.h. es wird unterschieden zwischen „echten“ *phrasal verbs* mit einem Partikel, der vor oder hinter der Nominalphrase stehen kann und Verben, die einfach immer eine Präpositionalphrase mit einer bestimmten Präposition nehmen. Jedes *phrasal verb* soll einen eigenen Eintrag bekommen: Z.B. „look up“, „look over“

- Komplemente (Subkategorisierung): Es soll erfasst werden ob das Verb Nominalphrasen oder Präpositionalphrasen als Komplemente nimmt. Bei Präpositionalphrasen muss die dazugehörige Präposition mitgeliefert werden. Die Anzahl der Komplemente hängt davon ab, ob es ein intransitives, ein transitives oder ein ditransitives Verb ist (vgl. nach der Aufzählung, 28).

- Typisierung: Die Verben sollten in Zustands- oder Ereignisverben eingeteilt werden (vgl. Abschnitt 2.4.3).

- „kollektive“ Verben: Verben wie „gather“, „accumulate“, „collect“ etc. verlangen, dass die Nominalphrase, auf die sie Bezug nehmen, eine Menge re-

[4]Ein ähnliches Phänomen gibt es auch im Deutschen. Es gibt auch dort zum Verb gehörende Partikel, die äusserlich wie eine Präposition aussehen: Im Satz „Er ist nach Zürich gefahren.“ ist „nach“ eine Präposition, in den Sätzen „Er fuhr ihr nach.“ bzw. „Er ist ihr nachgefahren“ ist „nach“ ein zum Verb gehörender Partikel. Noch deutlicher wird es, wenn beide „nach“ im Satz vorkommen: „Er fuhr ihr nach Zürich nach.“

präsentiert. Das kann eine Nominalphrase im Plural („the boys“), eine zusammengesetzte (*conjoined*) Nominalphrase („John and Mary“) oder eine Nominalphrase mit einem *collective noun* („the club“) sein.

- Wird das Verb intransitiv verwendet, verlangt es ein Subjekt, das eine Menge repräsentiert: „The boys (*plural*) / John and Mary (*conjoined*) / the club (*collective*) gathered.“ Im Gegensatz dazu ergeben sich bei singulären Subjekten keine grammatikalisch korrekten Sätze: „*A boy / *John gathered“[5].
- Wird das Verb transitiv verwendet, verlangt es ein Objekt, das eine Menge repräsentiert: „He collected books (*plural*) for years“, „He collected thyme and rosemary (*conjoined*) for the sauce“, „He gathered an army (*collective*)“.

Im Lexikon wird angezeigt, ob das Verb ein solches „kollektives“ Verb ist.

Die Verben müssen wie die Nomen in drei Untergruppen eingeteilt werden: In intransitive, transitive und ditransitive Verben.
Intransitive Verben nehmen keine Komplemente, d.h. sie können zusammen mit dem Subjekt alleine als Satz stehen: „He sleeps“.
Transitive Verben verlangen ein Komplement. Die Sprache ACE lässt nur Nominalphrasen und Präpositionalphrasen als Komplemente zu (vgl. Abschnitt 2.4.2.7). Ein Beispiel für ein transitives Verb mit einer Nominalphrase als Komplement ist „love“ im Satz „He loves her“. Sätze wie „He loves to eat“ oder „He loves that she always says what she thinks“ sind nicht erlaubt, da „love“ in diesen Beispielen keine Nominal- oder Präpositionalphrase als Komplement nimmt.
Ditransitive Verben nehmen zwei Komplemente. Ein typisches ditransitives Verb ist „give“: „He gave his mother a kiss“ oder „He gave a kiss to his mother“. Im

[5]Beispiele von: `tristram.let.uu.nl/UiL-OTS/Lexicon/` und aus [ROHEN WOLFF et al. 1998, S. 57]

ersten Fall nimmt „give“ zwei Nominalphrasen als Komplemente, im zweiten Fall eine Nominalphrase und eine Präpositionalphrase.
In den meisten Fällen ist ein Verb nicht eindeutig einer der drei Kategorien zuteilbar. Viele Verben können wie z.B. „go“ mehreren Kategorien angehören: „He went“ (intransitiv), „He went a long way“ (transitiv mit Nominalphrase), „He went after her“ (transitiv mit Präpositionalphrase). In diesen Fällen wird für jede Möglichkeit ein eigener Eintrag im Lexikon stehen.

3.1.3. Adjektive

Für Adjektive müssen folgende Informationen zur Verfügung gestellt werden:

- Positivform (normale Form): „beautiful“ / „red“
- Komparativ- und Superlativformen, falls das Adjektiv steigerbar ist: „more beautiful“, „most beautiful“ / „redder“, reddest“. Nicht steigerbar ist z.B. „dead“
- Präpositionalphrasen-Komplement inkl. Präposition: Einige Adjektive nehmen eine Präpositionalphrase mit einer bestimmten Präposition als Komplement. Bsp: „keen on“, „fond of“

3.1.4. Adverben

Für Adverben schliesslich werden die folgenden Informationen verlangt:

- Lexem (das Adverb): „today“
- Modifikationstyp: Ist es ein temporales, modales, lokales oder instrumentales Adverb? Diese Einteilung ist nicht als fixe Vorgabe zu verstehen. Nicht jedes Lexikon bietet bei der Kategorisierung die gleiche Granularität und Ausführlichkeit. Die Wahl des Lexikons wird sich somit auf AceLex auswirken und in einer mehr oder weniger differenzierten Einteilung resultieren.

3.1.5. Zusammenfassung der inhaltlichen Anforderungen

In Tabelle 3.1 werden noch einmal die inhaltlichen Anforderungen an AceLex zusammengefasst.

nouns	Singular und Plural-Formen
	grammatisches Geschlecht
	Domäne (count oder mass)
	Objekt Typ (person/time/object)
	collective noun (yes/no)
	Einteilung in *common*, *proper* und *measurement nouns*
	wenn möglich: standard dimension
	wenn möglich: standard unit
verbs	3. Person Singular und Plural simple present tense
	Phrasale Komponente (z.B. „off“ bei „take off“)
	Komplemente (Subkategorisierung): nur NP und PP inkl. Präposition
	Typisierung (event/state)
	„kollektives“ Verb: Subjekt/Objekt im Plural (yes/no)
adjectives	Steigerungsformen (Positiv, Komparativ, Superlative)
	Komplemente (Subkategorisierung): nur PP inkl. Präposition
adverbs	Lexem
	Modifikationstyp (Richtung/Ursprung/Zeit/Dauer/Instrument/Art und Weise etc.)

Tabelle 3.1.: Inhaltliche Anforderungen an AceLex

3.2. Mögliche Quellen

Nach [Hess 2004a, S. 104] sollte ein brauchbares Lexikon der englischen Sprache etwa 200'000 Wörter umfassen. Ob es sich bei dieser Angabe um ein Vollformen-Lexikon oder ein lemma-basiertes Wörterbuch handelt, ist nicht ausgeführt[6]. Nichts desto trotz zeigt uns diese Angabe eindrücklich, wieviel Arbeit in einer solchen

[6]Es scheint mir wahrscheinlich, dass es sich hier um ein Vollformen-Lexikon handelt, da der in Abschnitt 3.2.1 beschriebene *Longman Dictionary of Contemporary English* (1978) ca. 53'000 Lemmata enthält und als gedrucktes englisches Lexikon hohen Anforderungen an die Vollständigkeit genügen muss.

Datensammlung steckt. Die Auswahl an brauchbaren Ressourcen ist dadurch beschränkt. Die Anforderungen an AceLex sind hoch: Es müssen viele Informationen für Attempto durch AceLex zur Verfügung gestellt werden. Dadurch sind Lexika, die sich zu sehr spezialisiert haben, nicht geeignet, um als Grundlage für AceLex zu dienen. Auf den folgenden Seiten möchte ich drei Lexika vorstellen, die aufgrund der Fülle ihrer Informationen als Basis für AceLex geeignet sind.
Gleich zu Beginn kristallisierte sich heraus, dass ein Lexikon nicht reichen würde, um allen Anforderungen von AceLex zu genügen. Vor allem semantische Aspekte wie die Einteilung von *measurement nouns* zur zugehörigen Dimension oder die Einteilung der Verben in *state* bzw. *event verbs* sind bei vielen Lexika nicht trivial oder gar nicht extrahierbar. Aus diesem Grund habe ich als viertes Lexikon noch WordNet als lexikalische Datenbank in meine Auswahl aufgenommen, da es in der Lage ist, die semantischen Aspekte von AceLex zu erfüllen.

3.2.1. LDOCE

Als die automatische Sprachverarbeitung den Schritt von den kleinen Pilotprojekten in die komplexe Welt der realen Sprache machte, sahen sich die Forscher dem Problem gegenüber, wie sie sich die grossen Mengen an Informationen über lexikalische Einheiten, die sie benötigten, beschaffen konnten. Die Lösung lag in maschinenlesbaren Versionen bereits in gedruckter Form publizierter Lexika. Da diese Lexika nicht für diesen Zweck erstellt worden waren, waren sie nicht in einem Format gespeichert, das sich für die Verarbeitung mit dem Computer wirklich geeignet hätte. [Hess 2004a, S. 105] zählt folgende Schwierigkeiten auf:

- rein technische Konversionsprobleme, insbesondere aufgrund herstellerspezifischer Drucksteuerungsbefehle.
- Umsetzung vom graphischen Erscheinungsbild in inhaltliche Auszeichnungen ist enorm schwierig (und fehleranfällig), da die Semantik hinter der Darstellung erkannt werden muss.

- geringe Konsistenz von Papierwörterbüchern, schwer nachzuprüfen von Hand!
- Unvollständigkeit von Papierwörterbüchern

Die grosse Menge an Informationen, die aus den Lexika gewonnen werden können, lassen die Schwierigkeiten bei der Extraktion der Daten jedoch in den Hintergrund treten. Der *Longman Dictionary of Contemporary English* (LDOCE) als *Machine-Readable Dictionary (MRD)* geriet schon früh in das Blickfeld dieser Forscher. Von den Forschern wird eine „lispifizierte“ Version des originalen maschinen-lesbaren Schriftsatz-Bandes[7] benutzt, das für den Druck des Lexikons gebraucht wurde. Details über die Umformatierung zu LISP finden sich in [Boguraev und Briscoe 1989] und [LDOCE, Lisp version 1978]. Für die englische Sprache ist LDOCE nebst OALD (*Oxford Advanced Learner's Dictionary*) der bekannteste dieser *Machine-Readable Dictionaries*.

LDOCE enthält ca. 53'000 Lemmata und liefert sehr viele und ausführliche Informationen über die Wortsilben, die Betonung und die Subkategorisierung. Ebenfalls geliefert wird eine kurze Definition des Wortes, die mit Hilfe eines kontrollierten Kern-Vokabulars (ca. 2'000 Wörter) erstellt wurde, um die Konsistenz zu wahren.

Ein illustrativer Eintrag findet sich in Beispiel 3.1. Der Eintrag des Adjektivs „abhorrent“ ist hier in der lispifizierten Version des Druckersteuerbandes des 1978 publizierten Lexikons zu sehen. Dieser Beispieleintrag ist auf den ersten Blick ziemlich verwirrend. Das kommt daher, dass wir hier eine nur leicht veränderte maschinenlesbare Schriftsatz-Datei betrachten, die ursprünglich zur schriftlichen Publikation diente. Druckrelevante Informationen über die Darstellung, wie z.B. ein Wechsel der Schriftart oder ob ein Wort fettgedruckt oder kursiv geschrieben wird, sind deshalb Teil des Eintrags und werden in Zeichen der Form *AB umgewandelt. *46 bewirkt zum Beispiel, dass ab hier kursiv geschrieben wird, *44 hebt die Kursivschreibweise wieder auf. Diese Informationen haben durchaus eine semantische Aussage: So

[7] auch Druckersteuerband genannt

```
((abhorrent)
  (1 A0003500 !< ab *80 hor *80 rent)
  (3 Eb"hQrEnt = -!"hCr-)
  (5 adj !<)
  (6 (*46 to *44) !<)
  (7 100 !< !< ---- !< ----T---Y)
  (8 hateful !; *CA DETEST *CB able : *46 Cruelty is abhorrent to him)
  (7 200 !< !< ---- !< ----T)
  (8 completely opposed in nature : *46 Cruelty is abhorrent to love)
  (10 1 !< -rence !< !< n !< U !<))
```

Beispiel 3.1: Eintrag aus LDOCE

ist das Wort, das beschrieben wird, immer fettgedruckt, die Beispiele in den Definitionen der Wortbedeutungen kursiv, Verweise auf andere Worte im Lexikon mit Grossbuchstaben geschrieben (im Beispiel 3.1 „DETEST“), etc..[8]

Der Eintrag ist nach folgender Struktur aufgebaut[9]:

Jeder Eintrag beginnt mit einem *head word* (im Beispiel 3.1 „abhorrent“). Hat das Wort mehrere komplett verschiedene Bedeutungen oder kann es mehr als einer syntaktischen Kategorie angehören, hat es mehrere Einträge im LDOCE. Auf das *head word* folgen verschiedene *subentries*, die alle mit einer Nummer gekennzeichnet sind (vgl. Beispiel 3.1 Einträge eins bis zehn). Die Anzahl der *subentries* variiert je nach Eintrag.

Die wichtigsten *subentries* sind:

1. Zeigt die Silbengrenzen des Wortes

2. Wenn ein Wort mehrere Einträge im LDOCE (aufgrund der oben erwähnten Eigenschaften) hat, dann werden in diesem *subentry* diese verschiedenen Einträge sequentiell numeriert.

[8] Das „<“-Zeichen ist der ursprüngliche Feldseparator, dem jeweils der *escape-character* „!“ des *Cambridge Lisp* vorangestellt ist.

[9] vgl. `www.scs.leeds.ac.uk/nti-kbs/ai5/Mrd/ldoce.html`

3. Zeigt, wie das Wort betont wird. Dazu werden die Symbole des *International Phonetic Alphabet (IPA)*[10] benutzt.

4. Welche Information dieser Untereintrag beinhalten soll, ist mir nicht bekannt. Vielleicht gibt es diesen Untereintrag gar nicht.

5. Gibt die syntaktische Klasse des Wortes an

6. Hier stehen *Grammar Codes*, die die Subkategorisierungseigenschaften des Wortes beschreiben. Z.B. wird hier spezifiziert, ob ein Verb intransitiv, transitiv oder ditransitiv ist. Im Beispiel 3.1 ist dort festgehalten, dass das Adjektiv „abhorrent“ die Präposition „to“ verlangt.

7. Hier stehen *subject field codes*, die das Wort einer breitgefassten Domäne zuordnen, wie z.B. Sport, Religion etc.. Wie in Beispiel 3.1 zu sehen, kann ein LDOCE-Eintrag mehrere *subentries* mit den Nummern 7 und 8 haben, wenn das Wort mehrere Bedeutungsaspekte hat.[11]

8. In diesem Untereintrag steht die Definition der Wortbedeutung, beschrieben mit dem 2'000 Wörter umfassenden Kern-Vokabular.

9. nicht bekannt, siehe Eintrag Nummer vier.

10. Hier stehen die möglichen Suffixe, die das Wort nehmen kann. Z.B. kann das Adjektiv „abhorrent“ in Beispiel 3.1 mit dem Suffix „-rence“ zum Nomen „abhorrence“ werden.

Es ist auch eine SGML-basierte maschinenlesbare Form der dritten Ausgabe (1995) des *Longman Dictionary of Contemporary English* erhältlich. Sie enthält den vollständigen Text des gedruckten Lexikons, das mit zusätzlichen

[10] Homepage der International Phonetic Association: `www2.arts.gla.ac.uk/IPA/ipa.html`

[11] Z.B. hat das Nomen „bank“ mit der Bedeutung „repository“ je einen Eintrag 7 und 8 für die Bedeutung „blood bank“ und „money bank“. „Bank“ im Sinne von „side of a river“ jedoch erhält einen eigenen, separaten Eintrag im LDOCE.

Häufigkeitsangaben und semantischen Codes versehen ist. Das Beispiel 3.2 zeigt den Eintrag von „checkroom“ im gedruckten Lexikon, danach folgt die SGML-basierte Darstellung in der maschinenlesbaren Datenbank[12] (Beispiel 3.3).

checkroom

/.../ n[C] AmE a place in a restaurant, theatre etc where you can leave your coat, bags etc to be guarded; CLOAKROOM (1) BrE

Beispiel 3.2: Eintrag aus dem *Longman Dictionary of Contemporary English* (Druck; 1995)

```
<Entry>
    <Head>
        <HWD>checkroom</HWD>
        <HYPHENATION>check& cdot;room</HYPHENATION>
        <PronCodes>
            <PRON>...</PRON>
        </PronCodes>
        <POS>n</POS>
        <GRAM>C</GRAM>
    </Head>
    <Sense>
        <LITTLEWORDS>AmE</LITTLEWORDS>
        <DEF>
            a place in a restaurant, theatre etc where
            you can leave your coat, bags etc to be guarded;
            <NonDV>
                <REFHWD>cloakroom</REFHWD>
                <REFSNUM>1</REFSNUM>
            </NonDV>
        </DEF>
        <EXAMPLE>BrE</EXAMPLE>
    </Sense>
</Entry>
```

Beispiel 3.3: Eintrag aus LDOCE3

[12]Informationen finden sich in [LDOCE3 1995]

Neben den *tags*, deren Bedeutung aus ihren Namen erschliessbar ist, wie <HWD> für *headword*, <POS> für *part of speech* etc., gibt das *tag* <NonDV> an, dass das folgende Wort nicht Bestandteil des kontrollierten Vokabulars von 2000 Wörtern ist. <REFHWD> zeigt einen Verweis auf ein anderes im Lexikon stehendes Wort an, wobei <REFSNUM> bei mehreren Einträgen die Nummer des gemeinten Eintrags angibt.
Die Benutzung der beiden LDOCE-Versionen ist nicht kostenlos. Es muss beim Verlag eine Lizenz erworben werden.

3.2.2. Celex

Das CEntre for LEXical Information (CELEX)[13] in den Niederlanden konstruierte eine lexikalische Datenbank für Holländisch, Englisch und Deutsch. Sie wurde als ein Gemeinschaftsunternehmen der Universität von Nijmegen, dem Institut für holländische Lexikologie in Leiden, dem Max Planck Institut für Psycholinguistik in Nijmegen und dem Institut für Wahrnehmungs-Forschung in Eindhoven gegründet. Finanziell unterstützt wurde CELEX vor allem von der *Netherlands Organization for Scientific Research* (NWO) und dem holländischen Amt für Wissenschaft und Bildung. CELEX ist nun ein Teil des Max Planck Instituts.
Die erste Version der CELEX *Lexical Database*[14] für Englisch wurde im Jahr 1988 herausgegeben. In diesem Abschnitt stütze ich mich auf die letzte Version 2.5 aus dem Jahr 1993. Das Projekt wurde im Jahr 2000 eingestellt.
CELEX enthält 52'446 Lemmata und liefert umfassende Informationen über Orthographie, Phonologie, Morphologie, Syntax und Häufigkeitsanalysen. Zum Beispiel liefert CELEX orthographische Information über die Anzahl Schreibweisen eines Wortes und ob es britisches oder amerikanisches Englisch ist. Der phonologische Teil

[13] Homepage von CEntre for LEXical Information: `www.kun.nl/celex`. In diesem Abschnitt stütze ich mich vor allem auf Informationen aus [CELEX English Linguistic Guide 1995] und [CELEX Readme-Datei 1995].
[14] In Zukunft werde ich die Datenbank nur noch CELEX nennen.

hält nicht nur fest, wie das Wort betont wird, sondern liefert auch eine phonetische Transkription, d.h. es zeigt wie das Wort ausgesprochen wird. Die morphologische Komponente spezifiziert, wie das Wort flektiert wird, aus welchen Morphemen es besteht, Derivationsinformationen und Ähnliches. Die syntaktische Komponente liefert die Subkategorisierungsstrukturen für die einzelnen Wortarten.

Die Basis für die englische Datenbank bildeten LDOCE und OALD (Oxford Advanced Learner's Dictionary, Roger Mitton's Version). Ebenfalls konsultiert wurde die *Collins-Birmingham University International Language Database* (COBUILD), wodurch korpusbasierte Häufigkeitsangaben gewonnen werden konnten. Ausserdem wurde für die Silbentrennung *Webster's Dictionary of American English* hinzugezogen.

Konzipiert wurde CELEX als Online-Datenbank (in einer ORACLE RDBMS-Umgebung), die man benutzen kann, um sich durch gezieltes Auswählen der benötigten Informationen ein eigenes, auf die jeweiligen Anforderungen zugeschnittenes Lexikon zu generieren. Eine reine ASCII Version kann auf CD-ROM für US-$150.00 vom *Linguistic Data Consortium* erworben werden[15]. Für jede der drei Sprachen steht sowohl ein lemma-basiertes Lexikon, wie auch ein Wortformen Lexikon zur Verfügung. Die lemma-basierten Lexika sind auf der CD-ROM in fünf Dateien aufgeteilt: Je eine für die Orthographie, die Phonologie, die Morphologie, die Syntax und die Häufigkeitsangaben. Die Wortformen Lexika sind analog aufgeteilt, nur die Dateien für die syntaktischen Angaben fehlen, da diese Angaben nicht abhängig von der Wortform sind und sie sich nicht von den Angaben für die Lemmata unterscheiden würden.

Bei CELEX gibt es zu beachten, dass es für Wörter, die sich nur in der Wortbedeutung unterscheiden, keine einzelnen Einträge als Lemma im Lexikon gibt. Das Nomen „bank“ hat in CELEX nur einen Eintrag für die beiden völlig verschiedenen

[15]Eine kostenlose Version kann auch online unter `www.mpi.nl/world/celex/` abgefragt werden, jedoch ist dies nur mit einem Netscape-Browser möglich. Ausserdem ist das web-basierte CELEX eine „experimentelle“ Version, die nicht immer korrekt funktioniert.

Bedeutungen „Ufer“ und „Finanzinstitut“. Nach [Burnage 1990, S. 22f.], unterscheidet Celex aufgrund fünf Kriterien, ob die Worte als separate Lemmata im Lexikon erscheinen:

- Orthographie der Wortformen: „peek“ und „peak“ erhalten verschiedene Einträge, da sie verschieden geschrieben werden.
- Syntaktische Klasse: Das Nomen „water“ und das Verb „water“ gehören verschiedenen Wortklassen an.
- Flexions-Paradigma: Das Nomen „antenna“, wie in (Radio)-Antenne, und das Nomen „antenna“, das die Fühler von Insekten beschreibt, sind zwei Lemmata in Celex, da das erstere die Pluralform „antennas“, das zweite die Pluralform „antennae“ hat.
- Morphologische Strukturen: Das Nomen „rubber“ (jemand der rubbelt) ist zurückzuführen auf *rub + er*: Das Verb „rub“ wurde durch die Anhängung des Suffix „-er“ nominalisiert. „rubber“ (Gummi) hingegen kann nicht mehr weiter in seine Bestandteile aufgelöst werden, es ist monomorphem. Die beiden Nomen unterscheiden sich in ihrer morphologischen Struktur.
- Aussprache: Das Verb „recount“ (wieder zählen) und das Verb „recount“ (eine Geschichte erzählen) sind als verschiedene Lemmata aufgeführt, da sie verschieden ausgesprochen werden.

Jeder Eintrag besteht aus einer Anzahl von Spalten, die durch einen Backslash „\“ separiert sind. Die Einträge selbst werden durch einen Zeilenumbruch getrennt. Jede Spalte steht für eine bestimmte Information, die meistens mit einem *YES*- oder *NO*-Flag bestätigt oder verneint wird. Dabei ist die Antwort nicht ausschliessend: D.h. ein Nomen, das je nach Umgebung *countable* oder *uncountable* ist, erhält bei beiden Spalten ein *YES*-Flag.

Besser verständlich wird der Aufbau von CELEX durch einen Beispieleintrag (3.4). Nehmen wir das Verb „accentuate“ und zwar aus der Datei `esl.cd`, die die syntaktischen Eigenschaften englischer Lemmata beschreibt.

```
207\accentuate\71\4\N\N\N\N\N\N\N\N\N\N\Y\N\N\N\N\N
\N\N\N\N\N\N\N\N\N\N\N\N\N\N\N\N\N\N\N\N\N\N\N\N\N
\N\N
```

Beispiel 3.4: Eintrag aus CELEX

In der ersten Spalte steht immer die *ID-Number*, die für jedes Lemma in CELEX eindeutig ist. Diese Nummer dient als Schlüssel, um die einzelnen Lexika miteinander zu verbinden[16]. In der zweiten Spalte wird das Lemma aufgeführt, das mit diesem Eintrag beschrieben wird, hier also „accentuate“. Die dritte Spalte gibt die Häufigkeit an, mit der das Lemma im COBUILD-Korpus erscheint[17], „accentuate“ findet sich insgesamt 71 mal in irgendeiner flektierten Form im COBUILD-Korpus. Die vierte Spalte gibt Auskunft über die Wortart. Die Klassennummern sind folgendermassen definiert: 1 noun, 2 adjective, 3 numeral, 4 verb, 5 article, 6 pronoun, 7 adverb, 8 preposition, 9 conjunction, 10 interjection, 11 single contraction (Bsp: „'re“ - Kurzform aus einem Wort „are“), 12 complex contraction (Bsp: „you're“ Kurzform aus zwei Wörtern „you“ und „are“).
Die restlichen Spalten sind nach Wortarten aufgeteilt. Die Spalten fünf bis 15 beschreiben Eigenschaften von Nomen, 16 bis 24 solche von Verben, 25 bis 29 sind den Adjektiven gewidmet, 30 bis 34 den Adverben, etc.. Ein Wort hat bei allen Spalten, die nicht seine Wortart betreffen, standardmässig ein *NO*-Flag. Beim Verb „accentuate“ sind also neben den bereits erwähnten Spalten eins bis vier nur die Spalten

[16] Wortformen-Lexikon-Dateien haben zwei *ID-Numbers*, eine, die sie mit dem jeweiligen Lemma verbindet und eine, um die Wortformen-Lexika untereinander zu verlinken.

[17] Wird eine Wortformen-basierte Datei von CELEX angeschaut, gibt diese Spalte darüber Auskunft, wie oft *genau* diese Wortform (ohne Rücksicht auf verschiedene Bedeutungen) im COBUILD-Korpus erscheint. In der Lemma-basierten Datei werden die Häufigkeiten der einzelnen flektierten Formen dieses Lemmas zusammengezählt und ergeben die erwähnte Nummer in der dritten Spalte.

16 bis 24 interessant.

Spalte 16 antwortet auf die Frage, ob das Verb (manchmal) ein direktes Objekt als Komplement nehmen kann, also transitiv verwendet werden kann. „accentuate“ als transitives Verb hat hier ein Flag mit dem Wert *YES*. Wäre das Verb z.B. transitiv und intransitiv (wie „leave“) verwendbar, hätte es zusätzlich zum *YES*-Flag in Spalte 16 („She left a will“) auch noch ein *YES*-Flag in Spalte 18, die auf die Frage antwortet, ob das Verb manchmal kein direktes Objekt nehmen kann („She left at ten o'clock“)[18].

Wie bereits oben erwähnt, möchte CELEX dem Benutzer die Möglichkeit geben, sich ein ganz individuelles Lexikon zusammenzustellen. Durch Verbindung zwischen den einzelnen Dateien und der Wahl der in der jeweiligen Datei interessierenden Spalten kann ein genau abgestimmtes Lexikon erstellt werden.

Die lexikalische Datenbank ist auf CD-ROM durch die computerlinguistische Abteilung des Instituts für Informatik der Universität Zürich erworben und lizenziert.

3.2.3. Comlex

An der Universität von New York haben Adam Meyers, Catherine Macleod und Ralph Grishman im Rahmen des Projekts *Proteus* ein monolinguales englisches Wörterbuch-Lexikon entwickelt, das ungefähr 38'000 Lemmata umfasst. Das Lexikon trägt den Namen COMLEX *Syntax*[19] und enthält ca. 21'000 Nomen, 8'000 Adjektive und 6'000 Verben. Vertrieben wird es vom *Linguistic Data Consortium (LDC)*. Die erste Version wurde im Mai 1994 an das *LDC* übergeben, die letzte Version 3.1 im Dezember 1997. Es wurde ganz gezielt für den Gebrauch im Bereich des *natural language processing* konzipiert. Die Entwickler versuchten, das Lexikon für einen möglichst grossen Benutzerkreis attraktiv zu gestalten, indem sie eine grosse Anzahl von syntaktischen Merkmalen (*Features*) verwendeten und diese in einer

[18] Details und genaue Erklärungen finden sich auf der CD-ROM in der Datei `eug_a4.ps`.

[19] `nlp.cs.nyu.edu/comlex/index.html`

relativ theorie-unabhängigen Weise implementierten. Die Zielsetzung der Entwickler geht dahin, dass dieses syntaktische Lexikon mit einem phonologischen[20] und semantischen Lexikon (z.B. WordNet) kombiniert wird. So würden lexikalische Datensammlungen entstehen, die verschiedensten Anforderungen genügen.
Die Einträge sind in einer Lisp-Listennotation (in einer sogenannten *typed feature structure*) gehalten, die auf einfache Weise in andere Formate wie Prolog oder SGML-markierten Text überführt werden kann. Jeder Eintrag besteht aus einem (Wort-)Klassensymbol, das von *feature-value*-Paaren (Schlüssel-Wert-Paaren) gefolgt wird. Jeder Wert wiederum kann ein Symbol, eine Zeichenkette (*String*), eine Liste von *strings*, eine *feature-value*-Liste oder eine Liste von *feature-value*-Listen sein. Die Abbildung 3.1 zeigt den Aufbau der *typed feature structure* von Comlex in UML-Notation[21]. Im Beispiel 3.5 sind einige Comlex-Einträge aufgeführt.

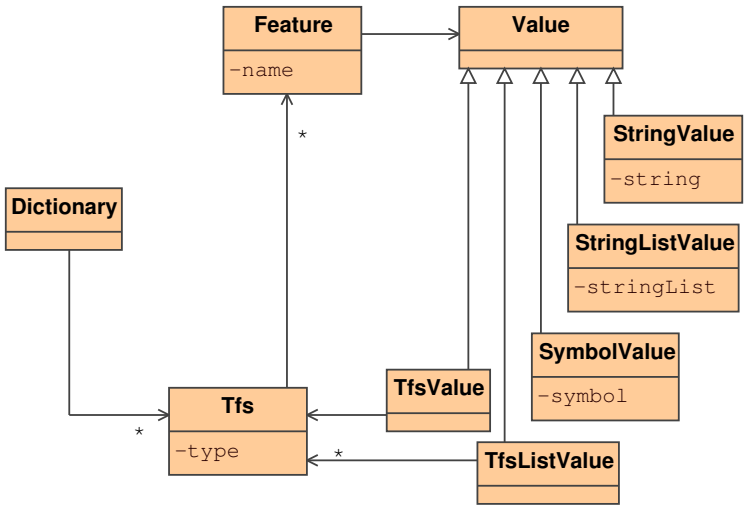

Abbildung 3.1.: Die *typed feature structure* von Comlex

Wie bereits erwähnt gibt das erste Symbol die Wortklasse an. Ist es möglich, ein Wort in mehrere Wortklassen einzuteilen, steht für jede Klasse ein eigener Eintrag

[20] Comlex *English Pronouncing Lexicon*, auch bekannt als PRONLEX
[21] Unified Modeling Language, Homepage: `www.uml.org`

```
(VERB          :ORTH       „accentuate“
               :SUBC       ((NP)
                           (THAT-S)
                           (POSSING))
               :FEATURES   ((VVERYVING    :PASTPART T)))
(NOUN          :ORTH       „assertion“
               :SUBC       ((NOUN-THAT-S)
                           (NOUN-BE-THAT-S)))
(ADVERB        :ORTH       „even“
               :MODIF      ((PRE-SCONJ)
                           (PRE-DET&PRO)
                           (PRE-COMPARATIVE)
                           (CLAUSAL-ADV   :PRE-VERB T
                                          :INIT T
                                          :VERB-OBJ T)
                           (PRE-ADJ)
                           (PRE-ADV)
                           (PRE-PREP)
                           (PRE-QUANT)))
(ADJECTIVE     :ORTH       „abhorrent“
               :SUBC       ((ADJ-PP :PVAL („to“))
                           (EXTRAP-ADJ-THAT-S)
                           (EXTRAP-ADJ-FOR-TO-INF))
               :FEATURES   ((GRADABLE)))
(NOUN          :ORTH       „checkroom“
               :FEATURES   ((COUNTABLE)))
(TITLE         :ORTH       „Prof.“)
(NOUN          :ORTH       „professor“
               :FEATURES   ((NHUMAN)
                           (NTITLE)
                           (COUNTABLE     :PRED T)))
```

Beispiel 3.5: Einträge aus COMLEX

im Lexikon. COMLEX ist ein lemma-basiertes Lexikon. Jeder Eintrag hat ein Feature :ORTH, welches die Grundform des Wortes als Wert hat (im Beispiel 3.5 „accentuate“, „assertion“ etc.). Verben, Nomen und Adjektive, welche bei der Flexion unregelmässige Formen hervorbringen, haben für diese zusätzliche Einträge wie z.B. :PAST, :PLURAL etc.. Regelmässige Formen können mit in [MACLEOD et al. 1998, S. 7] angegebenen Regeln berechnet werden. COMLEX stellt ausserdem Lisp Makros zur Verfügung, die das Lexikon in ein Vollformen-Lexikon expandieren, d.h. es würde dann ein Eintrag für jede flektierte Form eines Wortes im Lexikon stehen.
Wörter, die Komplemente nehmen, haben ein Merkmal :SUBC, welches die verschiedenen Subkategorisierungsmöglichkeiten[22] enthält. Für Verben ist dieses Subkategorisierungsmerkmal obligatorisch, für Adjektive und Nomen optional. Andere syntaktische Merkmale werden unter :FEATURES gefasst. Nomen haben neun mögliche Features und neun mögliche Subkategorisierungs-Komplemente, Adjektive sieben Features und 14 Komplemente, Verben fünf Features und 92 Komplemente, wobei für 750 häufige Verben noch weitere vier mögliche Features und 32 Komplemente dazukommen. Zusätzlich sind die 750 häufigsten Verben mit *tags* gekennzeichnet, die auf ihr Vorkommen in einem Korpus[23] verweisen und mit der jeweiligen COMLEX-Klasse markiert sind. Auch für Adverben liefert COMLEX ausführliche syntaktische Informationen: Sie haben 12 mögliche Features. Ausserdem sind unter dem Merkmal :MODIF (*Modification Structure*) die möglichen Positionen erfasst, die das Adverb im Satz einnehmen kann.[24] Einige exemplarische Einträge finden sich in Beispiel 3.5. COMLEX ist ein Wörterbuch, dass sich auf die Syntax der Wörter konzentriert. Wörter der gleichen Wortklasse, die gleich geschrieben sind, aber andere Bedeu-

[22]Die Namen dieser Subkategorisierungen sind meistens selbsterklärend. Zum Beispiel nehmen Verben, die mit „NP“ markiert sind, eine Nominalphrase als Komplement. Jedes Komplement wird durch einen Subkategorisierungs-*Frame* definiert, die in [MACLEOD et al. 1998] beschrieben sind.

[23]Der Korpus besteht unter anderem aus dem *Brown Corpus*, Literaturauszügen aus der *Library of America* und mehreren Zeitungen wie z.B. dem *Wall Street Journal* und dem *San Jose Mercury*.

[24]Detaillierte Informationen über die verschiedenen Merkmale finden sich in [ROHEN WOLFF et al. 1998].

tungen haben, erhalten keine eigenen Einträge, sondern sind in einem Eintrag zusammengefasst. So sind „bank “ im Sinne von Ufer und Finanzinstitut durch einen Eintrag in COMLEX repräsentiert. Auch die beiden Nomen „antenna“ im Sinne von (Radio-)Antenne und Fühler von Insekten (vgl. Abschnitt 3.2.2, Seite 37) und die beiden Verben „recount (wieder zählen) und „recount“ (eine Geschichte erzählen) sind jeweils in einem Eintrag zusammengefasst.
Wie auch für CELEX besitzt die computerlinguistische Abteilung eine Lizenz für COMLEX. Sie hat für die ganze Universität Zürich Gültigkeit.

3.2.4. WordNet

WORDNET ist eine lexikalische Datenbank für die englische Sprache, die online abgefragt werden kann. Eine Version für Windows und Unix kann jedoch auch von der Homepage[25] von WORDNET heruntergeladen werden. Seit 1985 wird es am Cognitive Science Laboratory der Princeton University, New Jersey, USA entwickelt und immer wieder erweitert. Die Benutzung ist kostenlos[26].
Das Design von WORDNET wurde an aktuelle psycholinguistische Theorien über das menschliche Wortgedächtnis angelehnt. Englische Verben, Nomen, Adjektive und Adverben sind in Synonymgruppen, sogenannten *Synsets* geordnet. Jedes dieser Synsets repräsentiert ein lexikalisches Konzept, d.h. die Datenbank ist nach Kriterien der Wortbedeutung und nicht nach Wortformen organisiert. Begriffe mit gleicher oder ähnlicher Bedeutung (Synonyme) werden in einem Synset zusammen abgelegt[27]. In WORDNET wird folgende Regel verwendet, um Synonyme zu bestimmen: Wenn der Austausch des einen Wortes durch das andere den Wahrheitswert einer Aussage nicht verändert, werden die Wörter als Synonyme behandelt (z.B. Fanta-

[25] `www.cogsci.princeton.edu/~wn/`
[26] Verschiedene Länder haben seitdem ähnliche Datenbanken aufgebaut. Es existiert auch eine deutsche Version, GermaNet, die jedoch kostenpflichtig ist. Homepage `www.sfs.uni-tuebingen.de/lsd/`
[27] In der aktuellen Version WORDNET 2.0 sind 152'059 Wörter erfasst, die in 115'424 Synsets eingeteilt sind.

sie und Vorstellungskraft). Semantische Beziehungen wie Hyperonymie, Meronymie und Antonymie verbinden die Synsets.

Um die Relationen zwischen den Synsets darzustellen und zu untersuchen, stellt WordNet ein GUI (Graphical User Interface) zur Verfügung. Die Abfrage über einen Kommandozeilenprompt ist ebenfalls möglich. Die Datenbank selbst ist in zwei Textdateien aufgeteilt: Eine Index-Datei und eine Daten-Datei. Die Index-Datei enthält die Worteinträge, wobei jeder Eintrag die Wortform als Index-Schlüssel und Zeiger in die Daten-Datei auf alle Synsets enthält, in denen dieses Wort vorkommt. Die Daten-Datei besteht aus der Sammlung von Synsets. Jedes Synset enthält eine Liste mit den synonymen Wortformen und eine Liste mit Zeigern, die die semantischen Beziehungen zwischen diesem und anderen Synsets repräsentieren. In der Abbildung 3.2 wird der Aufbau von WordNet anhand des in Beispiel 3.6 vorgestellten Nomens „assertion“ aufgezeigt. Neben den im Beispiel 3.6 beschriebenen Hypernym-Beziehungen (nur eine Stufe), sind in der Abbildung 3.2 auch noch einige Hyponym-Beziehungen eingezeichnet.

Nicht jede semantische Beziehung eignet sich für die Beschreibung jeder Wortart: Für alle Wortarten stehen Informationen über Synonymie zur Verfügung. Bei den Nomen sind alle Synsets durch Hyperonymie/Hyponymie in einer Hierarchiestruktur organisiert, wobei sich jedes Synset auf mindestens ein „Ur-Synset“, einen sogenannten *unique beginner* zurückführen lässt. Beispiele für *unique beginners* sind: Entitäten, psychologische Merkmale, Abstraktionen, Raum, Ereignisse, Zustände etc.. Im Beispiel 3.6 findet sich der Eintrag für das Nomen „assertion“, inklusive seiner Hypernym-Beziehungen bis zum *unique beginner*. „assertion“ ist zwei Synsets zugeordnet, hat also zwei Bedeutungen. Die erste Bedeutung lässt sich auf den *unique beginner abstraction*, die zweite Bedeutung auf *human action* zurückführen.

Bei den Adjektiven ist eine solche Kategorisierung jedoch nicht so einfach möglich. Es gibt hauptsächlich zwei Arten von Adjektiven: beschreibende (deskriptive) und relationale Adjektive. Beschreibende Adjektive dienen der näheren Erläuterung eines Nomens (Beispiel: großer Mann), während relationale Adjektive von einem Nomen

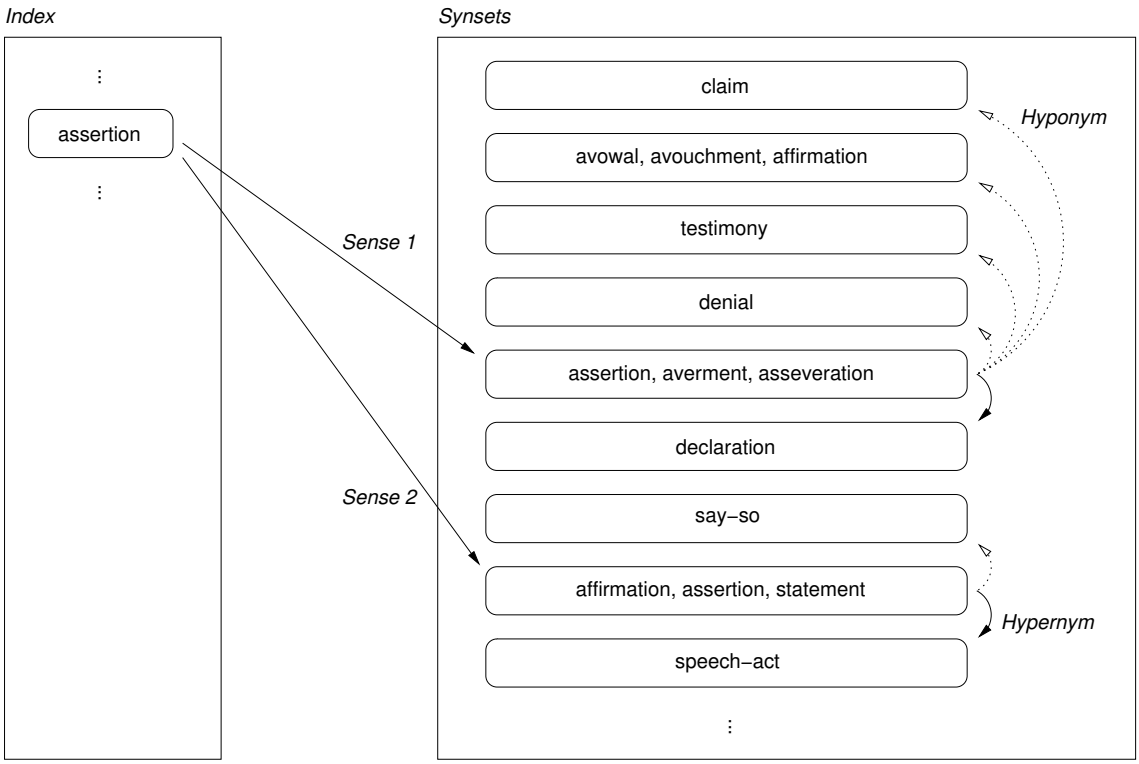

Abbildung 3.2.: Aufbau der Datenbank WordNet

1. assertion, averment, asseveration – (a declaration that is made emphatically (as if no supporting evidence were necessary))

   ⇒ declaration – (a statement that is emphatic and explicit (spoken or written))

   ⇒ statement – (a message that is stated or declared; a communication (oral or written) setting forth particulars or facts etc; „according to his statement he was in London on that day“)

   ⇒ message, content, subject matter, substance – (what a communication that is about something is about)

   ⇒ communication – (something that is communicated by or to or between people or groups)

   ⇒ social relation – (a relation between living organisms (especially between people))

   ⇒ relation – (an abstraction belonging to or characteristic of two entities or parts together)

   ⇒ abstraction – (a general concept formed by extracting common features from specific examples)

2. affirmation, assertion, statement – (the act of affirming or asserting or stating something)

   ⇒ speech act – (the use of language to perform some act)

   ⇒ act, human action, human activity – (something that people do or cause to happen)

Beispiel 3.6: Eintrag aus WordNet: „assertion“

abgeleitet sind und in enger Beziehung zu ihm stehen (Beispiel: majestätisch). Bei den beschreibenden Adjektiven werden Gruppen (*cluster*) ähnlicher Adjektive gebildet, deren Kern ein zentrales Adjektiv ist, das ein Antonym besitzt. Mit Hilfe dieser zentralen Adjektive werden dann die Beziehungen zu anderen solchen Gruppen charakterisiert. In der Abbildung 3.3 ist „damp“ ein Adjektiv, das durch seine Ähnlichkeit zum zentralen Adjektiv „wet“ definiert ist. Sucht man Antonyme zu „damp“, liefert WordNet „dry“, da es das Antonym vom zentralen „wet“ ist[28].

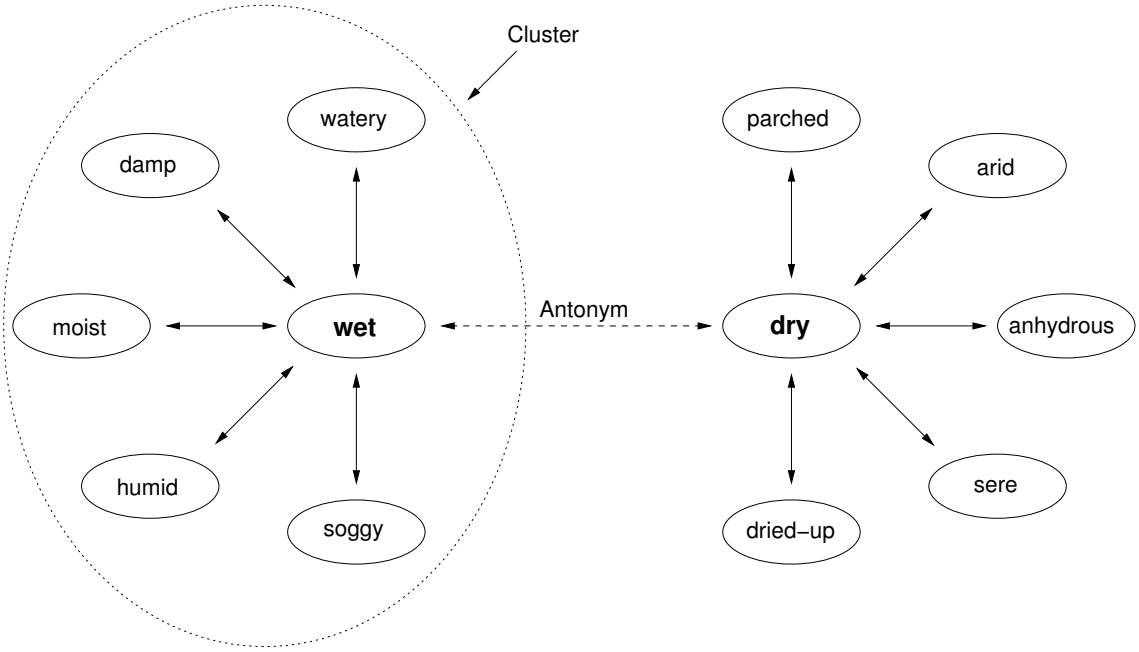

Abbildung 3.3.: Clustering von Adjektiven

Relationale Adjektive besitzen keine Antonyme und werden darum nicht in *cluster* zusammengefasst. Sie werden zusammen mit Zeigern, die auf die Nomen verweisen, von denen sie abgeleitet sind, in einer eigenen Datei gespeichert.

Verben werden in 15 Klassen eingeteilt, die nach semantischen Gesichtspunkten unterschieden werden (z.B. Kommunikationsvorgänge, Verben der Bewegung, Emotionen etc.). Bei den Verben liefert WordNet wie bei den Nomen zusätzlich zu

[28]Text und Grafik angelehnt an ein Beispiel von `www.aifb.uni-karlsruhe.de/~sst/Teaching/KDDSemSS00/node8.html`

den Synonymie-Beziehungen Informationen über Antonyme, Hyper- und Hyponyme. Ausserdem enthalten Einträge von Verben Angaben über Subkategorisierungsrahmen in Form eines verallgemeinerten Beispielsatzes, wie der Satz „somebody accentuates something“ in Beispiel 3.7. Das dort beschriebene Verb „accentuate“ ist in zwei Synsets eingeteilt, hat also zwei verschiedene Bedeutungsaspekte. Die Einträge und dazugehörigen Subkategorisierungsrahmen finden sich im Beispiel 3.7.

1. stress, emphasize, emphasise, punctuate, accent, accentuate – (to stress, single out as important; „Dr. Jones emphasizes exercise in addition to a change in diet“)
   Somebody ____s something
   Something ____s something
   Somebody ____s that CLAUSE
2. stress, accent, accentuate – (put stress on; utter with an accent; „In Farsi, you accent the last syllable of each word“)
   Somebody ____s something

Beispiel 3.7: Eintrag aus WordNet: „accentuate“

3.2.5. Aufstellung

In Tabelle 3.2 folgt eine Aufstellung, welches Lexikon, welche Informationen für AceLex liefern kann. WordNet ist nicht in dieser Aufstellung enthalten, da sich diese Quelle zu sehr auf die semantischen Aspekte konzentriert und zuwenig Informationen über grammatische Eigenschaften der Worte liefert. Es bietet sich jedoch als ergänzende Informationsquelle an, um eventuelle semantische Lücken der anderen Lexika zu schliessen. LDOCE1 und LDOCE3 sind nicht einzeln aufgeführt. Wie aus Tabelle 3.2 zu ersehen ist, kann keines der Lexika die *standard dimension* und die *event/state*-Domäne liefern. LDOCE stellt als einziges der drei Lexika Informationen über das grammatische Geschlecht zur Verfügung. Das Geschlecht und

Wortart	Anforderungen	LDOCE	CELEX	COMLEX
nouns	Sg.+ Pl.	ja	ja	ja
	gr. Geschlecht	ja	nein	nein
	count/mass	ja	ja	ja
	Objekt Typ	ja	nein	time, person
	collective noun	ja	ja	ja
	common, proper, measurement n.	nur aus Def.	nein	ja
	standard dimension	nur aus Def.	nein	nein
	standard unit	nur aus Def.	nein	ja
verbs	3.P. Sg.+ Pl.	ja	ja	ja
	Phrasale Komponente	(ja)	ja	ja
	Komplemente	(ja)	ja	ja
	Typ: Event/State	nein	nein	nein
	Subjekt/Objekt im Plural	nein	nein	ja
adjectives	Steigerungsformen	ja	ja	ja
	PP-Komplemente	ja	nein	ja
adverbs	Lexem	ja	ja	ja
	Modifikationstyp	nein	nein	teilweise

Tabelle 3.2.: Erfüllung der Anforderungen an AceLex

die *standard dimension* sind Informationen, die nur für wenige Worte von Bedeutung sind. Falls diese Informationen aus der lexikalischen Quelle nicht extrahierbar sind, kann man die betreffenden Worte ohne grösseren Aufwand manuell klassifizieren. Anders sieht es bei der Einteilung der Verben in *events* und *states* aus. Jedes Verb muss eine solche Aufteilung aufweisen. Das kann nicht von Hand geschehen. Leider liefert keines der Lexika diese Information. CELEX fällt ein bisschen aus dem Rahmen, denn es kann zusätzlich wichtige Informationen nicht liefern, wie z.B. den Objekt Typ bei den Nomen, die PP-Komplemente der Adjektive und die Unterscheidung zwischen *common, proper* und *measurement nouns*. Von den insgesamt 17 Kriterien in der Tabelle 3.2, werden nur acht durch CELEX erfüllt.

LDOCE ist ein Lexikon, das sehr viele Informationen enthält. Im Gegensatz zu den anderen beiden liefert es auch eine kurze Definition des Wortes. Eventuell wäre es möglich, aus diesen Definitionen die noch fehlenden semantischen Informationen zu ziehen, wie z.B. die über die *standard dimension* oder *standard unit*. Dies wäre jedoch eine keineswegs triviale Aufgabe. Die Extraktion dieser Information aus WORDNET wäre einfacher zu bewerkstelligen. Da die *standard units* nicht bestimmt werden können, scheitert auch die Unterscheidung von *common nouns* und *measurement nouns*. *Proper nouns* können erkannt werden, da sie mit einem Grossbuchstaben beginnen. Bei den Adverben stellt LDOCE keine Informationen über den Modifikationstyp bereit, die Information liesse sich auch aus den Definitionen nicht extrahieren. LDOCE erfüllt elf der insgesamt 17 Kriterien der Tabelle 3.2. Eine weitere Schwierigkeit ist, dass im LDOCE nicht zwischen eigentlichen *phrasal verbs*, die einen Partikel nehmen und *prepositional verbs*, die immer eine PP mit einer bestimmten Präposition verlangen, unterschieden wird. Darum sind in Tabelle 3.2 die beiden *ja* bei der phrasalen Komponente und den Komplementen in Klammern gesetzt. In LDOCE findet sich z.B. „care for“ als ein eigener Eintrag als *phrasal verb*, obwohl es sich bei „for“ um eine Präposition handelt: Es ist unmöglich,

„for“ hinter die Nominalphrase zu setzen[29] (vgl. Abschnitt 3.1.2). ATTEMPTO verlangt jedoch eine Differenzierung zwischen verbalem Partikel und Präposition. Die Unterscheidung könnte jedoch über einen Umweg trotzdem erreicht werden.
Bei COMLEX fehlen nur drei der 17 geforderten Informationen. Im Gegensatz zu LDOCE kann durch die Kennzeichnung von *units* in COMLEX die Einteilung in die drei Kategorien *common, proper* und *measurement noun* vorgenommen werden. COMLEX liefert bei den Adverben eine grobe Einteilung in *temporal*, *space* und *manner* Adverben. In COMLEX ist die Unterscheidung zwischen verbalem Partikel und Präposition gegeben, sie muss im Gegensatz zu LDOCE nicht noch speziell erarbeitet werden. COMLEX scheint mit seinen 38'000 Lemmata etwas kleiner als LDOCE (ca. 53'000 Lemmata) und CELEX (52'446 Lemmata) zu sein. Jedoch kann nicht abgeschätzt werden, inwiefern dieser mengenmässige Unterschied in COMLEX aufgrund der fehlenden Unterscheidung der einzelnen Wortbedeutungen entsteht (vgl. Abschnitt 3.2.3, Seite 43). Die Grössenangaben dieser Lexika hat daher keinen Einfluss auf die Wahl des Lexikons für ACELEX.

3.3. Basis für AceLex

Ich habe mich entschieden, COMLEX als Basis für ACELEX zu verwenden. Nicht alle gewünschten Informationen werden aus COMLEX extrahierbar sein: das Geschlecht von Personen, die *standard dimension* und die *event/state*-Domäne bei Verben können nicht aus COMLEX gewonnen werden und müssten auf andere Weise ACELEX hinzugefügt werden. Dabei könnten eventuell auch die Modifikationstypen der Adverben etwas verfeinert werden. Die *event/state*-Domäne und die Modifikationstypen sind die Schwierigkeiten, die unbedingt noch gelöst werden müssen, die anderen beiden Kriterien könnten von Hand eingefügt werden. Als zusätzliches

[29] „He cares for his mother.“ „*He cares his mother for“

Lexikon bietet sich WordNet[30] an: Es würde die fehlenden Informationen über Geschlecht, *standard dimension* und die *event/state*-Domäne[31] bereitstellen können. Diese Vervollständigung des Lexikons anhand einer zweiten lexikalischen Quelle wäre Bestandteil einer weiterführenden Arbeit. In dieser Arbeit wird nur ein Lexikon auf der Basis von Comlex erarbeitet.

[30] FrameNet wäre wahrscheinlich geeigneter: Jedoch sind erst ca. 10'000 Einträge erfasst. Es gibt jedoch auch Bestrebungen, WordNet und FrameNet zu verknüpfen. Website von FrameNet: `www.icsi.berkeley.edu/~framenet/`

[31] Reine Zustandsverben können mit Hilfe von WordNet gut erkannt werden. Die Einteilung in *state* oder *events* von Prozessverben, die sowohl Eigenschaften von Zuständen, wie auch von Ereignisverben haben (vgl. Abschnitt 2.4.3), müsste noch ausführlich untersucht werden.

4. Von Comlex zu AceLex

Nach der Entscheidung für ein Lexikon stellt sich nun die Frage, wie AceLex aufgebaut ist. Dieses Kapitel soll als Referenz dienen, um die in AceLex gespeicherten Informationen richtig verstehen und einordnen zu können. Die Struktur der Einträge sind den vom lexikalischen Editor (vgl. Abschnitt 2.4.1) generierten Einträgen nachempfunden, damit die Einträge falls nötig mit dem Editor noch angepasst werden könnten.
In den folgenden Abschnitten wird für jede Wortart vorgestellt, wie die Einträge in AceLex dargestellt sind. Ausserdem wird in detaillierter Form dargelegt, aus welchen Merkmalen in Comlex die für AceLex relevanten Informationen gewonnen werden. Dabei wird auf Unzulänglichkeiten bei der Extraktion hingewiesen.

4.1. Nomen

Nomen müssen in AceLex in drei Kategorien eingeteilt werden: In die *common*, die *proper* und die *measurement nouns* (vgl. Abschnitt 3.1.1).
Common nouns sind „normale“, klein geschriebene Nomen wie z.B. „cat“, „tree“ oder „water“. Sie können in *countable* oder *uncountable (mass) nouns* unterteilt werden. *Proper nouns* sind Namen von spezifischen Personen, Firmen, Orten, Sportteams etc. und werden in der englischen Sprache gross geschrieben. Beispiele hierfür sind „John“, „Mary“, „England“, „Sony“, „McDonalds“ oder wie in Beispiel 4.2 Monatsnamen. Auch die Wochentage, Feiertage („Christmas“) und Namen von Büchern („The Lord of the Rings“) und Filmen sind *proper nouns*.

In Comlex gibt es gemäss Spezifikation folgende Möglichkeiten, um die *proper nouns* zu identifizieren: Es gibt ein Merkmal NAME, das nach seiner Definition in [Rohen Wolff et al. 1998, S. 3] Personennamen wie „John“ oder „Smith“ liefern sollte. Das Merkmal NAME ist folgendermassen definiert[1]:

> „A noun is classified as NAME if it can occur after words such as PROF. (NTITLE)[2], INVESTIGATOR (PREDNOUN), initials (INITIAL), or MARY (NAME).

Bei einer Überprüfung, welche Worte in Comlex mit NAME markiert sind, habe ich nur drei Wörter gefunden: „missy“, „mister“ und „poppa“. Ganz offensichtlich liegen hier gleich bei allen drei Wörtern Fehler vor: Weder die beiden Verniedlichungen „missy“ und „poppa“, noch „mister“ könnten hinter z.B. „Professor“ stehen. Das Merkmal NAME kann nicht als Indikator für einen Personennamen verwendet werden, obwohl die Spezifikation das Merkmal so beschreibt.
In Comlex gibt es keine weiteren Merkmale, die für die Unterscheidung nützlich wären. Einzig die von den *proper nouns* verlangte Grossschreibung der Wörter kann noch bei der Einteilung behilflich sein. Bei einer Überprüfung stellte sich heraus, dass in Comlex nur 32 Nomen gross geschrieben werden. Das sind die Wochentage und die Monatsnamen, sowie „Christmas“, „Christmastime“ und die Abkürzungen „ATM“, „CEO“, „CFC“, „CPU“, „GDP“, „GNP“, „PC“, „PVC“, „TV“, „VCR“ und „VTR“. Die Wochentage, Monatsnamen und der Feiertag „Christmas“ bzw. „Christmastime“ sind *proper nouns*. Die Abkürzungen wie „CPU“ für „central processing unit“ hingegen sind *common nouns*. Es fällt auf, dass die einzigen *proper nouns* Zeitangaben sind. Anscheinend wurde bewusst auf die Aufnahme von anderen *proper nouns*, wie z.B. Orte („New York“) und Firmennamen („McDonalds“) verzichtet.

[1] Details zu den Definitionen können jeweils in [Rohen Wolff et al. 1998] nachgelesen werden.
[2] In Klammern steht jeweils das Comlex-Merkmal, womit das vorangehende Wort markiert ist. Hier ist es aber gleich etwas verwirrend: Prof. als Abkürzung wäre eigentlich Wortart TITLE, professor hingegen korrekterweise NOUN mit Merkmal NTITLE (vgl. Fussnote 10 in diesem Kapitel).

Die aufgenommenen Abkürzungen und die gefundenen *proper nouns* können deshalb durch das Nicht-Vorhandensein bzw. Vorhandensein eines Merkmals NTIME1 oder NTIME2 voneinander unterschieden werden[3]. Eine andere Möglichkeit ist in COMLEX nicht gegeben.

Measurement nouns sind Nomen, die eine Masseinheit ausdrücken. In COMLEX existiert ein Merkmal NUNIT, das in [ROHEN WOLFF et al. 1998, S. 8] spezifiziert ist:

> NUNIT nouns can occur in the measure sequence Q [Quantifier] + noun + pp [prepositional phrase] or Q + noun + adj, where the pp or adj express a dimension (IN LENGTH, OF AGE, LONG, OLD) (cf. ASCALE[4]). In predicate position, an NUNIT agrees in number with Q.

Nach dieser Definition sind Wörter, die Masseinheiten ausdrücken, mit NUNIT markiert: „meter“, „kilogramm“, „years“. Ausserdem sind Geldwährungen als NUNIT gekennzeichnet. Neben diesen Standardeinheiten sind in COMLEX auch Einheiten wie „tablespoon“, „block“ „day“, „hand“, „slice“ etc. mit NUNIT erfasst, die nicht nur als Einheiten angesehen werden können. Aus diesem Grund werden die mit NUNIT markierten Wörter in ACELEX nicht nur als *measurement nouns*, sondern auch als *common nouns* aufgeführt. Es findet also ein Verdoppelung der Einträge statt.

Um einen Überblick über diese verschiedenen Nomen zu geben, wird von jedem Nomen-Typ ein Beispieleintrag vorgestellt. Die Einträge von *common nouns* sind analog des Beispiels 4.1 aufgebaut, Einträge von *proper nouns* sind in ACELEX wie in Beispiel 4.2 dargestellt und ein Eintrag eines *measurement nouns* findet sich in Beispiel 4.3.

Jeder Eintrag ist in der Form eines zweistelligen komplexen PROLOG-Terms[5] *lexicon/2* aufgebaut. Wie oben schon erwähnt, ist die Struktur der Einträge in ACELEX

[3] Das Nomen „Christmastime“ wird bei diesem Vorgehen jedoch nicht erfasst: Es ist aus ungeklärten Gründen nicht als NTIME1 markiert. Hier ist eine Korrektur von Hand nötig.

[4] In COMLEX sind diese *dimensions* ausdrückenden Adjektive mit ASCALE markiert, die *dimensions* ausdrückenden Nomen mit NSCALE.

[5] Eine Einführung in PROLOG findet sich unter `www.ifi.unizh.ch/cl/siclemat/lehre/ws0405/pcl1/`. Ein ganzes Buch „Logic, Programming and Prolog“ (Nilsson et al.) kann unter dieser Adresse kostenlos heruntergeladen werden: `www.ida.liu.se/~ulfni/lpp/`

lexicon(cn, [logical_relation([cat]), singular([cat]), plural([cats]), singular_aliases([[**kitten**], [**kittycat**]]), plural_aliases([[**kittens**]]), type([object]), gender([neuter]), collective_noun([no]), group([countable]), comment([**'cats are afraid of dogs'**])]).

Beispiel 4.1: AceLex-Eintrag *common noun*

lexicon(pn, [logical_relation([April]), singular([April]), plural([Aprils]), singular_aliases([[**Apr.**]]), plural_aliases([]), type([time]), gender([neuter]), comment([**'In this month, the weather is always changing'**])]).

Beispiel 4.2: AceLex-Eintrag *proper noun*

lexicon(mn, [logical_relation([meter]), singular([meter]), plural([meters]), singular_aliases([]), plural_aliases([]), dimension([**length**]), comment([])]).

Beispiel 4.3: AceLex-Eintrag *measurement noun*

der Struktur nachempfunden, die vom lexikalischen Editor erzeugt wird. Als erstes Argument des Terms *lexicon/2* folgt die Wortart, hier steht *cn* für *common noun*, *pn* für *proper noun* und *mn* für *measurement noun*. Das zweite Argument ist eine Liste, welche die wortspezifischen Angaben aufnimmt. Diese Angaben wiederum sind als einstellige komplexe Terme dargestellt, wobei jeder Term für eine bestimmte Information steht. Diese Terme nehmen als einziges Argument eine Liste[6]. Diese Liste kann leer sein oder Elemente enthalten, wobei sich jedoch bei mehreren Elementen jedes einzelne davon wiederum in einer Liste befindet (vgl. Beispiel 4.1).
Informationen, die nicht aus COMLEX gewonnen werden, sind **fett** gedruckt. Dabei handelt es sich entweder um optionale Angaben, die der Endbenutzer selbst spezifizieren soll, wie z.B. Kommentare (Abschnitt 4.1.1.4) oder Aliase (Abschnitt 4.1.1.3), oder um Informationen (z.B. *dimension* bei *measurement nouns*), die in ACELEX vorhanden sein sollten, aus COMLEX aber nicht extrahierbar sind (vgl. Abschnitt 3.2.5 und 4.1.1.9).

4.1.1. Die Nomen-Terme

Allen drei Nomen-Typen gemeinsam sind die Terme: *logical_relation/1*, *singular/1*, *plural/1*, *singular_aliases/1*, *plural_aliases/1* und *comment/1*. Ein *common noun*-Eintrag weist neben diesen Termen noch die Terme *type/1*, *gender/1*, *collective_noun/1* und *group/1* auf. Der Aufbau eines Eintrags für ein *proper noun* ähnelt demjenigen des *common nouns*. Der einzige Unterschied besteht darin, dass die Terme *collective_noun/1* und *group/1* fehlen. *Proper nouns* können keine *collective nouns* sein, da sie immer als spezifische Einheit angesehen werden. Sie können auch nicht einer Zählbarkeitsdomäne zugeordnet werden, sondern folgen bei der Verwendung von Artikeln ihren eigenen Regeln. Die *measurement nouns* erhalten zusätzlich

[6]Ich werde bei der Beschreibung der Terme meistens abgekürzt davon sprechen, dass *der Term einen Wert bzw. eine Wortform enthält* oder dass *ein Wert bzw. eine Wortform im Term steht, gespeichert ist, etc.*. Damit ist immer gemeint, dass der Wert bzw. die Wortform *in der Liste steht oder enthalten ist, die das Argument des Terms bildet*

zu den oben aufgezählten gemeinsamen Termen nur noch den Term *dimension/1*.
In den nachfolgenden Abschnitten werde ich die Terme der Reihe nach durchgehen und bei jedem darlegen, wo die entsprechende Information in COMLEX zu finden ist und wie diese Information in ACELEX umgesetzt wird.

4.1.1.1. logical_relation/1

Der Term *logical_relation/1* steht für die Nennform, d.h. das Wort, das in der Liste im Argument dieses Terms steht, ist das Wort, das mit diesem Eintrag beschrieben wird. In COMLEX gibt es das Merkmal :ORTH, das bei allen Einträgen vorhanden ist. Es enthält die Nennform – das Lemma – des Wortes[7]. Bei den Nomen ist es die Singularform, ausser das Nomen hat keinen Singular; dann wird die Pluralform verwendet. Der Wert dieses :ORTH-Merkmals liefert den Wert für den Term *logical_relation/1* in ACELEX.

4.1.1.2. singular/1 und plural/1

Der Term *singular/1* enthält die Singularform des Wortes. Hat ein Nomen keine Singularform, wie z.B. „acoustics“, ist die Liste im Argument des jeweiligen Terms leer. Analog dazu speichert der Term *plural/1* die Pluralform des Nomens. Existiert keine Pluralform, bleibt die Liste im Argument des *plural/1*-Terms leer.
Bei den meisten Nomen ist der Wert des COMLEX :ORTH-Merkmals gleichzeitig die Singularform. Gibt es jedoch ein Merkmal :SINGULAR, ist der Wert dieses Merkmals die korrekte Singularform, ausser der Wert des :SINGULAR-Merkmals ist *NONE*. In diesem Falle gibt es keine Singularform für das Wort und das Argument des Terms *singular/1* in ACELEX ist eine leere Liste. In den anderen Fällen wird die extrahierte Singularform in die Liste im Argument des *singular/1*-Terms geschrieben.

[7] Wie bereits in Abschnitt 3.2.3, Seite43 unterscheidet COMLEX nicht zwischen gleichgeschriebenen Wörtern der gleichen Wortklasse: „bank“ im Sinne von Ufer und Finanzinstitut sind in einem Eintrag zusammengefasst.

Im Eintrag eines regelmässigen Nomens gibt es weder ein :SINGULAR- noch ein :PLURAL-Merkmal. Die Pluralform des Wortes kann durch einfache Regeln generiert werden. Diese Regeln sind im [Macleod et al. 1998, S. 7f.] genau dokumentiert. Gibt es im Comlex-Eintrag jedoch ein :PLURAL-Merkmal, ist der Wert dieses Merkmals die Pluralform, ausser der Wert ist *NONE*. Hat es ein :SINGULAR-Merkmal mit dem Wert *NONE* und kein :PLURAL-Merkmal, heisst das, das keine Singularform existiert und der Wert des :ORTH-Merkmals entspricht dem Plural. In AceLex wird die auf diese Weise extrahierte Pluralform im Term *plural/1* gespeichert.

Ein *proper noun* hat normalerweise entweder eine Singularform („John“) oder eine Pluralform („The United States“). Das würde in AceLex dazu führen, dass entweder der Term *singular/1* oder der Term *plural/1* eine leere Liste als Argument nimmt. Bei den zeitanzeigenden *proper nouns* wie den Wochentagen und Monatsnamen kann aber trotz der „normalen“ singulären Form auch eine Pluralform existieren: Es ist z.B. möglich von den „last four Fridays“ zu reden oder etwas Ähnliches wie „In the previous years, the Aprils were really cold“ zu sagen. In Comlex sind nur solche zeitanzeigenden *proper nouns* aufgeführt, d.h. diese Nomen haben in AceLex sowohl im Term *singular/1* wie auch im Term *plural/1* einen Eintrag (vgl. Beispiel 4.2).

4.1.1.3. singular_aliases/1 und plural_aliases/1

Der Term *singular_aliases/1* kann Worte oder Abkürzungen (Aliase) aufnehmen, die in einem Ace-Text alternativ für die Singularform des im Eintrag beschriebenen Wortes eingesetzt werden können. Wenn in einem Ace-Text ein Satz mit dem Singular „kitten“ steht, wird es in der Paraphrase automatisch mit „cat“ ersetzt (vgl. Beispiel 4.1).

Der Term *plural_aliases/1* kann ebenfalls solche Aliase aufnehmen, diese können jedoch für die Pluralform des im Eintrag beschriebenen Wortes eingesetzt werden.

Steht in einem ACE-Text ein Satz mit dem Plural „kittens“, wird es in der Paraphrase automatisch mit „cats“ ersetzt (vgl. Beispiel 4.1).
Da COMLEX keine Synonyme liefert, bleibt die Liste im Argument dieses Terms leer. Der Benutzer soll hier nach seinen Bedürfnissen Wörter bzw. Abkürzungen einfügen können. Diese dürfen jedoch nicht noch als eigener Lexikoneintrag vorhanden sein, da sonst auf den separaten Lexikon-Eintrag zugegriffen wird und das Wort in der Paraphrase nicht als Alias erkannt wird. In Beispiel 4.2 wurde die Abkürzung „Apr.“ als Alias für „April“ definiert, in Beispiel 4.1 „kitten“ und „kittycat“ für „cat“. Wie in diesem Beispiel zu sehen, sind mehrere Wörter pro Eintrag erlaubt, jedes muss in einer eigenen Liste in die Liste des Termarguments eingefügt werden.

4.1.1.4. comment/1

Der Term *comment/1* soll dem Benutzer die Möglichkeit geben, einem Wort einen Kommentar beizufügen. Der Kommentar steht als ein PROLOG-Atom in der Liste im Argument des Terms. Der *comment/1*-Term steht dem Benutzer zur freien Verfügung. Hier wird nichts eingefügt.

4.1.1.5. type/1

Der Term *type/1* liefert den semantischen Typ des Nomens: Mögliche Werte sind *object*, *person* oder *time*[8]. COMLEX ist ein Lexikon, das sich auf die Syntax konzentriert[9]. Viele Merkmale sind daher rein syntaktisch motiviert und bringen für semantische Entscheidungen, wie z.B. der für ACELEX geforderten Einteilung, ob es sich beim Referenzobjekt des Wortes um eine Person, Zeit oder Objekt handelt, keinen Nutzen. Trotzdem können aus einigen Merkmalen semantische Aspekte herausgefiltert werden.

[8]Wie im Beispiel 4.1 zu sehen ist, werden Tiere den Objekten zugeordnet.
[9]Wie bereits erwähnt, ist der volle Name des Lexikons COMLEX *Syntax*

type: person COMLEX enthält ein Merkmal NHUMAN. Das Merkmal ist folgendermassen definiert (vgl. [ROHEN WOLFF et al. 1998, S. 13]):

> „A noun with the feature NHUMAN can occur as the head noun of relative clauses introduced by „who“ or „whom“.“

Der semantische Hintergrund wird schon durch den Namen des Merkmals ausgedrückt: Es bezeichnet eine Person. Die Liste im Argument des Terms *type/1* erhält den Wert „person“.
Es gibt noch ein Merkmal NTITLE, das in COMLEX nach folgender Regel bestimmt wird (vgl. [ROHEN WOLFF et al. 1998, S. 7]):

> „Here belong words that can occur as a title preceding names: [...]“[10]

Solche Wörter sind z.B.„doctor“ oder „professor“. Sie sollten eigentlich immer auch mit NHUMAN markiert sein, da eine Überprüfung ergeben hat, dass in COMLEX nur Nomen, die Personen bezeichnen, mit NTITLE gekennzeichnet sind. In der Spezifikation ist dieser Zusammenhang jedoch nicht festgehalten[11]. Die zusätzliche Markierung mit NHUMAN wäre eigentlich nötig, ist jedoch nicht konsequent durchgehalten. Damit die nicht mit NHUMAN markierten Nomen ebenfalls als Personen erkannt werden, werden auch alle mit NTITLE markierten Nomen in ACELEX mit *type* „person“ aufgeführt.
Ein drittes Merkmal, das Hinweise auf eine Person liefert, ist PREDNOUN. Die Definition in [ROHEN WOLFF et al. 1998, S. 12] sieht folgendermassen aus:

> „PREDNOUNs are a small subset of count nouns that can occur without a preceding article when they are predicated of the subject NP“

[10] In [ROHEN WOLFF et al. 1998, S. 7] werden hier Abkürzungen wie z.B. „Dr.“ oder „Mr.“ aufgezählt. Das ist aber nicht gemäss Spezifikation: Solche abgekürzten Titel sind in COMLEX als eigene „Wortart“ erfasst, d.h. sie werden nicht als Nomen klassifiziert. Das ist nur einer der häufigen Fehler in der Spezifikation für COMLEX.

[11] Was wäre z.B. mit dem in COMLEX nicht enthaltenen Übersetzung von „Alphatier“? Das wäre sozusagen ein NTITLE für Tiere.

Auf den erstem Blick fragt man sich natürlich, was dieses Merkmal mit der Einteilung in die semantische Kategorie der Personen zu tun hat. Es handelt sich hier jedoch um Wörter, die mit einem Titel gleichgesetzt werden können. Beispiele dafür sind: „president“, „director“, „judge“. Eigentlich sollten diese Nomen ebenfalls mit NHUMAN markiert sein, aber auch hier wurde das nicht konsequent durchgezogen. Die mit PREDNOUN markierten Nomen erhalten in AceLex den *type* „person“.
Zusätzlich zu den drei aufgeführten Einteilungskriterien, kann noch das bereits in Abschnitt 4.1, Seite 54 beschriebene Merkmal NAME als Unterscheidungskriterium dienen. Dieses Merkmal markiert eigentlich Namen von Personen, ist aber in Comlex Version 3.0 nicht korrekt angewendet. Trotzdem lässt sich aus diesem Merkmal ableiten, dass es sich bei solchermassen markierten Nomen um Personen handelt. Sie erhalten in AceLex den *type* „person“.

type: time Um zu bestimmen, ob ein Nomen eine Zeit ausdrückt, gibt es in Comlex zwei Merkmale, die dazu Informationen liefern: NTIME1 und NTIME2. NTIME1 ist folgendermassen definiert (vgl. [Rohen Wolff et al. 1998, S. 14]):

> „A noun denoting time that cannot occur alone as a sentence adjunct but does occur in this capacity with a premodifying ATIMETAG[12] adjective.“

Ein Beispiel für ein NTIME1-Nomen wäre: „week“. Es ist möglich zu sagen: „He went last week„ („last“ = ATIMETAG), aber der Satz wird ungrammatikalisch, wenn „week“ allein steht: „*He went (a) week“. Diese Eigenheit des syntaktischen Verhaltens ist für AceLex jedoch bedeutungslos. Die wichtige Information steht zu Beginn des Satzes: Ein Nomen, das die Zeit anzeigt. In AceLex sind mit NTIME1 gekennzeichnete Nomen vom *type* „time“.
NTIME2 bezeichnet ein zeitanzeigendes Nomen, das allein als Adjunkt stehen könnte (vgl. [Rohen Wolff et al. 1998, S. 15]):

[12] Ein Comlex-Merkmal bei Adjektiven.

„A noun has the feature NTIME2 if it can occur alone as a sentence adjunct.“

Ein Beispiel hierzu wäre „He left Friday“. Auch NTIME2 markierte Nomen erhalten in COMLEX den Typ *time*.

type: object Für die Einteilung zu den Objekten gibt es in COMLEX keine Merkmale. In ACELEX werden alle Nomen, die nicht schon mit *type* „person“ oder „time“ markiert sind, als „object“ eingeteilt. Das führt dazu, dass auch abstrakte Konzepte als Objekte definiert werden.

4.1.1.6. gender/1

Der Term *gender/1* enthält Informationen über das grammatische Geschlecht des Wortes. Mögliche Werte sind: *masculine*, *feminine*, *neuter* oder *masculine/feminine*. Weitaus am häufigsten werden die Werte *neuter* und *masculine/feminine* auftauchen.
Im Englischen hat das Geschlecht nur bei Personen[13] sichtbaren Einfluss auf die Sätze, indem z.B. bei anaphorischen Referenzen das richtige Personalpronomen gewählt werden muss. Männliche Personen müssen mit „he“ referenziert werden, weibliche mit „she“. Es gibt folgende Arten von Personenbezeichnungen, die die Verwendung des korrekten Personalpronomens verlangen:

- Es gibt Personenbezeichnungen für weibliche bzw. männliche Lebewesen, die eine morphologische Geschlechtsmarkierung tragen, wie z.B. „policewoman“ bzw. „policeman“ oder „she-devil“, „heroine“ bzw. „widower“, „bridegroom“. Sie müssen mit „she“ bzw. „he“ referenziert werden.

- Es existieren auch Personenbezeichnungen für weibliche bzw. männliche Lebewesen, die keine morphologischen Geschlechtsmarkierung tragen: „mother“

[13] teilweise werden in der englischen Sprache auch „höhere“ Tiere, wie z.B. Katzen und Hunde mit „she“ bzw. „he“ referenziert. In ACELEX sind sie jedoch „Objekte“ und neutrum.

bzw. „father“ oder „nun“ bzw. „monk“. Auch sie müssen mit „she“ bzw. „he“ referenziert werden.

- Es gibt viele Personenbezeichnungen, die für beide Geschlechter gebraucht werden, jedoch bei Referenzen mit dem entsprechenden Personalpronomen ersetzt werden müssen: „The artist is very young, but *she* is already famous“. Dies kann jedoch nicht ohne den Kontext entschieden werden und das *gender* kann nicht im Lexikon festgelegt werden.

- Einige wenige Personenbezeichnungen können für beide Geschlechter gebraucht werden und mit „he“ oder „it“ bzw. „she“ oder „it“ ersetzt werden: Beispiele sind „baby“ und „child“. Auch dies kann ohne Kontext nicht entschieden werden.

- Sammelnamen wie z.B. „committee“ oder „aristocracy“ werden mit „it“, bei pluralischer Verwendung mit „they“ referenziert. Das Geschlecht ist in diesen Fällen nicht von Bedeutung.

Comlex liefert keinerlei Informationen über das Geschlecht. Nomen, die den *type: person* erhalten, werden vorläufig mit *gender: masculine/feminine* gekennzeichnet. Die korrekte Einteilung müsste entweder von Hand[14] (es sind nicht sehr viele Wörter) vorgenommen werden oder es muss auf ein anderes Lexikon zugegriffen werden.
Das Geschlecht von Eigennamen ist nicht einfach zu bestimmen. Oft werden z.B. Länder und Schiffe als weiblich angesehen: „From this map of England you can see that she/it lies north of the 50th parallel“, „The Titanic sank on her/its maiden voyage“[15] In AceLex erhalten aber nur Personen ein anderes Geschlecht als *neuter*.

[14]Die Personenbezeichnungen, die eine morphologische Geschlechtsmarkierung tragen, könnten ev. automatisch erkannt werden. Der Aufwand würde aber den Nutzen übersteigen: Die Einteilung von Hand wäre effizienter.

[15]Beispiele von der Grammatiksammlung des Seminars für Englische Philologie der Universität Mainz `www.uni-mainz.de/FB/Philologie-II/fb1413/english_grammar/chapter_6_5_2.htm`.

4.1.1.7. collective_noun/1

Mit einem boole'schen Wert (*yes/no*) gibt der Term *collective_noun/1* an, ob das Nomen ein *collective noun* ist. Wie in Abschnitt 3.1.1 beschrieben, ist ein *collective noun* ein singuläres Nomen, das sowohl als Subjekt von im Singular stehenden Verben, wie auch als Subjekt von im Plural stehenden Verben fungieren kann. In Comlex sind diese Nomen mit dem Merkmal AGGREGATE getagt.
In Comlex existiert ausserdem ein Merkmal NCOLLECTIVE, das Nomen bezeichnet, die in ihrer Singularform als Subjekt oder Objekt bei den in Abschnitt 3.1.2 beschriebenen „kollektiven" Verben wie „gather", „accumulate" etc. dienen können. Diese Verben verlangen normalerweise ein Subjekt bzw. Objekt (falls sie transitiv verwendet werden) im Plural, ein zusammengesetztes (*conjoined*) Subjekt bzw. Objekt oder ein *collective noun*: „The boys (*plural*) / John and Mary (*conjoined*) / the club (*collective*) gathered.". Es gibt aber bestimmte Nomen, die ebenfalls als Subjekt oder Objekt für diese Verben dienen können. Beispiele solcher Nomen sind: „dust", „knowledge", „fortune" und „alcohol". Der Satz „The shelf will gather dust" ist korrekt, „*The shelf will gather book" jedoch nicht. Oftmals sind NCOLLECTIVE markierte Nomen unzählbar, also *mass nouns* („dust"). Einige sind jedoch auch Nomen, die sowohl zählbar wie auch unzählbar verwendet werden können, wie z.B. „debt", „fortune". In Comlex ist das Merkmal NCOLLECTIVE folgendermassen definiert:

> Nouns which can occur in their singular form as the subject or object of a verb with the feature VCOLLECTIVE bear the feature NCOLLECTIVE. VCOLLECTIVE verbs ordinarily require a plural noun phrase in these positions.

Diese NCOLLECTIVE Nomen werden in AceLex wie die AGGREGATE markierten Nomen den *flag yes* im Term *collective_noun/1* erhalten. Auf diese Weise kann die in Comlex beschriebene Beziehung zwischen VCOLLECTIVE getagten Verben und NCOLLECTIVE (bzw. AGGREGATE) markierten Nomen erhalten werden. Je-

doch kann dann nicht mehr zwischen eigentlichen *collective nouns* (AGGREGATE) und diesen COMLEX-spezifischen NCOLLECTIVE Nomen unterschieden werden. Das ist aber in ACELEX nicht nötig. Die Eigenschaften, die *collective nouns* auszeichnen (vgl. Abschnitt 3.1.1), spielen in der Sprache ACE keine Rolle: Ein Nomen im Singular nimmt immer ein Verb im Singular. Die *collective nouns* sollten nur aufgrund ihrer Verbindung mit den „kollektiven“ Verben in ACELEX ausgezeichnet werden.

4.1.1.8. group/1

Der Term *group/1* teilt die Nomen in *countable nouns* und *mass nouns* ein. Nomen, die auf beide Arten verwendet werden können, sollen zwei Einträge erhalten, einen mit *group([countable])* und einen mit *group([mass])* (vgl. Abschnitt 3.1.1).
Zählbare Nomen sind in COMLEX mit COUNTABLE markiert und werden auch mit *countable* als Wert im Term *group/1* markiert. *Mass nouns* haben keine Pluralform, d.h. sie haben ein Merkmal :PLURAL *NONE* und sie sind nicht mit COUNTABLE markiert. Diese Nomen erhalten den Wert *mass* im Term *group/1*. *Countable mass nouns*, die je nach Kontext als zählbares oder unzählbares Nomen verwendet werden können, sind in COMLEX nicht speziell markiert: Ist das Nomen weder mit COUNTABLE, noch mit PLURAL: *NONE* markiert, ist es ein *countable mass noun*. Wie oben erwähnt, werden in ACELEX die *countable mass nouns* zweimal aufgeführt, je einmal als *countable noun* und einmal als *mass noun*.

4.1.1.9. dimension/1

Der Term *dimension/1* soll die Einheit, die durch das *measurement noun* beschrieben wird, einer Dimension zuordnen. Z.B. sollen „meter“, „inch“ etc. der Dimension „length“ zugeordnet werden, „kilogramm“ der Dimension „weight“ etc..
COMLEX markiert Nomen wie „length“ und „weight“ zwar mit einem Merkmal NSCALE, liefert aber keine Hinweise, welche Einheiten (mit NUNIT markiert, vgl.

Abschnitt 4.1), welche Dimensionen ausdrücken. Diese Information muss entweder aus einem anderen Lexikon extrahiert oder von Hand nachträglich eingefügt werden. In dieser Version von AceLex steht im Term *dimension/1* eines *measurement nouns* der Wert „dimension“ als Platzhalter.

4.2. Verben

Wie bereits in Abschnitt 3.1.2 erwähnt, werden Verben in drei Untergruppen eingeteilt: Die intransitiven, die transitiven und die ditransitiven Verben.
Comlex stellt sehr detaillierte Subkategorisierungsinformationen zur Verfügung. Die Verben sind nicht einfach in den oben genannten drei Kategorien eingeteilt, sondern es sind total 124 Subkategorisierungsstrukturen definiert. AceLex verlangt nur einen Bruchteil dieser Information.
Die Subkategorisierungsstrukturen sind einfach verständlich benannt: Die Konstituenten werden der Reihe nach, wie sie im Satz erscheinen, aufgezählt. Nimmt z.B. ein Verb eine Nominalphrase als Komplement (z.B. „love“), ist es in Comlex mit NP markiert. Ist es ein ditransitives Verb und verlangt zwei Nominalphrasen (z.B. „He asked me my name“), ist es mit NP-NP gekennzeichnet. Ein Verb kann mehrere Subkategorisierungseinträge (*frames*) haben. Das Verb „love“ hat neun *frames*, neben dem schon erwähnten NP-*frame*, z.B. noch THAT-S für Verben, die ein Satzkomplement nehmen, das durch ein obligatorisches „that“ eingeleitet werden muss („He loves that she always says what she thinks“). Ich möchte hier nicht weiter auf die Subkategorisierungsstrukturen eingehen, sondern verweise für Details auf [Macleod et al. 1998] und [Rohen Wolff et al. 1998]. Tabelle 4.2 listet auf, welche Kategorisierung in AceLex aus welchen Subkategorisierungsstrukturen von Comlex gewonnen wird.
In Comlex haben fast alle Verben eine der in der Tabelle 4.2 aufgeführten Subkategorisierungsstruktur und können einer AceLex-Kategorie zugeteilt werden. Die 21 Ausnahmen sind: „be“, „become“, „behoove“, „behove“, „beseem“, „chance“, „con-

Comlex		AceLex
intransitive verb	INTRANS	„He worked“
	INTRANS-RECIP	„They met“, „*John met“
	PART	„They lined up“
transitive verb	NP	„He loves her“
	PP	„He depends on her“
	PART-NP	„He looked the number up“
	PART-PP	„She stood up for him“
ditransitive verb	NP-NP	„He asked me my name“
	NP-PP	„They attributed the painting to Masaccio“
	PP-PP	„They apologized to him for their behavior“
	PART-NP-PP	„He split the project up into three parts“
	NP-TO-NP	„She gave a big kiss to her mother“, oder „She gave her mother a big kiss“
	NP-FOR-NP	„She bought a book for him“, oder „She bought him a book“

Tabelle 4.2.: Subkategorisierung: Von Comlex zu AceLex

jecture“, „construe“, „covenant“, „deem“, „dub“, „endeavor“, „endeavour“, „end up“, „entrance “, „have to“, „opine“, „seem“„term“, „trend“ und „turn out“. Diese Verben wurden nicht in AceLex aufgenommen, da sie Komplemente verlangen, die in Ace nicht zugelassen sind.

Die drei Beispiele 4.4, 4.5 und 4.6 zeigen je einen Eintrag eines intransitiven (*iv*), eines transitiven (*tv*) und eines ditransitiven Verbs (*dv*), wie sie in AceLex aussehen. **Fett** gedruckt sind wiederum Informationen, die in dieser Version von AceLex nicht geliefert werden. Bei den Verben ist das hauptsächlich der nicht aus Comlex extrahierbare *type*. Ausserdem ist die Darstellung der Verben im Term *logical_relation/1* nicht optimal (vgl. Abschnitt 4.2.1.1).

lexicon(iv, [logical_relation([line_**up**]), third_singular([lines]), third_plural([line]), third_singular_aliases([]), third_plural_aliases([]), type([**event**]), phrasal_particle([up]), collective_subject([no]), comment([])]).

Beispiel 4.4: AceLex-Eintrag intransitives Verb

lexicon(tv, [logical_relation([love]), third_singular([loves]), third_plural([love]), third_singular_aliases([]), third_plural_aliases([]), type([**state**]), phrasal_particle([]), collective_object([no]), complement_direct([noun_phrase]), direct_preposition([]), comment([])]).

Beispiel 4.5: AceLex-Eintrag transitives Verb

lexicon(dv, [logical_relation([give]), third_singular([gives]), third_plural([give]), third_singular_aliases([]), third_plural_aliases([]), type([**event**]), phrasal_particle([]), collective_object([no]), complement_direct([noun_phrase]), complement_indirect([noun_phrase]), direct_preposition([]), indirect_preposition([]), comment([])]).

Beispiel 4.6: AceLex-Eintrag ditransitives Verb

4.2.1. Die Verb-Terme

Folgende Terme sind in allen Verb-Einträgen zu finden: *logical_relation/1*, *third_singular/1*, *third_plural/1*, *third_singular_aliases/1*, *third_plural_aliases/1*, *type/1*, *phrasal_particle/1* und *comment/1*.

Intransitive Verben verlangen keine Komplemente und erhalten deshalb nur noch den Term *collective_subject/1*.

Transitive Verben verlangen ein Objekt als Komplement, ohne das sie nicht vollständig sind: „*She takes [?]“, „She takes a piece of cake“. Wie in Abschnitt 2.4.2.7 beschrieben, sind in ACE nur Nominalphrasen und Präpositionalphrasen zugelassen. Der Eintrag eines transitiven Verbs besteht neben den allen Verben gemeinsamen Termen aus den Termen *collective_object/1*, *complement_direct/1* und *direct_preposition/1*.

Ditransitive Verben nehmen zwei Komplemente. Die folgenden Terme bilden mit den allen Verben gemeinsamen Termen den Eintrag eines ditransitiven Verbs: *collective_object/1*, *complement_direct/1*, *complement_indirect/1*, *direct_preposition/1* und *indirect_preposition/1*.

4.2.1.1. logical_relation/1

Wie bei den Nomen und allen anderen Wortarten steht die Nennform im Term *logical_relation/1*. Bei den Verben dient der Wert des Merkmals :ORTH als Quelle für die Nennform, wobei es sich immer um die Infinitivform handelt. COMLEX unterscheidet im ORTH-Merkmal nicht zwischen normalen Verben und *phrasal* bzw. *prepositional verbs* (vgl. Abschnitt 3.1.2). D.h. es gibt nur einen Eintrag in COMLEX für „look“, obwohl das normale Verb „look“, das *phrasal verb* „look up“ und das *prepositional verb* „look after“ verschiedene Bedeutungen haben. Ob das Verb mit einem verbalen Partikel ein *phrasal verb* werden kann oder ob es eine bestimmte Präposition selektiert, ist erst im :SUBC-Merkmal festgehalten. In ACELEX wird für jedes dieser Verben ein eigener Eintrag stehen. In der aktuellen Version von ACELEX

unterscheiden sich die Einträge jedoch nicht im Term *logical_relation/1*. Jeder der Einträge von „look“ hat diese Form als *logical_relation*. Sie unterscheiden sich aber darin, dass sie z.B. im Term *phrasal_particle/1* eine leere Liste (wie „look“ und „look after“) oder einen Wert („up“ bei „look sth up“) nehmen. Das kann aber auf einfache Weise erweitert werden, indem der im Term *phrasal_particle/1* (vgl. Abschnitt 4.2.1.6 stehende verbale Partikel an die Form im *logical_relation/1*-Term angehängt wird. Auch die in den Termen *direct_preposition/1* und *indirect_preposition/1* (vgl. Abschnitt 4.2.1.10 bzw. 4.2.1.11) gespeicherten Präpositionen könnten auf diese Weise an die im Term *logical_relation/1* stehende Verbform angehängt werden. Da die Einträge für die Verarbeitung im ATTEMPTO-System jedoch sowieso in ein anderes Format gebracht werden müssen, wird das in dieser Arbeit nicht durchgeführt.

4.2.1.2. third_singular/1

Im Term *third_singular/1* steht die dritte Person Singular des Verbs und zwar im Präsens (*simple present tense*) Indikativ Aktiv.
Für regelmässige Verben muss die Singularform durch eine in [MACLEOD et al. 1998, S. 7] definierte Regel generiert werden. Einige unregelmässige Verben sind in COMLEX mit dem Merkmal :3PSING markiert. Es enthält als Wert die gesuchte Singularform der dritten Person Präsens. In ACELEX sind die durch die Regeln generierten oder die aus dem Merkmal :3PSING extrahierten Singularformen im Term *third_singular/1* gespeichert.

4.2.1.3. third_plural/1

Im Term *third_plural/1* steht die dritte Person Plural des Verbs in der *simple present tense* im Indikativ Aktiv.
Bei allen Verben ausser „be“ dient der Wert des :ORTH-Merkmal als Quelle für die Pluralform, die Form des Infinitivs ist die gleiche wie die der dritten (und ersten und zweiten) Person Plural. „Be“ hat als einziges Verb eine Pluralform, die sich

vom Infinitiv unterscheidet. Das Merkmal :PLURAL liefert diese Pluralform: „are“. Die Pluralformen sind in AceLex im Term *third_plural/1* enthalten.

4.2.1.4. third_singular_aliases/1 und third_plural_aliases/1

In den Termen *third_singular_aliases/1* und *third_plural_aliases/1* können die Singularform bzw. die Pluralform von Verben erfasst werden, die in einem Ace-Text anstelle des im Eintrag definierten Verbs stehen können. Dabei sollte es sich um ein Verb handeln, das zur gleichen Untergruppe gehört, d.h. bei einem transitiven Verb sollte das Alias-Verb ebenfalls transitiv verwendet werden können. Wie bei den Nomen (vgl. Abschnitt 4.1.1.3) bleiben die Listenargumente dieser Terme leer, bis der Benutzer seine eigenen Abkürzungen oder „Synonyme“ definiert.

4.2.1.5. type/1

Im Term *type/1* ist festgehalten, ob ein Verb einen *state* oder ein *event* ausdrückt (vgl. Abschnitt 2.4.3).
Comlex liefert keine Hinweise darüber, ob das Verb einen *state* oder ein *event* beschreibt. Diese Information muss in einer weiterführenden Arbeit aus einem anderen Lexikon gewonnen werden. In der in dieser Arbeit generierten Version von AceLex enthält der Term *type/1* den informationsleeren Füllwert „eventuality“.

4.2.1.6. phrasal_particle/1

Handelt es sich bei dem Eintrag um denjenigen eines *phrasal verbs* (vgl. Abschnitt 3.1.2), nimmt der Term *phrasal_particle/1* dessen verbalen Partikel auf.
Verben, die einen verbalen Partikel nehmen können (*phrasal verbs*) sind in Comlex mit einer Subkategorisierungsstruktur markiert, die mit dem Schlüsselwort „PART“ beginnt. Für AceLex sind die folgenden Strukturen relevant: PART, PART-NP, PART-PP und PART-NP-PP (vgl. Tabelle 4.2). Diese Strukturen besitzen ein Merkmal :ADVAL, das die für das jeweilige Verb möglichen verbalen Partikel aufführt.

In AceLex soll im Term *phrasal_particle/1* jeweils nur ein Wort stehen. Deshalb wird für jeden in der ADVAL-Liste stehenden verbalen Partikel ein eigener Eintrag generiert, die sich bis auf den im Term *phrasal_particle/1* stehenden Partikel nicht unterscheiden. Ein Beispiel: „look“ ist nebst anderen Subkategorisierungsstrukturen mit PART-NP gekennzeichnet. Mitgeliefert wird dabei eine ADVAL-Liste mit den verbalen Partikeln „over“ und „up“. Es geht hier also um die *phrasal verbs* „look something over“ und „look something up“. „look over“ erhält den in Beispiel 4.7 dargestellten Eintrag in AceLex:

lexicon(tv, [logical_relation([look]), third_singular([looks]), third_plural([look]), third_singular_aliases([]), third_plural_aliases([]), type([]), phrasal_particle([over]), collective_object([no]), complement_direct([noun_phrase]), direct_preposition([]), comment([])]).

Beispiel 4.7: AceLex-Eintrag des transitiven *phrasal verb* „look over“

„look up“ erhält einen anderen beinahe identischen Eintrag: Die Liste im Term *phrasal_particle/1* enthält den Partikel „up“ statt „over“. Würde man die in Abschnitt 4.2.1.1, Seite 72 beschriebene Umformung der im Term *logical_relation/1* stehenden Verb-Form durchführen, würde die neue Nennform für die beiden erwähnten *phrasal verbs* folgendermassen aussehen: „look_over“ bzw. „look_up“.
Etwas komplizierter wird es, wenn das Verb neben dem verbalen Partikel zusätzlich eine Präposition selektiert. Dies ist bei den Subkategorisierungsrahmen PART-PP und PART-NP-PP der Fall. In diesen Fällen wird für jede Kombination zwischen verbalem Partikel und Präposition ein Eintrag generiert. „stand“ hat in Comlex einen Subkategorisierungs-*frame* PART-PP, der in Beispiel 4.8 dargestellt ist. Aus dieser Subkategorisierungsstruktur werden 49 Einträge in AceLex generiert: Den ersten mit „across“ im Term *phrasal_particle/1* und „among“ im Term *direct_preposition/1*, dann mit „across“ im Term *phrasal_particle/1* und „over“ im Term *direct_preposition/1*, etc. bis alle Kombinationen abgearbeitet sind. Dabei kann eine Übergenerierung nicht ausgeschlossen werden, nicht jede Kombination *verb +*

stand: (PART-PP :ADVAL („across“
„apart“
„away“
„off“
„up“
„in“
„out“)
:PVAL („among“
„over“
„against“
„for“
„from“
„to“
„in“))

Beispiel 4.8: Subkategorisierungseintrag PART-PP von „stand“

verbal particle + preposition liefert eine sinnvolle Konstruktion. In COMLEX wird jedoch nicht angegeben, welche Kombinationen zulässig sind. Die Übergenerierung sollte keine schwerwiegenden Konsequenzen haben: Normalerweise wird die durch Übergenerierung entstandene inkorrekte Form im Text der Spezifikation nicht auftauchen. Auch für das Verb „stand“ in Beispiel 4.8 könnte man die Form im Term *logical_relation/1* (wie am Beispiel 4.7 mit „look“ gezeigt) umformen. Dabei würden neue Nennformen gebildet, wie z.B. die beiden durch Übergenerierung entstandenen Formen „stand_across_among“ und „stand_across_over“ oder die korrekte Struktur „stand_up_for“. In solchen Strukturen steht der verbale Partikel immer vor der Präposition.

4.2.1.7. comment/1

Der Term *comment/1* steht bei allen Wortarten dem Benutzer zur Verfügung, um Kommentare und Hinweise einzufügen.

4.2.1.8. collective_subject/1

Der Term *collective_subject/1* zeigt mit einem boole'schen Wert an, ob das Verb ein „kollektives" intransitives Verb ist, im Sinne der Beschreibung in Abschnitt 3.1.2. In COMLEX gibt es für die „kollektiven" Verben ein Merkmal VCOLLECTIVE. Die damit markierten Verben erhalten im Term *collective_subject/1* den Wert *yes*. Mit VCOLLECTIVE markierte intransitive Verben verlangen, dass das Subjekt im Plural, zusammengesetzt (*conjoined*) oder ein Nomen sein muss, das in COMLEX mit AGGREGATE oder NCOLLECTIVE markiert ist. Diese Nomen sind in ACELEX im Term *collective noun/1* (vgl. Abschnitt 4.1.1.7) durch den *flag yes* gekennzeichnet.

4.2.1.9. collective_object/1

Mit einem boole'schen Wert gibt der Term *collective_object/1* an, ob das Verb ein „kollektives" transitives oder ditransitives Verb ist, wie es in Abschnitt 3.1.2 beschrieben wurde. Die „kollektiven" (di-)transitiven Verben verlangen ein Objekt, das eine Menge anzeigt. Diese Verben sind – wie die „kollektiven" intransitive Verben auch – in COMLEX mit VCOLLECTIVE markiert. In ACELEX wird im Term *collective_object/1* mit einem boole'schen Wert angegeben, ob das Verb in diese Kategorie fällt.

4.2.1.10. complement_direct/1 und direct_preposition/1

Im Term *complement_direct/1* ist festgehalten, welche der beiden erlaubten Komplementarten das transitive oder ditransitive Verb als erstes Komplement nimmt: eine Nominalphrase oder eine Präpositionalphrase. Falls dieses Komplement eine Präpositionalphrase ist, speichert der Term *direct_preposition/1* die spezifische, vom Verb selektierte Präposition. Nimmt das Verb hingegen eine Nominalphrase, ist die Liste leer.

Transitive Verben Mit NP markierte Verben sind normale transitive Verben und verlangen eine Nominalphrase als Komplement: „She takes a piece of cake“. Verben, die mit PART-NP markiert sind, sind transitive *phrasal verbs*, die eine Nominalphrase verlangen: „He looked the number up“. Der verbale Partikel ist im Term *phrasal_particle/1* (vgl. Abschnitt 4.2.1.6) definiert.
Verben, die mit PP oder PART-PP markiert sind, selektieren eine Präpositionalphrase eingeleitet durch eine bestimmte Präposition. Sie sind sogenannte *prepositional verbs* (vgl. Abschnitt 3.1.2): „He depends on her“. Die vom Verb verlangte Präposition ist im Term *direct_preposition/1* festgehalten. PART-PP gekennzeichnete Verben sind *phrasal verbs*, die eine Präpositionalphrase verlangen: „She stood up for him“. Die möglichen Präpositionen werden den Verben in Comlex mit einer PVAL-Liste mitgegeben, im Beispiel 4.8 sieht man die Darstellung einer solchen Subkategorisierungsstruktur. Bei mit PP markierten Verben wird für jede Präposition in dieser Liste ein eigener Eintrag in AceLex generiert. Diese Einträge sind bis auf die im Term *direct_preposition/1* stehende Präposition identisch. Bei mit PART-PP gekennzeichneten Verben wird, wie in Abschnitt 4.2.1.6 beschrieben, für jede Kombination zwischen verbalem Partikel und Präposition ein Eintrag in AceLex generiert.

Ditransitive Verben In Comlex sind die Namen der Subkategorisierungsstrukturen so gewählt, dass die unmarkierte, d.h. die am häufigsten vorkommende Satzstellung beschrieben wird: Das direkte Objekt steht vor dem indirekten Objekt. Ditransitive Verben, deren Subkategorisierungsrahmen mit NP oder mit PART-NP beginnen, nehmen als erstes Objekt eine Nominalphrase. Das sind die Strukturen NP-NP, NP-PP, PART-NP-PP, NP-TO-NP und NP-FOR-NP. Im Argument des Terms *direct_preposition/1* bleibt die Liste leer.
Die mit PP-PP markierten Verben hingegen verlangen in beiden Komplementen eine Präpositionalphrase. Bei dieser Subkategorisierungsstruktur kann nicht entschieden werden, welche Präpositionalphrase zuerst kommt. Darum ist in Comlex nur eine

PVAL-Liste für die vom Verb selektierten Präpositionen beider Präpositionalphrasen definiert. Die Reihenfolge der Präpositionen in der Liste ist nicht aussagekräftig, es müssen einfach mindestens zwei[16] sein. Klarer wird das an den Beispielen 4.9 und 4.10.

apologize: (PP-PP :PVAL („for“ „to“))

Beispiel 4.9: Subkategorisierungseintrag PP-PP von „apologize“

differ: (PP-PP :PVAL („to“
„in“
„about“
„on“
„with“
„from“
„by“))

Beispiel 4.10: Subkategorisierungseintrag PP-PP von „differ“

Im ersten Beispiel sind genau zwei Präpositionen in der PVAL-Liste: „for“ und „to“. Damit lassen sich Sätze bilden wie „They apologized to him for their behavior“ oder umgekehrt „They apologized for their behavior to him“. In AceLex wird je ein Eintrag für diese beide Möglichkeiten generiert. Einmal steht „for“ im Term *direct_preposition/1* und „to“ im Term *indirect_preposition/1*, einmal „to“ in *direct_preposition* und „for“ in *indirect_preposition/1*. Die Möglichkeit, dass zweimal „for“ oder zweimal „to“ gebraucht werden, wird ausgeschlossen.
Das Beispiel 4.10 ist nicht mehr so übersichtlich lösbar. Es sind mehr als zwei Präpositionen in der PVAL-Liste. Sätze wie „Mary differed with John on that question“ und „Rosalie differed from Janet in her style of dressing“ sind daraus ableit-

[16] In der Spezifikation [Rohen Wolff et al. 1998, S. 76] ist festgehalten, dass mindestens zwei Präpositionen in der Liste enthalten sein müssen. Jedoch kann in einem Falle auch nur ein Wert in der Liste stehen: Wenn der Wert „p-dir“ ist. „P-dir“ ist eine Art „Meta“-Präposition, die für eine ganze Gruppe von Präpositionen steht, die Richtungen anzeigen.

bar. Dabei ist aber nicht jede Präposition mit jeder kombinierbar: „*Mary differed with John from Janet“. Die gültigen Kombinationen sind in COMLEX nicht gekennzeichnet. Trotzdem wird für jede Kombination von Präpositionen ein Eintrag in ACELEX generiert, jedoch mit der Einschränkung, dass nicht zweimal die gleiche Präposition in den Termen *direct_preposition/1* und *indirect_preposition/1* (vgl. Abschnitt 4.2.1.11) steht. Die dabei stattfindende Übergenerierung wird in Kauf genommen.

4.2.1.11. complement_indirect/1 und indirect_preposition/1

Im Term *complement_indirect/1* ist festgehalten, ob das zweite Objekt eines ditransitiven Verbs eine Nominalphrase oder eine Präpositionalphrase ist. Ist das zweite Komplement eine Präpositionalphrase, speichert der Term *indirect_preposition/1* die Präposition dieses zweiten Objekts.
Die Subkategorisierungsstruktur NP-NP[17] nimmt eine Nominalphrase als zweites Objekt und eine leere Liste im Term *indirect_preposition/1.*
NP-PP, PART-NP-PP, NP-TO-NP, NP-FOR-NP und das schon im Abschnitt 4.2.1.10, Seite 77 beschriebene PP-PP verlangen eine Präpositionalphrase. Speziell sind NP-TO-NP und NP-FOR-NP: Mit diesen Strukturen sind ditransitive Verben markiert, die sowohl die Struktur *Subject-NP + verb + NP_1 + PP_2 (mit „for“/„to“)* („She gave a big kiss to her mother“) wie auch *Subject-NP + verb + NP_2 (ohne preposition) + NP_1* („She gave her mother a big kiss“) haben können. Sie sind in COMLEX nicht noch zusätzlich mit NP-NP markiert, es steht ein Subkategorisierungsrahmen für zwei verschiedene Strukturen. In ACELEX werden beide Möglichkeiten aufgeführt. Es werden zwei Einträge generiert: Einer mit zwei Nomi-

[17] Diese Verben sind eigentlich keine ditransitiven Verben, sondern werden als *factitive verbs* bezeichnet. *Factitive verbs* sind *transitive verbs* die ein *direct object* nehmen, das durch ein sogenanntes *objective complement* modifiziert wird. Das *direct object* erfährt durch das *objective complement* eine Veränderung: „The faculty elected Dogsbreath the new Academic Dean“ (Dogsbreath – *direct object* und „Academic Dean“ – *objective complement*), „He made the water wine“ („water“ – *direct object* „wine“ – *objective complement*). In ACELEX werden diese Verben aber zu den ditransitiven Verben gezählt.

nalphrasen als Komplemente[18] und einer, der im Term *complement_indirect/1* eine Präpositionalphrase nimmt. In diesem Fall wird im Term *indirect_preposition/1* bei NP-TO-NP „to “ stehen, bei NP-FOR-NP „for“[19].
Ist das Verb mit NP-PP markiert, wird für jede in der PVAL-Liste stehende Präposition ein eigener Eintrag generiert. Bei der Struktur PART-NP-PP wird, wie in Abschnitt 4.2.1.6 dargestellt, für jede Kombination zwischen verbalen Partikeln in der ADVAL-Liste und Präpositionen in der PVAL-Liste ein eigener Eintrag erstellt.

4.3. Adjektive

Die Adjektive werden nicht in Untergruppen aufgeteilt. In Beispiel 4.11 findet sich der Eintrag des Adjektivs „cold“.

lexicon(adj, [logical_relation([cold]), positive([cold]), comparative([colder]), superlative([coldest]), positive_aliases([]), comparative_aliases([]), superlative_aliases([]), complement([no_complement]), complementing_preposition([]), comment([])]).

Beispiel 4.11: AceLex-Eintrag Adjektiv

4.3.1. logical_relation/1

Im Term *logical_relation/1* soll die Nennform stehen. Die Nennform wird aus dem Comlex-Merkmal :ORTH herausgelesen. Dabei handelt es sich um die Positiv-Form, das heisst die ungesteigerte „normale“ Form des Adjektivs. Einige Adjektive verlangen eine Präpositionalphrase mit einer bestimmten Präposition als Komplement. Wie bei den Verben (vgl. Abschnitt 4.2.1.1) könnte man diese Abhängigkeit

[18] Bei Verben, die gleichzeitig auch mit NP-NP markiert sind, kann dies zu identischen Einträgen führen, die in AceLex jedoch unterdrückt werden.
[19] Dies kann bei Verben, die zusätzlich mit NP-PP markiert sind und in ihrer Liste von Präpositionen „to“ bzw. „for“ enthalten, zu identischen Einträgen führen. Diese werden in AceLex unterdrückt.

bereits im Term *logical_relation/1* darstellen, indem die Präposition an die normale Adjektiv-Form angehängt würde. In dieser Arbeit wird diese einfache Mutation jedoch nicht gemacht: „fond of“ ist im Term *logical_relation/1* nur mit „fond“ referenziert.

4.3.2. positive/1

Der Term *positive/1* enthält die Positiv-Form, d.h. die Grundform des Adjektivs. Die Positiv-Form ist in COMLEX im Merkmal :ORTH gespeichert. Dieser Wert wird für den Term *positive/1* übernommen.

4.3.3. comparative/1 und superlative/1

Der Term *comparative/1* enthält die Komparativ-Form des Adjektivs, falls es steigerbar ist. Ist es nicht steigerbar, bleibt die Liste im Argument des Terms leer. Die Superlativ-Form des Adjektivs ist im Term *superlative/1* gespeichert. Ist das Adjektiv nicht steigerbar, enthält der Term eine leere Liste.
In COMLEX sind Adjektive, die steigerbar sind, mit dem Merkmal GRADABLE markiert. Dabei können drei Arten unterschieden werden: Mit GRADABLE, mit GRADABLE :ER-EST T und mit GRADABLE :BOTH T markierte Adjektive. Mit GRADABLE markierte Adjektive sind mehrsilbig und der Komparativ kann durch die Verwendung von „more“, der Superlativ mit „most“ gebildet werden. Die Struktur GRADABLE :ER-EST T zeigt an, dass das Adjektiv durch die Anhängung von „-er“ bzw. „-est“ in den Komparativ bzw. Superlativ gesetzt werden kann. Die korrekte Bildung dieser Form wird durch die Anwendung einer Regel erreicht, die in [MACLEOD et al. 1998, S. 7] beschrieben ist. Mit GRADABLE :BOTH T markierte Adjektive können sowohl mit „more“/„most“, wie auch mit der Bildung einer Form mit „-er“/„-est“ gesteigert werden. Dabei werden die Formen mit „-er“/„-est“ in den Termen *comparative/1* und *superlative/1* abgelegt, die Formen, die mit „more“/„most“ gebildet werden, in den Termen *comparative_aliases/1* bzw. *super-*

lative_aliases/1 gespeichert.

Einige Adjektive haben unregelmässige Steigerungsformen. Diese Adjektive haben ein Merkmal :COMPARATIVE bzw. ein Merkmal :SUPERLATIVE, das die Komparativ-Form bzw. Superlativ-Form als Wert enthält.

4.3.4. positive_aliases/1, comparative_aliases/1 und superlative_aliases/1

Im Term *positive_aliases/1* kann ein Adjektiv eingesetzt werden, das in ACE-Texten als Alternative für die Positiv-Form des im Eintrag definierten Wortes gebraucht werden kann. Analog dazu kann im Term *comparative_aliases/1* ein Adjektiv definiert werden, das für die Komparativ-Form des im Eintrag definierten Adjektiv eingesetzt werden kann und der Term *superlative_aliases/1* kann Adjektive aufnehmen, die für die Superlativ-Form des im Eintrag definierten Adjektivs verwendet werden können.

Wie in Abschnitt 4.3.3 beschrieben, enthalten die in COMLEX mit GRADABLE :BOTH T markierten Adjektive in den ACELEX-Termen *comparative_aliases/1* bzw. *superlative_aliases/1* die Steigerungsformen, die mit „more“/„most“ gebildet werden. In den anderen Fällen liefert COMLEX für die Alias-Terme keine möglichen Werte. Wie in den anderen Wortarten hat der Benutzer die Möglichkeit, in diesen Termen „Synonyme“ zu definieren. Dabei muss er darauf achten, dass die Aliase der selben Flexionsstufe angehören, wie die zu ersetzenden Adjektive. D.h. als Alias für eine Superlativ-Form sollte auch eine Superlativ-Form im Term stehen.

4.3.5. complement/1 und complementing_preposition/1

Der Term *complement/1* gibt an, ob das Adjektiv eine Präpositionalphrase als Komplement verlangt. Die spezifische Präposition wird im Term *complementing_preposition/1* gespeichert.

Adjektive, die in COMLEX mit ADJ-PP markiert sind, nehmen eine

Präpositionalphrase mit einer bestimmten Präposition. Sie erhalten im Term *complement/1* den Wert *prepositional_phrase*, alle anderen Adjektive den Wert *no_complement*. Beispiele für ADJ-PP markierte Adjektive sind „fond of“, „keen on“ etc.. In einer PVAL-Liste werden die möglichen Präpositionen mitgeliefert. Für jede Präposition wird ein neuer Eintrag in AceLex generiert: Sie unterscheiden sich nur durch die im Term *complementing_preposition/1* gespeicherte Präposition voneinander.

Viele der mit ADJ-PP markierten Adjektive können auch ohne diese Präposition verwendet werden: „She is sceptical about the matter“, „She is sceptical“. Andere wiederum können nur mit der Präpositionalphrase stehen: „She is fond of him “, „*She is fond“. Comlex unterscheidet diese beiden Arten nicht. Um in Ace trotzdem beide Konstruktionen zuzulassen, wird neben dem Eintrag mit der Präpositionalphrase auch ein Eintrag ohne dieses Komplement generiert. Beispiel 4.12 zeigt die beiden Einträge, die dabei entstehen. Wieder wird eine Übergenerierung und damit die Entstehung inkorrekter Einträge wie in Beispiel 4.13 in Kauf genommen.

```
lexicon(adj, [logical_relation([fond]), positive([fond]), comparative([fonder]), superlative([fondest]), positive_aliases([]), comparative_aliases([]), superlative_aliases([]), complement([prepositional_phrase]), complementing_preposition([of]), comment([])]).
```

Beispiel 4.12: AceLex-Eintrag des Adjektivs „fond of“

```
lexicon(adj, [logical_relation([fond]), positive([fond]), comparative([fonder]), superlative([fondest]), positive_aliases([]), comparative_aliases([]), superlative_aliases([]), complement([no_complement]), complementing_preposition([]), comment([])]).
```

Beispiel 4.13: AceLex-Eintrag des Adjektivs „fond of“: Übergenerierung

4.3.6. comment/1

Im Term *comment/1* kann der Benutzer einen Kommentar für diesen Adjektiv-Eintrag einfügen.

4.4. Adverben

Nicht alle in COMLEX aufgeführten Adverben werden in ACELEX übernommen. Die Sprache ACE lässt z.B. modale Adverben („probably“) nicht zu. In COMLEX sind satzmodifizierende Adverben, wie z.B. epistemische Unsicherheiten („probably“), Wahrnehmungsarten („metaphorically“) etc.[20], mit META-ADV markiert. Um Anwendungsfehler der Benutzer auszuschliessen, werden alle mit META-ADV markierten Adverben in ACELEX nicht aufgenommen.
Ein Eintrag eines Adverbs sieht in ACELEX wie in Beispiel 4.14 aus.

lexicon(adv, [logical_relation([accurately]), adverb([accurately]), adverb_aliases([]), type([manner]), comment([])]).

Beispiel 4.14: ACELEX-Eintrag eines Adverbs

4.4.1. logical_relation/1

Das Merkmal :ORTH in COMLEX enthält die Grundform des Adverbs. Sie wird für den Term *logical_relation/1* daraus übernommen.

4.4.2. adverb/1

Im Term *logical_relation/1* steht die Nennform des Adverbs. Auch für den Term *adverb/1* liefert das Merkmal :ORTH in COMLEX den Wert. Adverben sind normalerweise unveränderlich in ihrer äusserlichen Form, darum braucht es keine weiteren

[20]META-ADV sind in [ROHEN WOLFF et al. 1998, S. 258-263] beschrieben.

Terme für flektierte Formen. Einige Adverben sind jedoch steigerbar: Z.B. „He drives faster than she“ oder „She left sooner“. Diese Information ist für AceLex nicht vorgegeben und wird nicht berücksichtigt.

4.4.3. adverb_aliases/1

Der Term *adverb_aliases/1* kann wie die *alias*-Terme der anderen Wortarten, Wörter oder Abkürzungen aufnehmen, die anstelle des im jeweiligen Eintrags beschriebenen Wortes stehen können. Comlex liefert keine Synonyme für Adverben. Der Benutzer kann hier nach seinen eigenen Bedürfnissen „Synonyme“ oder Abkürzungen einfügen.

4.4.4. type/1

Wie bereits mehrfach erwähnt ist Comlex ein Lexikon, das hauptsächlich die syntaktischen Eigenschaften der Worte beschreibt. Es liefert daher keine einheitliche Klassifizierung nach semantischen Eigenschaften. Comlex verfügt zwar neben syntaktischen Merkmalen über semantische Einteilungen wie *manner adverbs*, *temporal adverbs* und *location/direction adverbs*, aber nicht jedes Adverb ist zwingend einer dieser semantischen Kategorien zugeteilt. Das führt dazu, dass die einen Adverben in AceLex einen Modifikationstyp erhalten, andere jedoch nicht.
In Comlex existiert das Merkmal MANNER-ADV[21]. Damit sind Adverben markiert, die angeben, wie etwas getan wird („She did it accurately“). Comlex unterscheidet zusätzlich eine zeitliche Untergruppe dieser *manner adverbs*, die mit MANNER-ADV :TEMP T markiert sind. In diese Kategorie fallen modale Adverben, die einen zeitlichen Aspekt haben, wie z.B. „slowly“, „suddenly“ und „unceasingly“. Dabei soll keine Überschneidung mit den in Comlex TEMPORAL-ADV markierten Adverben – wie z.B. „already“ – auftreten, die nicht beschreiben, wie etwas getan wird. Diese Unterscheidung ist auch in AceLex gewünscht: Alle mit

[21]Das Merkmal ist in [Rohen Wolff et al. 1998, S. 252] beschrieben

MANNER-ADV oder MANNER-ADV :TEMP T markierten Adverben sollen in AceLex mit dem *type* „manner“ aufgeführt werden.
Neben den mit MANNER-ADV markierten Adverben existieren in Comlex Adverben, die mit EVAL-ADV gekennzeichnet sind[22]. Diese bezeichnen Adverben, die als evaluative adverbiale Komplemente von gewissen Verben[23] fungieren können. Ein Beispielsatz wäre „He behaves badly, but he means well“, wobei „badly“ bzw. „well“ als evaluative adverbiale Komplemente der Verben „behave“ bzw. „mean“ dienen. Da diese Adverben anzeigen, wie etwas getan wird, werden sie in Comlex ebenfalls mit *type* „manner“ aufgeführt.
Wie schon erwähnt, gibt es in Comlex ein Merkmal TEMPORAL-ADV. Die Beschreibung dazu findet sich in [Rohen Wolff et al. 1998, S. 253]. Mit diesem Merkmal sind nur Adverben markiert, die nicht auch *manner adverbs* sind. Die mit TEMPORAL-ADV gekennzeichneten Adverben repräsentieren entweder eine Zeit relativ zu einem bestimmten Zeitpunkt („soon“, „now“, „already“) oder zeigen die Häufigkeit eines Ereignisses an („occasionally“, „often“). Diese Adverben erhalten in AceLex den *type* „temporal“. Neben dem Merkmal TEMPORAL-ADV existiert in Comlex ein weiteres Merkmal, das die Zeit ausdrückt: TIMETAG. TIMETAG ist ein syntaktisch motiviertes Merkmal, das in [Rohen Wolff et al. 1998, S. 251] beschrieben ist:

> This type of adverb occurs in time expressions. They allow NTIME1[24] nouns to appear as bare NP's in clausal adverb positions.

In diese Kategorie fallen Adverben wie „ago“ („A year ago, he traveled to Spain“) und „early“ („She arrived three days earlier“). In AceLex erhalten auch diese Adverben den *type* „temporal“.
In Comlex mit LOC&DIR-ADV markierte Adverben zeigen einen Ort und/oder

[22] In [Rohen Wolff et al. 1998, S. 247] ist das Merkmal EVAL-ADV spezifiziert
[23] Diese Verben sind in Comlex mit den Subkategorisierungsstrukturen ADVP oder NP-ADVP markiert.
[24] vgl. Abschnitt 4.1.1.5

eine Richtung an. Beispiele hierfür sind: „there“, „inside“, „away“. Sie erhalten in AceLex den *type* „location_direction“.
Alle Adverben, die in keine dieser Kategorien fallen, erhalten den *type* „unspecified“.

4.4.5. comment/1

In diesen Term kann der Benutzer optional einen Kommentar einfügen.

5. Der AceLex-Compiler

In den folgenden Abschnitten stelle ich den Aufbau der Applikation vor, die im Rahmen dieser Arbeit entwickelt wurde. Die Applikation – der *AceLex-Compiler* – extrahiert die Informationen aus dem Quelllexikon COMLEX und baut das neue Lexikon ACELEX auf. Dabei werde ich nicht auf Implementierungs-Details eingehen, sondern die Verarbeitungsschritte auf einer relativ hohen Abstraktionsebene beschreiben. Informationen zur Implementierung der Applikation finden sich in Kapitel 6.

Der *AceLex-Compiler* überführt das in LISP implementierte Lexikon COMLEX in die Prolog-Struktur von ACELEX (vgl. Abbildung 5.1).

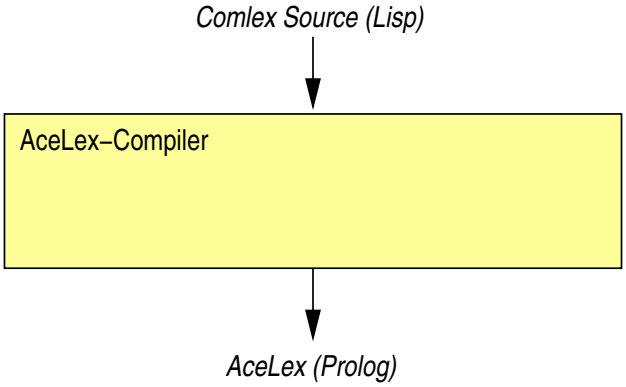

Abbildung 5.1.: Der *AceLex-Compiler*

Der Begriff *Compiler* wird in dieser Arbeit gemäss dieser Definition verwendet:

> Ein Compiler ist ein Computerprogramm, das eine in einer bestimm-

ten Sprache verfasste Quelldatei in eine Zieldatei übersetzt, die in einer anderen Sprache geschrieben ist.

Dieser Compiler-Begriff weicht etwas ab von seiner Definition in der Softwareentwicklung. Dort wird der Begriff für ein Computerprogramm verwendet, das ein in einer Quellsprache geschriebenes Programm in ein Programm umwandelt, das in einer anderen Programmiersprache geschrieben ist. Das Quellprogramm und das Zielprogramm müssen dabei semantisch äquivalent sein (vgl. Definition auf Wikipedia[1]). Der Unterschied zu dem in dieser Arbeit verwendeten Compiler-Begriff besteht v.a. darin, dass das generierte Lexikon AceLex nicht semantisch äquivalent zu Comlex ist: In AceLex sollen nicht alle in Comlex gespeicherten Informationen vorhanden sein, sondern nur die benötigten Informationen. Der Umfang des Lexikons wird reduziert, die übernommenen Informationen sind aber semantisch äquivalent übersetzt. Ein weiterer Unterschied zwischen dem in der Software-Entwicklung und dem in dieser Arbeit verwendeten Compiler-Begriff besteht darin, dass es sich nicht um Programme handelt, die übersetzt werden, sondern um Daten-Dateien, die in einer bestimmten Sprache geschrieben sind. Trotzdem ist es gerechtfertigt, in dieser Arbeit von Compilern zu sprechen: Wie bei den Compilern in der Softwareentwicklung wird eine lexikalische und syntaktische Analyse durchgeführt und die Ziel-Datei danach erzeugt.

5.1. Aufteilung in Front- und Backend

Die Arbeitsweise eines Compilers lässt sich im Wesentlichen in zwei Phasen unterteilen: Eine Analysephase, die den Quelltext analysiert, und eine Synthesephase, die aus dem Ergebnis der Analyse die Zieldatei erzeugt. Oftmals trennt man die beiden

[1] Wikipedia ist eine Enzyklopädie in mehr als 50 Sprachen, die von Freiwilligen auf der ganzen Welt aufgebaut wird. Ihre Inhalte dürfen dauerhaft frei kopiert und verbreitet werden. Die deutschsprachige Ausgabe wurde im Mai 2001 gestartet und umfasst derzeit 141841 Artikel. Website der deutschsprachige Ausgabe `http://de.wikipedia.org/wiki/Hauptseite`. Definition Compiler: `http://de.wikipedia.org/wiki/Compiler`

Phasen auch physisch voneinander: Die Analyse behandelt man in einem *Frontend*, die Synthesephase in einem *Backend*. Zwischen *Frontend* und *Backend* muss dann eine Schnittstelle geschaffen werden, ein Zwischenformat. Dieses Zwischenformat wird vom *Frontend* aus dem Quelltext generiert und durch das *Backend* in die Zielstruktur überführt.
Auch der *AceLex-Compiler* ist – wie in Abbildung 5.2 gezeigt – in ein *Frontend* und ein *Backend* aufgeteilt.

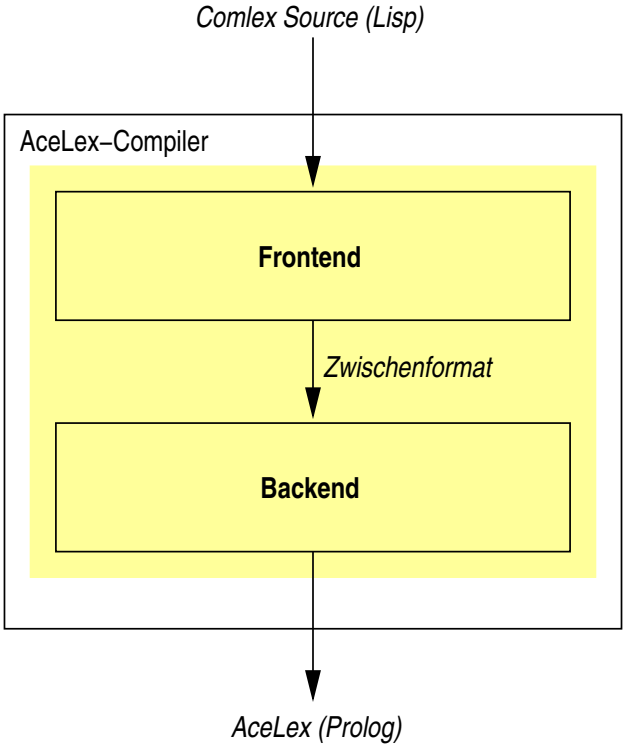

Abbildung 5.2.: Der *AceLex-Compiler*: Aufteilung in *Front-* und *Backend*

Die Trennung von *Frontend* und *Backend* hat folgende Vorteile:

- Das *Frontend* und das *Backend* sind unabhängig voneinander und können getrennt entwickelt werden. Das *Frontend* muss nichts über die Implementierung des *Backend* wissen und umgekehrt. Sie können unabhängig voneinander ausgetauscht und angepasst werden.

- Neue lexikalische Quellen können einfach eingebunden werden: Dafür muss ein weiteres unabhängiges *Frontend* geschrieben werden, das ebenfalls das Zwischenformat generiert. Das *Backend* muss im Idealfall[2] nicht erweitert werden. Dies war eigentlich die Hauptmotivation für die Aufteilung: Schon zu Beginn war klar, dass COMLEX nicht alle Informationen liefern kann, die für ACELEX vorgegeben sind. Auf der Stufe des Zwischenformats können mehrere lexikalische Quellen vereinigt werden, was aber eine nicht triviale Aufgabe ist.

- Änderungen im ACELEX-Format sind einfach durchzuführen. Es muss nur das *Backend* angepasst werden. Bei den verschiedenen *Frontends* müssen keine Anpassungen vorgenommen werden.

- Es kann auch ein Lexikon für ein anderes Projekt geschaffen werden, das sich ebenfalls auf das aus COMLEX generierte Zwischenformat stützt. Dazu muss lediglich ein neues *Backend* entwickelt werden, das die Datei im Zwischenformat als Input nimmt und daraus das neue Lexikon im entsprechenden Format generiert.

5.2. Das Zwischenformat

Wie schon erwähnt, bedingt die Aufteilung des *AceLex-Compilers* in ein *Front-* und ein *Backend* ein Zwischenformat. Wünschenswert für ACELEX ist ein Format, das bereits als Standard existiert und auch in Zukunft verwendet werden kann.
Die Verwendung eines standardisierten Formats hat folgende Vorteile:

- Da diverse Hersteller von Produkten den Standard unterstützen, werden eventuell Produkte entwickelt, die bereits diesem Standard entsprechen: D.h. in Zukunft könnten Lexika direkt im Standard-Format erworben werden. Dieses

[2]Werden durch das neue *Frontend* mehr Informationen bereitgestellt als durch das vorherige *Frontend* und sollen diese Informationen dann auch im Ausgabedatenstrom des Compilers erscheinen, muss das *Backend* erweitert werden.

neue Lexikon könnte auf der Stufe des Zwischenformats integriert werden: Damit würde für diese Quelle das *Frontend* überflüssig, das *Backend* könnte aber weiter verwendet werden.

- Das aus COMLEX generierte, standardisierte Zwischenformat könnte als Lexikon für andere Applikationen dienen, die sich auf diesen Standard stützen.

Ein Zwischenformat in XML (vgl. Abschnitt 6.2) bietet sich für diese Aufgabe an, da XML sich je länger je mehr als Standard für den Dokumentenaustausch etabliert und bereits viele Werkzeuge zur Verarbeitung von XML existieren.
In dieser Arbeit wird das *Open Lexicon Interchange Format, Version 2* (OLIF2) (vgl. Abschnitt 6.2.1) als Zwischenformat verwendet. Das *Frontend* des *AceLex-Compilers* übersetzt COMLEX in OLIF2 und diese OLIF2-Zwischenrepräsentation wird vom *Backend* in das finale ACELEX überführt (vgl. Abbildung 5.3).

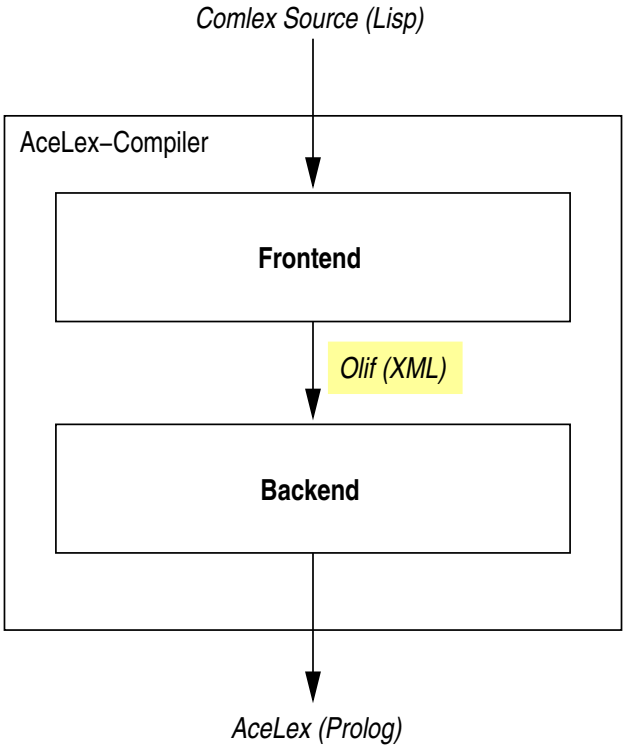

Abbildung 5.3.: Der *AceLex-Compiler*: Zwischenformat OLIF2

Ziel ist es, bei der Überführung möglichst alle Merkmale von COMLEX in die OLIF2-Repräsentation zu übernehmen, nicht nur die für ACELEX benötigte Information. Sollte ACELEX später einmal mehr Informationen benötigen, die in COMLEX enthalten sind, muss nur das *Backend* des *AceLex-Compilers* erweitert werden. Ausserdem kann auf diese Weise die OLIF2-Repräsentation auch als Quelllexikon für andere Applikationen dienen, die andere Informationen als ACELEX von COMLEX benötigen.

5.3. Das Frontend des AceLex-Compilers

Das *Frontend* des *AceLex-Compilers* bildet der *Comlex-Compiler*. Er nimmt als Input COMLEX und generiert eine Repräsentation davon im OLIF2-Format. Der *Comlex-Compiler* selbst ist wiederum in ein *Frontend* und ein *Backend* gegliedert: Der *Comlex Scanner* und *Comlex Parser* bilden das *Frontend*, der *Comlex Olifgenerator* das *Backend* (vgl. Abbildung 5.4).

Der *Comlex Scanner* erhält als Input das COMLEX-Lexikon und führt eine lexikalische Analyse durch: Er analysiert das Lexikon auf die Gültigkeit seiner lexikalischen Elemente, d.h. er teilt die in COMLEX gefundenen Zeichen in sinnvolle Einheiten – wie z.B. Schlüsselwörter, Bezeichner, Zahlen und Operatoren – ein, die er dann in COMLEX-*Tokens* umwandelt und ausgibt. Trifft er zum Beispiel in COMLEX auf das Zeichen „(“, wandelt er es in das COMLEX-*Token* „LPAREN“ um, das für eine linke, öffnende Klammer steht. Diese COMLEX-*Tokens* werden in der COMLEX *Scanner Specification* definiert. Durch den *Scanner Generator* JFLEX (vgl. Abschnitt 6.1.1) wird der *Comlex Scanner* aufgebaut.

Der *Comlex Parser* überprüft die Syntax des COMLEX-Lexikons, er macht eine syntaktische Analyse. Als Input nimmt er die vom *Comlex Scanner* gelieferten COMLEX *Tokens* und generiert als Ergebnis eine JAVA-Repräsentation von COMLEX, die COMLEX *Syntax Representation*. Die Syntax von COMLEX ist in der COMLEX *Parser Specification* in einer linksrekursiven Grammatik festgehalten. In dieser Grammatik ist JAVA-Code eingebaut, der während dem Parse-Vorgang das Zwischenformat des

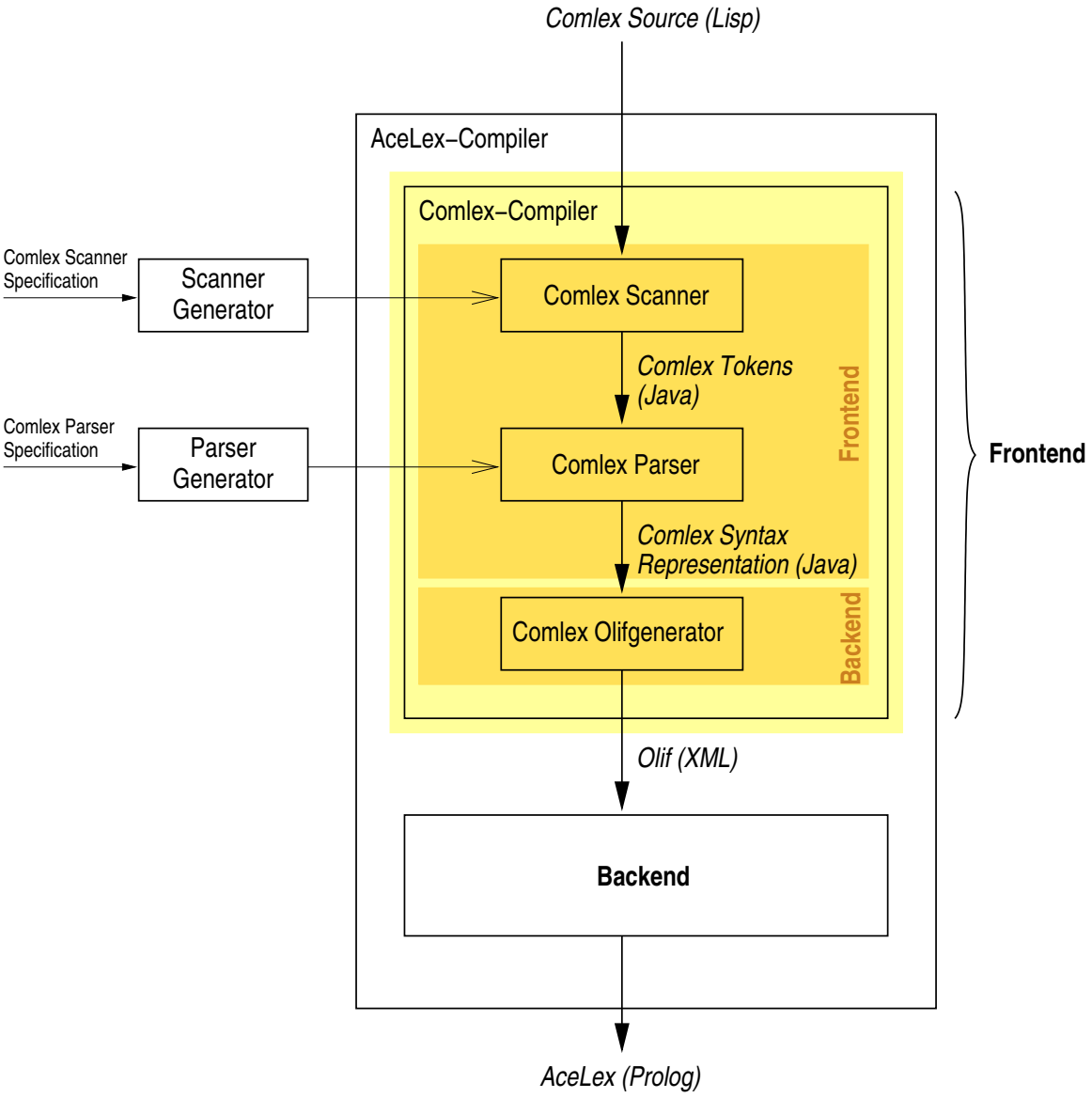

Abbildung 5.4.: Das *Frontend* des *AceLex-Compilers*

Comlex-Compilers aufbaut: Die COMLEX *Syntax Representation*. Durch den *Parser Generator* CUP (vgl. Abschnitt 6.1.1) wird aus der COMLEX *Parser Specification* der *Comlex Parser* aufgebaut.
Das *Backend* des *Comlex-Compilers* ist der *Comlex Olifgenerator*. Der *Comlex Olifgenerator* überführt die COMLEX *Syntax Representation* in das OLIF2-Format.

5.4. Das Backend des AceLex-Compilers

Das *Backend* des *AceLex-Compilers* bilden der *Olif-Transformer* und der *Prox-Converter* (vgl. Abbildung 5.5). Die Rolle des *Olif-Transformers* ist es, die OLIF-Datei einzulesen und durch XSL-*Transformations* (vgl. Abschnitt 6.2.2) die für ACELEX relevanten Informationen herauszufiltern und in ein für ACELEX geeignetes Format zu überführen. Als Output generiert der *Olif-Transformer* eine Datei im PROX-Format, das im Abschnitt 6.2.3 beschrieben ist.
Der *Prox-Converter* nimmt den vom *Olif-Transformer* generierten Ausgabedatenstrom im PROX-Format und überführt ihn in die normale PROLOG-Struktur. Der *Prox-Converter* kann jede im PROX-Format geschriebene Datei oder jeden PROX-Eingabedatenstrom in PROLOG übersetzen. Er setzt Klammern, Punkte, Kommata am richtigen Ort, setzt das *Escape*-Zeichen „\“ vor PROLOG-relevante Sonderzeichen und umschliesst Atome, die Leerschläge oder Sonderzeichen enthalten, mit Hochkommata. Ebenfalls in Hochkommata gesetzt werden Atome, die mit einem Grossbuchstaben beginnen.

5.5. Erweiterungen

Wie bereits erwähnt, wurde der *AceLex-Compiler* unter anderem in ein *Frontend* und ein *Backend* aufgeteilt, damit eine weitere lexikalische Quelle auf einfache Weise in die Verarbeitung eingebunden werden kann. Neben dem *Frontend* des *AceLex-Compilers*, das COMLEX in die OLIF2-Struktur übersetzt, kann in einer

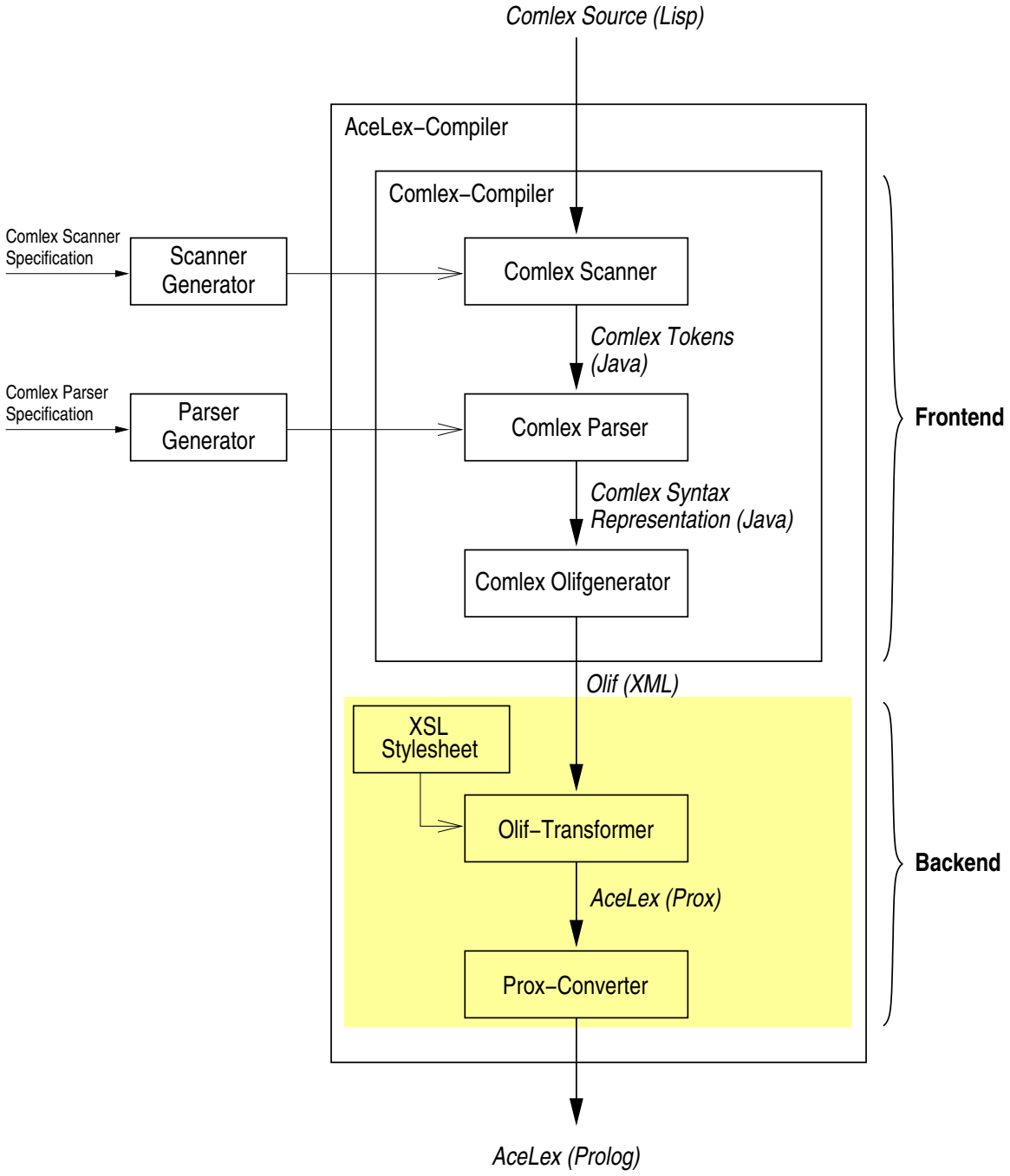

Abbildung 5.5.: Das *Backend* des *AceLex-Compilers*

weiterführenden Arbeit ein neues *Frontend* geschaffen werden, wie in der Grafik 5.6 durch den *X-Compiler* angedeutet.

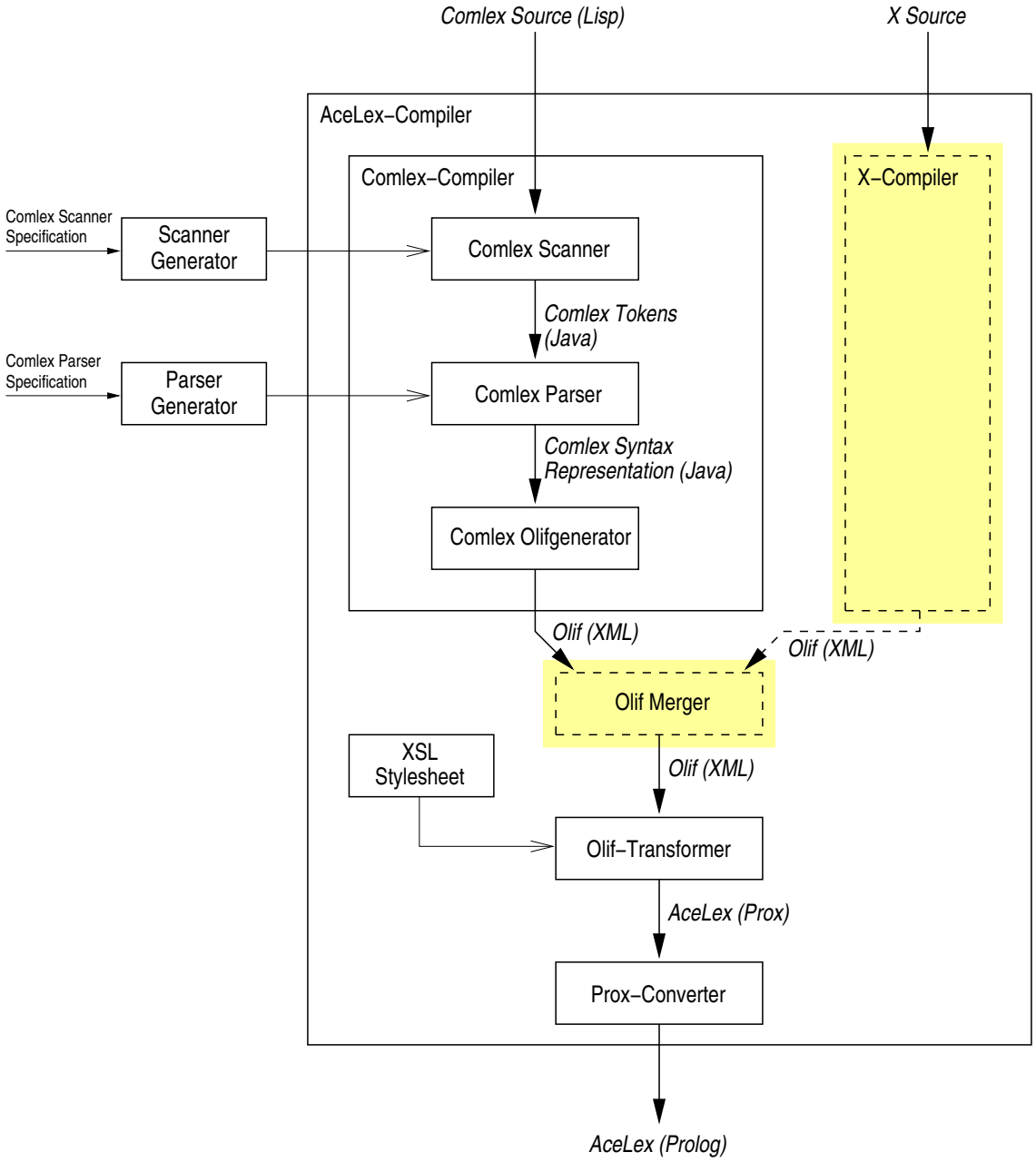

Abbildung 5.6.: Erweiterbarkeit des *AceLex-Compilers* durch ein neues *Frontend*

Der *X-Compiler* nimmt als Input das Lexikon X und generiert daraus eine Re-

präsentation in OLIF2. Mit Hilfe des *Olif Mergers* können die aus COMLEX generierte und die aus dem Lexikon X generierte OLIF-Struktur zu einer verschmolzen werden, was keine triviale Aufgabe ist. In der Grafik 5.6 sind die (noch) nicht existierenden Komponenten mit unterbrochenen Linien gezeichnet. Da bis jetzt nur das *Frontend* für COMLEX definiert ist und kein zweites Lexikon beigezogen wurde, existiert kein *X-Compiler* und kein *Olif Merger*.

5.6. Fehlerbehandlung

Bei jedem Verarbeitungsschritt können Fehler auftreten, die aufgefangen werden müssen. Tritt ein Fehler auf, bricht das Programm die Verarbeitung ab und gibt eine sinnvolle Fehlermeldung aus. Diese Meldungen erläutern, wo der Fehler aufgetreten ist und geben Hinweise für den Grund des Auftretens. Abbildung 5.7 zeigt die verschiedenen Fehlerquellen und Fehlerarten.

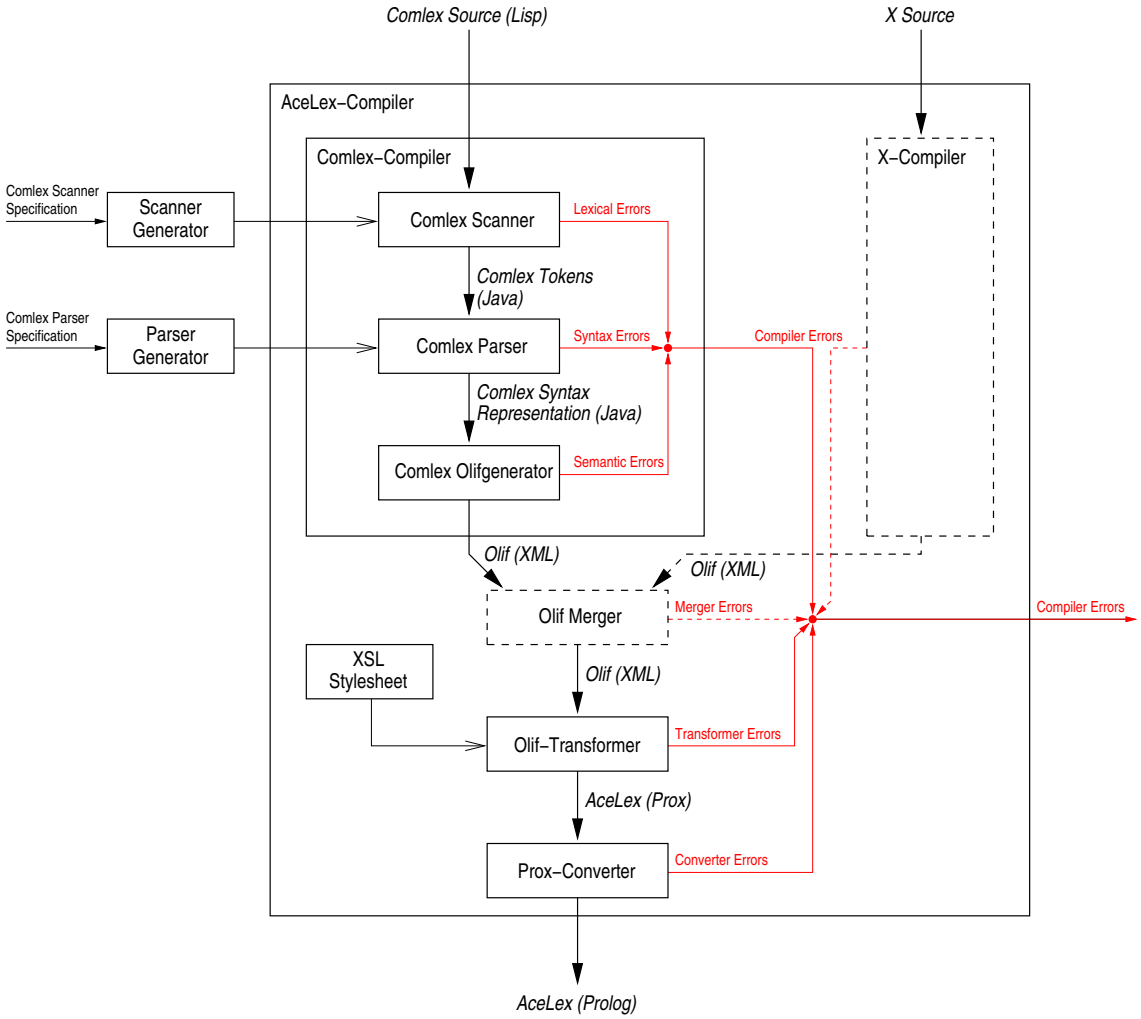

Abbildung 5.7.: Prozessüberblick mit Fehlerbehandlung

6. Implementierung

Dieses Kapitel ist der Implementierung des *AceLex-Compiler* gewidmet. Die folgenden Abschnitte sollen einen Überblick über die Sprachen, Formate und Werkzeuge bieten, die in dieser Arbeit verwendet wurden. Ausserdem werden einige wichtige Details der Implementierung besprochen.
In diesem Projekt wurde hauptsächlich mit zwei Arten von Sprachen gearbeitet: Mit der objektorientierten Programmiersprache Java und mit XML-basierten Sprachen und Formaten. In den folgenden Abschnitten werden zuerst die Java-basierten Komponenten vorgestellt, danach wird kurz auf XML und seine Verwendung in dieser Arbeit eingegangen.

6.1. Java

Die Entscheidung, ein Zwischenformat in XML (vgl. Abschnitt 5.2) zu verwenden, hatte einen direkten Einfluss auf die Wahl der Programmiersprache, in der dieses Projekt implementiert werden sollte. Die Sprache Java[1] ist für die Verarbeitung von XML sehr gut geeignet. Sie hat sich ausserdem in den letzten Jahren mehr und mehr in der Software-Entwicklung etabliert. Es werden viele Werkzeuge und Umgebungen[2] bereit gestellt, die das Arbeiten mit Java sehr erleichtern. Das Vorhandensein von *Compiler Construction Tools* wie JFlex und CUP, die im Abschnitt 6.1.1

[1] Informationen und Downloads sind unter `java.sun.com` zu finden.
[2] In dieser Arbeit wurde mit der *Eclipse*-Plattform gearbeitet. Informationen und Downloads unter `www.eclipse.org/`.

vorgestellt werden, unterstützten die Wahl von JAVA als Programmiersprache in dieser Arbeit. In Abbildung 6.1 sind die Komponenten, die in dieser Arbeit in JAVA implementiert sind, hervorgehoben.

Das *Frontend* des *AceLex-Compiler* ist vollständig in JAVA implementiert: JFLEX und CUP generieren gemäss Spezifikation einen *Scanner* bzw. *Parser* in JAVA (vgl. Abschnitt 6.1.1). Der *Comlex Scanner* und der *Comlex Parser* generieren auch einen Output in JAVA: Der *Comlex Scanner* liefert die COMLEX *Tokens* in Form von JAVA-Konstanten, der *Comlex Parser* baut die ebenfalls in JAVA geschriebene COMLEX *Syntax Representation* auf. Aus dieser Repräsentation generiert der in JAVA implementierte *Comlex Olifgenerator* das OLIF2 Zwischenformat, das jedoch XML basiert ist.

Das *Backend* ist ebenfalls in JAVA implementiert, die Ausgabeformate jedoch sind nicht mehr in JAVA. Die einzigen Programmanweisungen im *AceLex-Compiler*, die nicht in JAVA geschrieben sind, befinden sich im XSL-Stylesheet (vgl. Abschnitt 6.2.2). Verarbeitet wird das Stylesheet jedoch durch den *Olif-Transformer*, der in JAVA implementiert ist.

6.1.1. JFlex und CUP

JFLEX und CUP sind *Compiler Construction Tools*[3], die dafür benutzt werden, lexikalische (JFLEX) bzw. syntaktische (CUP) „Analysers“ aufzubauen. In diesem Abschnitt stütze ich mich hauptsächlich auf [SEVENICH 1999] und [HUDSON 1999]. CUP (Abkürzung für Java Based Constructor of Useful Parsers) ist ein System, das benutzt wird, um *LALR* (Left Associated Left Recursion) Parser mit Hilfe simpler Spezifikationen zu generieren. CUP wurde YACC (Yet Another Compiler Compiler, in der Sprache C geschrieben) nachempfunden und implementiert die meisten Eigenschaften dieses weitverbreiteten Parser-Generators. Wie der Name schon sagt, ist

[3]Homepage von JFLEX: `www.jflex.de`; Homepage von CUP: `www.cs.princeton.edu/~appel/modern/java/CUP/`

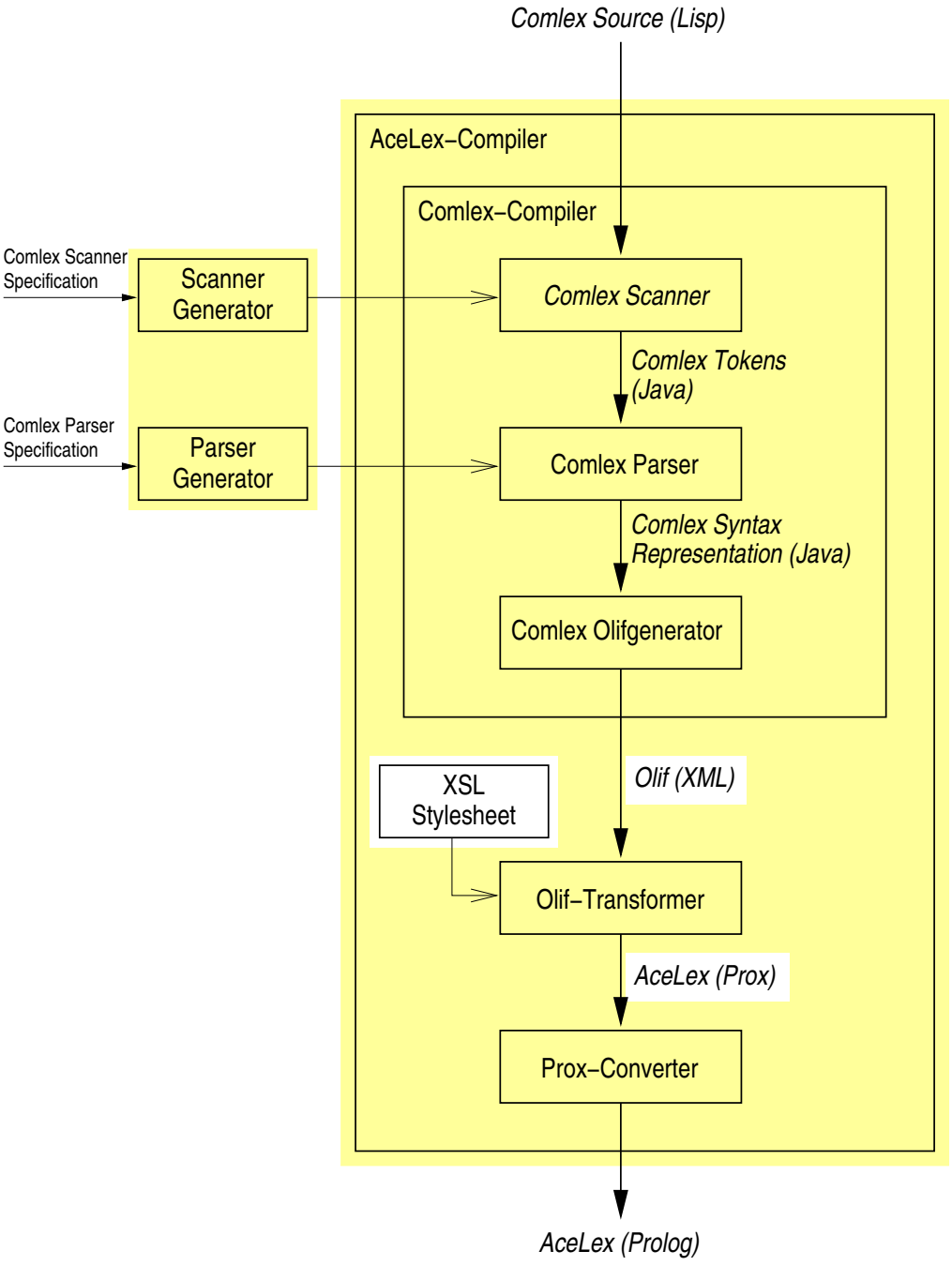

Abbildung 6.1.: Applikationen und Formate in JAVA

CUP jedoch in Java geschrieben, benutzt Spezifikationen, die Java-Code enthalten und produziert Parser, die in Java implementiert sind.
Der grosse Vorteil solcher *Compiler Construction Tools* ist die einfache Anwendung. Man kann damit schnell und ohne grosse Einarbeitungszeit einen Parser herstellen.

6.1.1.1. Der Scanner

Um CUP benutzen zu können, braucht es zwei Komponenten. Als erstes muss ein lexikalischer „analyser“, ein sogenannter Scanner oder Lexer kreiert werden. Ein solcher Scanner teilt die Zeichen der zu parsenden Datei in gültige oder ungültige Einheiten (*tokens*) auf, wie zum Beispiel Nummern, Schlüsselwörter, spezielle Symbole oder Zeichen, die nicht verarbeitet werden sollen. So werden die Zeichen für die Weiterverarbeitung im syntaktischen „analyser“, dem Parser (hier: CUP) und für die spezifische Sprache des gewünschten Outputs aufbereitet. Man könnte einen Scanner mit einem „Spell-Checker“ vergleichen. JFlex ist ein Werkzeug, das den Scanner aufbaut. Es braucht lediglich eine vom Benutzer hergestellte Spezifikationsdatei, in der die gültigen *tokens* definiert sind. Durch den Aufruf JFlex <Datei> wird dann die Java-Version des Scanners produziert.
In diesem Projekt ist die Comlex *Scanner Specification* die von mir hergestellte Spezifikationsdatei, in der die Comlex *Tokens* definiert sind. Der *Scanner Generator* JFlex generiert dann den *Comlex Scanner* (vgl. Abbildung 6.1).

6.1.1.2. Der Parser

Die zweite Komponente prüft die Syntax der Quelldatei. So wie man den Scanner als „Spell-Checker“ bezeichnen könnte, wäre der Parser der „Grammatik-Checker“. Um diese Komponente zu generieren, muss eine einfache Spezifikation mit der Grammatik der zu parsenden Datei erstellt werden. Kompiliert man diese Spezifikationsdatei, werden mehrere Dateien generiert, worunter auch die Java-Version des syntaktischen „analysers“, die eigentliche Parser Datei ist. Der Parser ermöglicht nicht nur

die Grammatik auf ihre Gültigkeit zu prüfen, sondern auch bei jedem gefundenen gültigen grammatischen Konstrukt beliebigen Programmcode auszuführen, der vom Benutzer spezifiziert ist.
In diesem Projekt ist die COMLEX *Parser Specification* die vom Benutzer herzustellende Grammatik-Spezifikation. Der *Parser Generator* CUP stellt den *Comlex Parser* her (vgl. Abbildung 6.1).

6.2. XML

In diesem Abschnitt folgt eine kurze Beschreibung der *eXtensible Markup Language* (XML). In Abbildung 6.2 sind die XML-basierten Komponenten und Formate hervorgehoben. In dieser Arbeit ist sowohl das Zwischenformat OLIF2 wie auch das PROX-Format (vgl. Abschnitt 6.2.1 bzw. 6.2.3) in dieser Sprache gehalten. Ausserdem arbeitet der *Olif-Transformer* mit XSLT, einer auf XML aufbauenden Technologie, die in Abschnitt 6.2.2 beschrieben wird.
XML ist ein Standard zur Strukturierung von Daten[4]. XML wird vom *World Wide Web Consortium*[5] (W3C) entwickelt. Das W3C kümmert sich um die Weiterentwicklung des WWW und dessen Standards wie z.B. HTML, HTTP und vielen anderen. Da es sich bei XML um einen offiziellen und akzeptierten Standard handelt, werden zahlreiche Werkzeuge angeboten, die das Arbeiten mit XML erleichtern. Ausserdem werden immer mehr Technologien entwickelt, die auf XML aufbauen.
XML ist eine vereinfachte Form der *Standard Generalized Markup Language* (SGML). XML ist eine Meta-Sprache, die es dem Benutzer erlaubt, seine eigenen *Markups* zu kreieren. Im Gegensatz dazu sind die *Markups* in HTML festgelegt: <HEAD> und <BODY> z.B. sind fest in den HTML-Standard integriert und können nicht geändert werden. In XML gibt es keine vordefinierten *Tags*, es gibt

[4] In diesen Abschnitten stütze ich mich v.a. auf [NÄF 2002], [ECKSTEIN und CASABLANCA 2001], [MCLAUGHLIN 2001] und [FAASCH 2000].
[5] Die Website vom W3C: `www.w3c.org`. Auf dieser Seite finden sich die Spezifikationen der Standards, Tutorials, Werkzeuge zur Verarbeitung etc.

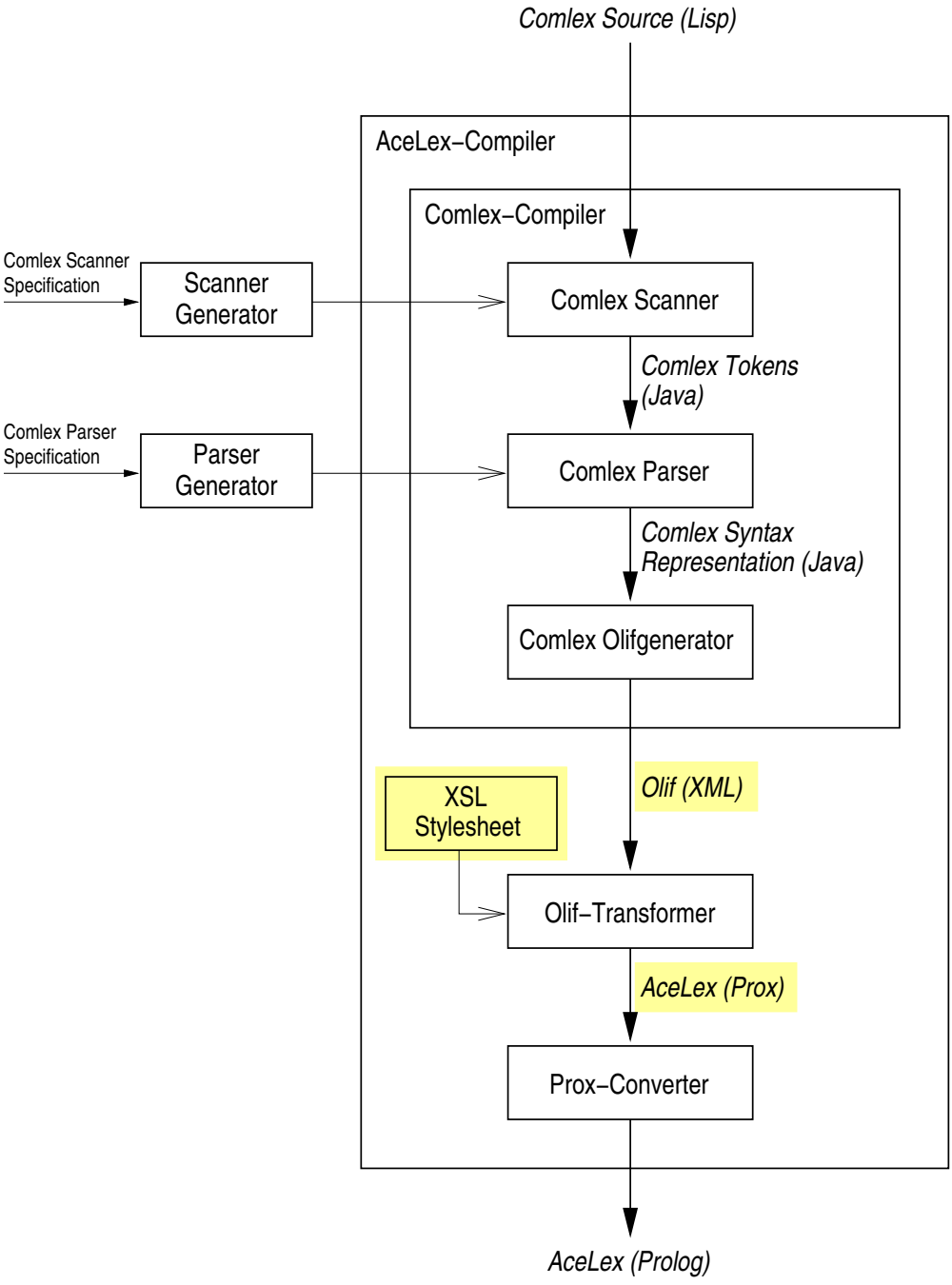

Abbildung 6.2.: XML-basierte Komponenten und Formate

nur diejenigen *Tags*, die man selbst definiert hat.
Da XML-Dokumente einfache Textdokumente sind, die auf eine bestimmte, vorgeschriebene Weise aufgebaut sind, können sie problemlos auf allen Betriebssystemen gelesen und durch viele Werzeuge und Programmiersprachen verarbeitet werden.
XML ist, wie der Name impliziert, erweiterbar. Es lassen sich beliebig komplexe Strukturen erstellen. Eine *Document Type Definition* (DTD) legt dabei den Aufbau des XML-Dokuments verbindlich fest. DTDs sind ein Teil der XML-Spezifikation, sind jedoch selbst nicht im XML-Format geschrieben. Sie sind für heutige Bedürfnisse aber zu eingeschränkt und können nicht auf einfache Weise erweitert werden. Darum werden sie von XML-*Schema* abgelöst. XML-*Schema*-Dokumente sind selbst in XML geschrieben, sie können wiederum durch DTDs oder XML-*Schema* definiert werden. *Schemas* sind erweiterbar und können so mit der extensiven Entwicklung von XML und dessen Erweiterungen mithalten.

6.2.1. Olif2

In diesem Abschnitt soll kurz auf das Zwischenformat OLIF2 eingegangen werden. Das *Open Lexicon Interchange Format, Version 2* (OLIF2) ist ein Versuch, einen Industrie-Standard für den Austausch von lexikalischen und terminologischen Daten in XML einzuführen. Entwickelt und unterstützt ist das Format vom OLIF2 *Consortium*, einem Zusammenschluss grösserer NLP[6]-Technologie Hersteller. Die Spezifikation von OLIF2 ist frei zugänglich und kann unter der URL: `www.olif.net` heruntergeladen werden. Viele grössere Firmen, die sich mit *Machine Translation* befassen, wie z.B. *Systran*, *Logos* und *Linguatec* benutzen OLIF2. OLIF2 ist sowohl für monolinguale wie auch für multilinguale Lexika geeignet, obwohl es ursprünglich für den Austausch von multilingualen sprachlichen Daten entwickelt wurde.
Die in diesem Projekt verwendete OLIF-Version wird durch eine *XML Data Type Definition* (DTD) (vgl. Abschnitt 6.2) spezifiziert, die von `www.olif.net/`

[6]Natural Language Processing

`Specification` heruntergeladen werden kann. Es sind jedoch Entwicklungen im Gange, OLIF in Zukunft mittels *XML Schema Definition Language* (XSD) zu spezifizieren, um gewisse Nachteile des DTD-Formats zu kompensieren. Informationen dazu finden sich in [McCORMICK et al.] und [THURMAIR und LIESKE 2002].
Die Struktur des in dieser Arbeit als Zwischenformat verwendeten OLIF2-Formats ist in [McCORMICK 2002] beschrieben. OLIF2 ist so definiert, dass es dem Benutzer bei gewissen Angaben erlaubt, die DTD zu erweitern, um eine für seine Zwecke geeignete Form der Einträge zu erhalten. Kategorien, in denen eine Expansion erlaubt ist, sind z.B. die Flexion und die Subkategorisierung. Wenn immer möglich, sollte natürlich den Vorschlägen von OLIF2 nachgekommen werden, um die Übertragbarkeit der Daten zu wahren.
In diesem Projekt wurden zwei Expansionen der DTD von OLIF2 vorgenommen[7]: Das Element <inflection> und das Element <synFrame> wurden erweitert. In OLIF2 werden als Werte für das <inflection> Element *inflection patterns* vorgeschlagen (z.B. *inflects_like book, books*). In COMLEX sind keine solchen Muster angegeben, sondern die Flexionsformen werden entweder durch Regeln generiert oder sind als ganzes Wort direkt im Eintrag angegeben. Es wäre zwar möglich gewesen, die Wörter auf die entsprechenden *inflection patterns* abzubilden, sie hätten aber dann für ACELEX wieder zurück auf ihre ursprüngliche Form geführt werden müssen. Um den Aufwand für diese Arbeit zu verringern, wurde in der DTD das Element <inflection> durch Elemente wie z.B. <plural> oder <thirdSing> erweitert. Das andere Element, das in der DTD erweitert wurde ist das Element <synFrame>, das die Subkategorisierungsrahmen der Wörter enthält. Es wurde ein Element <subc> eingefügt, in dem die COMLEX-Subkategorisierungsstrukturen stehen. Motiviert wurde diese Entscheidung dadurch, dass kein Informationsverlust in Kauf genommen werden wollte: Die ausführlichen, fein granularen Subkategorisierungsstrukturen von

[7] Ausserdem wurden für zwei COMLEX-Merkmale zwei neue Werte in OLIF2 eingeführt, was jedoch keine Anpassung der DTD erfordert: NCOLLECTIVE markierte Nomen erhalten im Element <synType> den Wert „ncoll“, VCOLLECTIVE den Wert „vcoll“.

COMLEX lassen sich schlecht in die vorgeschlagenen, mehr grobkörnigeren Werte von OLIF2 überführen. Es wurden nur diejenigen Subkategorisierungsstrukturen von COMLEX in die OLIF2-Datei übernommen, die für ACELEX relevant sind. Das entspricht nicht der ursprünglichen Idee: Es sollten alle Informationen von COMLEX in die OLIF2-Repräsentation aufgenommen werden, damit, falls ACELEX einmal mehr Informationen aus COMLEX benötigt, diese durch eine einfache Erweiterung des *Backends* des *AceLex-Compiler* gewonnen werden können. Ausserdem wäre eine vollständige OLIF2-Repräsentation von COMLEX auch für andere Applikationen interessant. Der Grund dafür, dass nur die von ACELEX benötigten Subkategorisierungsrahmen von COMLEX übernommen werden, liegt wiederum in der Minimierung des Aufwands für diese Arbeit. Das *Frontend* des *AceLex-Compiler* könnte aber für andere Applikationen einfach angepasst werden, um alle Subkategorisierungsrahmen zu übernehmen.

Die Erweiterungen von OLIF2 der beiden Elemente <inflection> und <synFrame> wurden keineswegs leichtfertig vorgenommen. Der Idealfall wäre die absolute Erfüllung des Standards, um von allen in Abschnitt 5.2 genannten Vorteilen zu profitieren. Der Aufwand, den Standard vollständig zu erfüllen, wäre jedoch unverhältnismässig hoch gewesen. Schon durch die Einführung des OLIF2-Standards für diese Arbeit, hat sich der Aufwand vervielfacht, da viel Energie in die Überführung der syntaktisch motivierten Merkmale von COMLEX in die standardisierten Werte von OLIF2 gesteckt werden musste. Viele dieser syntaktisch motivierten Merkmale konnten nicht direkt überführt werden: Sie sind in der OLIF2-Repräsentation im OLIF2-Element <usage> durch eine kurze Beschreibung charakterisiert. Diese Merkmale sind für ACELEX jedoch nicht relevant.

Die für diese Arbeit angepasste OLIF2-DTD und damit auch das Standard-Format, auf dem diese Arbeit aufbaut, befindet sich auf der im Anhang A beigefügten CD-ROM.

6.2.2. XSLT

In diesem Projekt überführt der *Olif-Transformer* durch *eXtensible Stylesheet Language Transformations*[8] (XSLT) die OLIF2-Datei in die Datei im PROX-Format. XSLT wurde entworfen, um XML-Dokumente in andere XML-Dokumente zu transformieren. Es eignet sich jedoch auch dazu, XML-Dokumente in andere textbasierte Formate zu überführen, wie zum Beispiel in HTML- oder Text-Dokumente. XSLT ist ein Bestandteil der *eXtensible Stylesheet Language* (XSL), die wiederum ein Teil von XML ist. XSL regelt die Formatierung der XML-Dokumente: Mit Hilfe von XSL kann festgelegt werden, wie ein XML-Dokument für unterschiedliche Ausgabemedien dargestellt werden soll. Der grosse Vorteil von XSL ist es, dass das gleiche Quelldokument durch Verwendung verschiedener *Stylesheets* in unterschiedliche Arten von Ausgabedokumenten überführt werden kann, je nach Verwendungszweck. Dabei wird das Quelldokument nicht verändert.

Es werden zwei Dokumente benötigt, um eine XSL-Transformation durchzuführen. Zuerst einmal braucht es ein Quelldokument, das transformiert werden soll. Dieses Dokument muss im XML-Format vorliegen. Dann wird ein *Stylesheet* benötigt, das die Transformationsregeln enthält. Durch das Abarbeiten dieser Regeln erhält man einen Ausgabedatenstrom, der das transformierte Dokument enthält.

Der Transformationsprozess wird nicht direkt auf dem Quelldokument ausgeführt, sondern arbeitet auf einer sogenannten *DOM-Repräsentation*, d.h. auf einer Baumstruktur, in der die einzelnen Elemente des Quelldokumentes als Knoten dargestellt sind. Von der Dokumentwurzel (*Root-Element*) zweigen seine Unterelemente ab, von denen wiederum ihre eigene Unterelemente etc.. Auch das *Stylesheet* selbst wird durch eine solche Struktur repräsentiert. Zu Beginn traversiert der Prozess das Quelldokument und beginnt dabei an der Dokumentwurzel. Im *Stylesheet* wird dann eine Regel gesucht, die beschreibt, was mit diesem Knoten geschehen soll. Eine solche

[8] [FAASCH 2000] ist eine übersichtliche und gut verständliche Einführung in XSLT. Auf der Website vom W3C (`www.w3c.org`) gibt es ausserdem eine ausführliche Beschreibung.

Regel wird *Template Rule* genannt. Findet sich eine solche *Template Rule* für die Dokumentwurzel, wird sie instantiiert. Instantiieren heisst in diesem Zusammenhang, dass die Regel (*Template Rule*) im *Stylesheet* abgearbeitet wird. Die Anweisungen in der *Template Rule* steuern das weitere Vorgehen. Ist in diesen Anweisungen z.B. eine neue *Template Rule* für ein weiteres Element des Quelldokuments definiert, wird das Quelldokument wieder traversiert, bis das Element gefunden wird. Danach werden wieder die Anweisungen der entsprechenden Regel im *Stylesheet* abgearbeitet. Auf diese Weise findet ein ständiger Wechsel zwischen dem Quelldokument und dem *Stylesheet* statt, bis das ganze *Stylesheet* durchgearbeitet ist.

6.2.3. Das Prox-Format

Wie schon erwähnt, wurden XSL-*Transformations* für die Transformation von Daten entwickelt, die im XML-Format vorliegen und in eine andere Datei im Markup-Format (XML, HTML) überführt werden sollen. Es können durchaus auch rein textbasierte Dateien erstellt werden, jedoch stellen die XSL-*Transformations* keine Formatierungshilfen zur Verfügung. Eine schöne, gut lesbare Darstellung des PROLOG-Formats von ACELEX wäre direkt vom OLIF2-Format über XSL-*Transformations* nur schwierig zu erreichen. Dieser Umstand motivierte zur Entwicklung des PROX-Formats (PROlog Xml), das die syntaktische Struktur von PROLOG mit XML-*Tags* abbildet. PROX ist ein applikationsunabhängiges Format zu Darstellung von PROLOG in XML. Zur Veranschaulichung des Aufbaus von PROX soll ein Beispiel dienen. In Beispiel 6.1 ist der schon in Abschnitt 4.3 vorgestellte Eintrag des Adjektivs „cold“ dargestellt, wie er in ACELEX vorhanden ist.

```
lexicon(adj, [logical_relation([cold]), positive([cold]), comparative([colder]), superlative([coldest]), positive_aliases([]), comparative_aliases([]), superlative_aliases([]), complement([no_complement]), complementing_preposition([]), comment([])]).
```

Beispiel 6.1: ACELEX-Eintrag des Adjektivs „cold“

Der Eintrag eines Wortes in AceLex ist ein Prolog-Fakt, gekennzeichnet durch den Punkt am Ende des Eintrags. Aufgebaut ist er durch komplexe Terme, die durch einen Funktor und ihre in Klammern stehenden Argumente definiert sind. Diese Argumente können weitere komplexe Terme, Listen oder Konstanten (Atome) sein, die durch Kommata voneinander getrennt sind. Der Eintrag von „cold“ sieht im Prox-Format wie in Beispiel 6.2 aus. Um es übersichtlicher zu gestalten, habe ich die Einträge für die *aliases*-Terme ausgelassen. Prolog-Fakten sind im Prox-Format durch das XML-*Tag* <fact> eingeleitet und werden mit dem *Tag* </fact> abgeschlossen. Komplexe Terme sind mit dem *Tag* <compound_term> gekennzeichnet, das ein Attribut *functor* nimmt, welches den Funktornamen als Wert hält. Listen sind mit <list> bzw. </list> umschlossen, Konstanten mit <constant> bzw. </constant>.

Um das Lexikon XML-konform zu machen, wurde ein *Root-Element* definiert: Die einzelnen Prolog-Fakten sind von den *Tags* <prox> bzw. </prox> umschlossen. Das Prox-Format müsste noch erweitert werden, um weitere Prolog-Konstrukte wie z.B. Regeln oder Operatoren darstellen zu können. Da es für die Bedürfnisse der Prolog-Einträge in AceLex entwickelt wurde, können im Prox-Format bis jetzt Fakten, komplexe Terme[9], Listen und Konstanten dargestellt werden. Eine Erweiterung des Prox-Formats erfordert eine Anpassung des *Prox-Converter*, damit eine korrekte Übersetzung in das Prolog-Format gewährleistet ist.

[9]Natürlich sind auch Operatoren komplexe Terme, hier sind jedoch nur „normale“ komplexe Terme gemeint, die in der Form $functor(argument^{+})$ vorliegen.

```
<fact>
    <compound_term functor=„lexicon“>
        <constant>adj</constant>
        <list>
            <compound_term functor=„logical_relation“>
                <list>
                    <constant>cold</constant>
                </list>
            </compound_term>
            <compound_term functor=„positive“>
                <list>
                    <constant>cold</constant>
                </list>
            </compound_term>
            <compound_term functor=„comparative“>
                <list>
                    <constant>colder</constant>
                </list>
            </compound_term>
            <compound_term functor=„superlative“>
                <list>
                    <constant>coldest</constant>
                </list>
            </compound_term>
            [...]
            <compound_term functor=„complement“>
                <list>
                    <constant>no_complement</constant>
                </list>
            </compound_term>
            <compound_term functor=„complementing_preposition“>
                <list/>
            </compound_term>
            <compound_term functor=„comment“>
                <list/>
            </compound_term>
        </list>
    </compound_term>
</fact>
```

Beispiel 6.2: PROX-Eintrag des Adjektivs „cold“

7. Schlusswort

In dieser Arbeit wurde ein Lexikon für Ace hergestellt: Das AceLex. Als Quelllexikon diente Comlex, das beinahe alle Informationen enthält, die für AceLex benötigt werden. Um die Informationen zu extrahieren, wurde hauptsächlich mit Java und XML-basierten Techniken gearbeitet.

Comlex kann nicht alle Informationen für AceLex bereitstellen. Die noch fehlenden Informationen müssen in einer weiterführenden Arbeit aus einem anderen Lexikon extrahiert werden. Dabei kann auf diese Arbeit aufgebaut werden: Bei der Implementierung wurde grossen Wert auf die einfache Integration eines neuen Lexikons gelegt: Auf der Stufe des standardisierten Zwischenformats Olif2 können die Lexika zusammengeführt werden. Mit der Einführung eines Standards als Zwischenformat wird diese Arbeit auch für andere Projekte interessant: Ausgehend davon können neue Lexika erstellt werden.

Neben der Vervollständigung von AceLex muss die in dieser Arbeit entstandene Version von AceLex noch gefiltert werden. In der Sprache Ace ist das Vokabular Einschränkungen unterworfen. Lexikalische Restriktionen, wie z.B. dass keine modalen und intensionalen Verben („may“, „can“ bzw. „hope“, „believe“) in Ace zugelassen sind, werden in dieser Version von AceLex bis jetzt nicht berücksichtigt. Diese Einschränkungen müssen in einem nächsten Schritt noch realisiert werden in AceLex.

Ungeachtet der noch zu erarbeitenden Komponenten von AceLex kann es bereits im Rahmen von Attempto eingesetzt werden. Ohne ein grosses Lexikon, bleibt Attempto ein Laborexperiment: Das Projekt hat mit diesem Lexikon die Möglichkeit,

einen Schritt in Richtung Praxis zu machen. Ich hoffe, dass sich sowohl das Lexikon, wie auch das ganze System in der Praxis bewähren können.

A. CD-ROM mit Programm-Code

Auf der beiliegenden CD-ROM finden sich folgende Informationen:

- Eine Readme-Datei im Textformat, die die gleichen Informationen wie dieser Anhang A enthält.
- Das Verzeichnis „AceLex“, das unter anderem folgende Informationen enthält:
  - Den Programm-Code des *AceLex-Compilers*
  - Das Quelllexikon: COMLEX *Syntax Version 3*
  - Das neue Lexikon: ACELEX

  Eine genaue Aufstellung des Inhalts des „AceLex“-Verzeichnisses ist in Abschnitt A.2 zu finden.
- Die beiden COMLEX Spezifikationen: [ROHEN WOLFF et al. 1998] und [MACLEOD et al. 1998]
- Die vorliegende Lizentiats-Arbeit im PDF-Format.
- Die LaTeX-Quellen und Bilddateien zur vorliegenden Lizentiatsarbeit.

A.1. Anweisungen zur Installation des AceLex-Compilers

Vorraussetzung für die Ausführung der Applikation ist die Installation von Java (`java.sun.com`). In diesem Projekt wurde mit der Java Version `J2SE1.4.2` gearbei-

tet. Soll die Applikation lediglich ausgeführt werden, genügt es, die Laufzeitumgebung (`J2RE`) zu installieren.
Um den *AceLex-Compiler* zu installieren, muss zuerst die Umgebungsvariable `ACELEX_HOME` definiert werden, die den Pfad bis und mit dem Verzeichnis „AceLex“ enthalten muss (ohne abschliessendem „/“ bzw. „\“). Im „AceLex“-Verzeichnis sind alle weiteren benötigten Informationen enthalten. In Abschnitt A.2 wird der Inhalt dieses Verzeichnisses kurz beschrieben.
Um die Ausführbarkeit der Applikation so einfach wie möglich zu gestalten, werden die Skriptdateien `alc.sh` für Unix-basierte Systeme bzw. `alc.bat` für Windows-Rechner mitgeliefert. Sie befinden sich im Unterverzeichnis „bin“. Von dort kann die Applikation über die Kommandozeile der Shell bzw. der Windows-Eingabeaufforderung mit folgendem Befehl gestartet werden:

alc <Pfad zur Lexikondatei `COMLEX-SYNTAX-3.1`> `acelex`

Neu generiert werden dabei das Zwischenformat `acelex.olif`, die Prox-Version von AceLex `acelex.prox` und AceLex selbst: `acelex.pl`. Bei Unix-Systemen ist darauf zu achten, dass die Shell-Datei `alc.sh` ausführbar ist: Die Rechte der Datei müssen mit dem Befehl „`chmod +x alc.sh`“ allenfalls angepasst werden (`x` macht die Datei ausführbar).
Um die Skript-Dateien nicht nur vom Verzeichnis, in dem sie sich befinden, ausführen zu können, empfiehlt es sich, den Suchpfad des jeweiligen Rechners so anzupassen, dass er die Skriptdatei enthält. Die Umgebungsvariable `PATH` muss dafür um den Pfad nach `$ACELEX_HOME/bin` (Unix) bzw. `%ACELEX_HOME%\bin` (Windows) ergänzt werden.

A.2. Inhalt des „AceLex“-Verzeichnisses

- Das Verzeichnis „bin“ enthält die zwei erwähnten Skriptdateien zur Ausführung des *AceLex-Compilers*:

alc.sh für Unix-basierte Systeme

alc.bat für Windows-basierte Systeme

- Im Verzeichnis „classes“ sind die von Java generierten Klassendateien abgelegt.
- Das Verzeichnis „doc“ enthält die JAVADOC-Dokumentation des *Application Programming Interface* (API) vom *AceLex-Compiler*.
- Das Verzeichnis „dtd“ enthält die OLIF2-DTD inkl. der Datei `olifx.dtd`, die die Erweiterungen für dieses Projekt spezifiziert.
- Im Verzeichnis „gensrc“ befinden sich die von JFLEX und CUP erstellten Dateien: Der Scanner, der Parser und eine Datei mit den COMLEX-Tokens.
- Das Verzeichnis „lib“ enthält die benötigten Programmbibliotheken:

  antlexcup Enthält zwei Ant-Tasks (`ant.apache.org`), die durch eine im Verzeichnis „spec“ enthaltene `build`-Datei aufgerufen werden: Der eine Task startet JFLEX, um den Scanner zu generieren, der andere Task ruft CUP für die Parser-Generation auf.

  jcmdline-1.0.2 Ist ein Werkzeug zur Verarbeitung von Kommandozeilen-Argumenten (`jcmdline.sourceforge.net`).

  jcup Wird für die Parser-Generation gebraucht und enthält Basisklassen, die der Parser bei der Verarbeitung benötigt (`www.cs.princeton.edu/~appel/modern/java/CUP`).

  jdom Wird für das effiziente Erstellen von XML-Dokumenten benötigt (`www.jdom.org`).

  jflex Wird für die Scanner-Generation benötigt und enthält Basisklassen, die der Scanner bei der Verarbeitung benutzt (`www.jflex.de`).

  junit Ermöglicht das Ausführen von automatischen Tests über einzelne Programmeinheiten (Unit Tests) (`junit.sourceforge.net`).

- Im Verzeichnis „spec“ sind die Scanner-Spezifikation und die Parser-Spezifikation abgelegt. Ausserdem findet sich hier die oben erwähnte `build`-Datei. Diese Datei wird von Ant (`ant.apache.org`) dazu verwendet, die im Verzeichnis „gensrc“ abgelegten Scanner- und Parser-Dateien aus den Spezifikationen zu generieren. Sollen diese Dateien neu generiert werden, muss Ant installiert sein.

- Im Verzeichnis „src“ ist der Source-Code abgelegt.

- Das Verzeichnis „testsrc“ enthält Unit-Tests für ausgewählte Komponenten.

- Im Verzeichnis „xsl“ befinden sich die Stylesheets für die XSL-Transformation.

- Neben diesen Verzeichnissen finden sich hier noch folgende Dateien:

  COMLEX-SYNTAX-3.1 Das Quelllexikon

  acelex.pl Das generierte Lexikon AceLex

  acelex.olif Das Zwischenformat des Lexikons im Olif2-Format

  acelex.prox Die Prox-Version von AceLex

Für alle weiteren Informationen wird auf die vollständige Lizentiatsarbeit verwiesen.

B. Hinweise zu Comlex

In COMLEX Version 3.1 sind folgende Format-Fehler zutage getreten:

- Die drei Nomen „cañon“, „señor“ und „vicuña“, sind nicht mit mit $\tilde{n}$ dargestellt, sondern folgendermassen: „ca∼non“, „se∼nor“, „vicu∼na“. In ACELEX sind sie darum in einfache Hochkommata gesetzt.

- Das Verb „compere“ (= „To serve as the master of ceremonies.“) ist in COMLEX folgendermassen eingetragen: „comp‘ere“. Es kommt ursprünglich aus dem Französischen. Jedoch ist der Graph in COMLEX normalerweise korrekt über dem e (= $è$) platziert, wie auch im nachfolgenden Eintrag des Nomens „compère“, das in dieser Form in COMLEX erscheint. In ACELEX ist „comp‘ere“ in einfache Hochkommata gesetzt.

- Beim Nomen „jabber“ ist das Symbol *NONE* im Merkmal PLURAL in Anführungszeichen gesetzt. D.h. der Wert wird als String interpretiert und führt in ACELEX im Term *plural/1* zu folgendem Eintrag: *plural(['*NONE*'])*

C. Curriculum vitae

Personalien

Name	Alexandra Bünzli
Adresse	Im Eichli 18
Telefon	044 853 17 40
Geburtsdatum	13.07.1977
Zivilstand	ledig
Heimatort	Maur ZH

Ausbildung

1990-1995	Kantonsschule Bülach	Maturitäts-Typus B (mit Latein)
ab Oktober 1997	Universität Zürich	Hauptfach: Germanistik 1. Nebenfach: Informatik Schwerpunkt Software-Engineering 2. Nebenfach: Computerlinguistik

Anstellungen

bis 1997	Diverse Ferienanstellungen
1997-2001	Mitarbeiterin der Kiosk AG (Valora)
seit Februar 2001	Buchhaltung Debitoren/Kreditoren Teilzeit 30% in der Firma Gemperli AG, Zürich

Mitgliedschaften

seit 1993	aktives Mitglied des Damenturnvereins Steinmaur seit 2003 Aktuarin
seit 1999	aktives Mitglied der Sport- und Freizeitvereinigung Rümlang
seit 1999	aktives Mitglied des Fachvereins der Computerlinguistik der Universität Zürich, Revisorin des Fachvereins

Sprachaufenthalte

Juli 1996	Drei Wochen Intensiv-Kurs in der Französischen Sprache in Nizza, Frankreich
Juni-Sept 1997	Drei Monate Australien

Hobbies

Lesen
Sport: Korbball, verschiedene Teamsportarten, Wintersport
Kino
Malen, Basteln

D. Hinweise zum Glossar

Das Glossar besteht aus Definitionen, die hauptsächlich aus folgenden beiden Quellen übernommen wurden:

- Aus dem Glossar des Instituts für Computerlinguistik der Universität Zürich
- Aus dem Glossar der Fakultät für Linguistik und Literaturwissenschaft der Universität Bielefeld

Die Quelle wird am Ende der Definition jeweils genannt. Ist keine Quelle angegeben, stammt die Defintion aus meiner Feder. Anfügungen meinerseits in den übernommenen Definitionen sind durch eckige Klammern [...] gekennzeichnet.

Glossar

Adjunkt Adjunkte werden – im Gegensatz zu Komplementen (auch obligatorisch Ergänzung genannt) – nicht durch den Selektionsrahmen eines Wortes verlangt, d.h. es sind grammatisch nicht notwendige Ergänzungen. [Glossar IfI 2004]

Antonym Ein Wort ist ein Antonym, wenn es die gegenteilige Bedeutung eines anderen Wortes beinhaltet. Im einfachsten Fall stammen die beiden sprachlichen Ausdrücke aus demselben Bereich, z.B. „männlich“ vs. „weiblich“. Etwas schwieriger wird es bei sprachlichen Ausdrücken derselben Familie, die semantisch jedoch schwieriger zu definieren sind, wie z.B. „fallen“ vs. „aufsteigen“, „kommen“ vs. „gehen“ oder „zurück“ vs. „vorwärts“. [Glossar IfI 2004]

Derivation Die Wortbildung untersucht die Muster, nach denen Wörter intern strukturiert sind und neue Wörter gebildet werden. Mit der Derivation (Ableitung) und der Komposition (Zusammensetzung) werden zwei Haupttypen von Wortbildungsmustern unterschieden. Bei der Derivation (Ableitung) werden neue Wörter durch das Anfügen wortartspezifischer Suffixe an einen Stamm (z.B. „Leit“ „-er“, „Leit“ „-ung“) oder das Anfügen nicht frei vorkommender Präfixe an freie Morpheme (z.B. „ver-“ „leiten“, „un-“ „möglich“) gebildet. [Glossar liOn, Linguistik Online 2004]

Flexion Flexion oder Beugung ist neben der Wortbildung einer der beiden Teilbereiche der Morphologie. Im Gegensatz zur Wortbildung, die die Bildung von Wortstämmen beschreibt, untersucht die Flexionslehre die Bildung von Wortformen. Lexeme verschiedener Wortarten können durch morphologisch unterschiedliche Wortformen realisiert werden, die unterschiedliche syntaktisch-semantische Informationen tragen. Es wird zwischen der Deklination von Nomina [z.B. „das Lexikon“, „des Lexikons“, „die Lexika“], der Konjugation von Verben [z.B. „ich rede“, „du redest“, „ich redete“] und der Komparation von Adjektiven [z.B. „schön“, „schöner“, „am schönsten“] unterschieden. Flexion kann auf unterschiedliche Art vollzogen werden: Im Deutschen wird sie überwiegend durch das Anfügen bestimmter Endungen, der Flexionsaffixe, oder durch Änderungen der Wortstamms realisiert. [Glossar liOn, Linguistik Online 2004]

Vollformenlexikon Ein Vollformenlexikon enthält einen Eintrag für jede Wortform. D.h. es ist nicht nur das Lemma (Grundform) eines Wortes eingetragen, sondern auch die flektierten Formen des Wortes. Beispiel: Beim Nomen „Brot“ sind neben der Form „Brot“ für den Nominativ, Akkusativ und Dativ, die Genitivform „Brotes“, wie auch die Pluralformen „die Brote“ (Nominativ, Akkusativ, Genitiv) und „den Broten“ (Dativ) aufgeführt.

Hyperonym/Hypernym [Verallgemeinerung] Ein Hyperonym eines Begriffs ist ein Oberbegriff dieses Begriffs. Die Extension des Oberbegriffs ist eine Obermenge der Extension eines Unterbegriffs. Ein Eigenname wird üblicherweise nicht als Unterbegriff eines andern Begriffs betrachtet. Beispiele: „Hund“ ist ein Hyperonym von „Pudel“, „Hund“ ist kein Hyperonym von „Fido“. [Glossar IfI 2004]

Hyponym [Verfeinerung] Ein Hyponym eines Begriffs ist ein Unterbegriff dieses Begriffs. Beispiel: „Pudel“ ist ein Hyponym von „Hund“.[Glossar IfI 2004]

Lemma Ein Lemma ist ein Eintrag bzw. ein einzelnes Stichwort in einem Lexikon oder Wörterbuch. [Glossar IfI 2004]

lemma-basiertes Lexikon Ein lemma-basiertes Lexikon führt nur einen Eintrag pro Lemma auf, das für alle zu einem Wort gehörenden Wortformen steht.

Lexem Das Lexem ist die abstrakte Basiseinheit des Lexikons , die in verschiedenen grammatikalischen Wortformen realisiert werden kann. Lexeme können auch Teil anderer Lexeme sein. Lexem im weiteren Sinn wird auch synonym verwendet für Wort als lexikalische Einheit, bzw. Element des Wortschatzes. [Glossar IfI 2004]

Meronym Teil-Ganzes Beziehung. Ein Begriff ist ein Meronym eines anderen Begriffes, wenn der erstere ein Teil des zweiten ist [Gegenteil von [Holonym]. Beispiel: „Hand“ ist ein Meronym von „Arm“. [Glossar IfI 2004]

Morphem Ein Morphem ist die kleinste bedeutungstragende Einheit einer Sprache, d.h. dass es als semantische Einheit nicht in kleinere Einheiten aufgeteilt werden kann. Beispiele: „Buch“, „drei“, „es“, „lang“, „un-“, „-lich“ [Glossar IfI 2004]

Morphologie Morphologie ist der Teilbereich der Sprachwissenschaft, der sich mit der internen Struktur von Wörtern und dem Zusammenhang zwischen den Wörtern einer Sprache beschäftigt. [Glossar IfI 2004]

Nominalphrase Eine Nominalphrase ist eine syntaktische Kategorie , die im Satz entweder Subjekt- oder Objektfunktion ausfüllt, aber auch Teil von Präpositionalphrasen sein kann. NPs bestehen jeweils aus einem nominalen

Kern, z.B. Nomen („Obst“), Eigennamen („Philip“), Pronomen („ich“, „jener“, „der“, „alle“) oder aus einem Gliedsatz („... dass du nicht lesen kannst“). Der nominale Kern kann in verschiedener Weise erweitert werden, u.a. durch Adjektive, Artikel, Präpositionalattribute („die Lust am Untergang“), Appositionen („Dieser Text, völlig unverständlich, ist unbrauchbar.“), Relativsätze u.a. [...] [Glossar IfI 2004]

Phonologie Die Phonologie ist eine Teildisziplin der Sprachwissenschaft, welche sich mit den bedeutungsunterscheidenden Sprachlauten (Phonemen, ein Phonem ist die kleinste bedeutungstragende Lauteinheit, die aus einem akustischen Sprachfluss ermittelt werden kann), ihren relevanten Eigenschaften, Relationen und Systemen einer Sprache sowie von Sprachen allgemein beschäftigt. [Glossar IfI 2004]

Phrase In der Phrasenstrukturgrammatik ist Phrase die Bezeichnung für eine Menge von syntaktischen Elementen, die eine syntaktische Konstituente (= Wortgruppe oder Satzteil von relativer Selbständigkeit) bilden. Die vier Phrasentypen sind Nominalphrasen (bestehend aus nominalen Ausdrücken mit entsprechenden attributiven Erweiterungen, z.B. „Rolf“/„der alte Rolf“/„er“/„Rolf, der gerne träumt“/u.a.), Verbalphrasen (z.B. „träumt“/„sieht das Feuer“/„glaubt, dass er recht hat“/u.a.), Präpositionalphrasen („auf dem Tisch“, „seit gestern“, „damit“) und Adjektivphrasen (“klein“, „ziemlich bunt“, „auf ihren Erfolg stolz“). [Glossar IfI 2004]

Präpositionalphrase Präpositionalphrasen sind komplexe syntaktische Kategorien, deren Leitelement (head) Präpositionen wie „an“ oder „mit“ sind. Sie erfüllen in erster Linie die syntaktischen Funktionen des Adverbials („Sie geht in das Haus“), des Attributs („der Tisch im Garten“) oder des Objekts („Er glaubt an den Erfolg“). [Glossar liOn, Linguistik Online 2004]

Subkategorisierung Die Untergliederung lexikalischer Kategorien in Unterklassen, um das syntaktische Verhalten der jeweiligen lexikalischen Einheiten beschreiben zu können, d.h. das Beschränkungsproblem bei der Erzeugung von Sätzen zu lösen. Die Generierung ungrammatischer Sätze kann auf diese Weise verhindert werden. [Glossar IfI 2004]

Synonym Worte mit (fast) gleicher Bedeutung sind Synonyme. Beispiel: „gehen“ und „sich fortbewegen“ sind Synonyme. [Glossar IfI 2004]

Syntax Die Syntax ist ein Teilbereich der Grammatik natürlicher Sprachen: Ein System von Regeln, die beschreiben, wie aus einem Inventar von Grundelementen (Morphemen ,Wörtern, Satzgliedern) durch spezifische syntaktische Mittel (Morphologische Markierung, Wort- und Satzgliedstellung [...] u.a.) alle wohlgeformten Sätze einer Sprache abgeleitet werden können. [Glossar IfI 2004]

Literaturverzeichnis

[Boguraev und Briscoe 1989] Boguraev, B. und T. Briscoe (1989). *Utilising the ldoce grammar codes.*. In: Boguraev, B. und T. Briscoe, Hrsg.: *Computational Lexicography for Natural Language Processing*, S. 85–116. Longman, London; New York.

[Burnage 1990] Burnage, Gavin (1990). *CELEX A Guide for Users*. Nijmegen. PostScript file with the introduction to CELEX of the CELEX User Guide, in European A4-format.

[CELEX English Linguistic Guide 1995] CELEX English Linguistic Guide (1995). *English Linguistic Guide*. CELEX, Centre for Lexical Information. PostScript file of the CELEX User Guide on English, in European A4-format.

[CELEX Readme-Datei 1995] CELEX Readme-Datei (1995). *Readme-Datei der CD-ROM Version der CELEX Lexical Database*. CELEX, Centre for Lexical Information. PostScript file of the CELEX User Guide on English, in European A4-format.

[Eckstein und Casablanca 2001] Eckstein, Robert und M. Casablanca (2001). *XML. Pocket Reference*. O'Reilly & Associates, Sebastopol, CA (USA), 2. Aufl.

[Faasch 2000] Faasch, Heiko (2000). *XSLT- Die XSL Transformationssprache*. `www.fh-wedel.de/~si/seminare/ws00/Ausarbeitung/5.xslt/xslt1.htm`, Stand 12. September 2004. Der Text ist im Rahmen des Seminars „XML und Java“ im Wintersemester 2000 an der Fachhochschule Wedel (DE) entstanden.

[Fellbaum 1999] Fellbaum, Christiane, Hrsg. (1999). *WordNet: an Electronic Lexical Database*. Language, speech, and communication. MIT Press, Cambridge, Massachusetts; London, England, 2 Aufl.

[Fuchs et al. 2004] Fuchs, Norbert E., U. Schwertel, S. Hoefler und G. Schneider (2004). *Extended Discourse Representation Structures in Attempto Controlled English*. Technischer Bericht, Department of Informatics, University of Zurich, Zurich, Switzerland.

[Fuchs et al. 1999] Fuchs, Norbert E., U. Schwertel und R. Schwitter (1999). *Attempto Controlled English - Not Just Another Logic Specification Language*. In: Flener, Pierre, Hrsg.: *Logic-Based Program Synthesis and Transformation*, Nr. 1559 in *Lecture Notes in Computer Science*, Manchester, UK. Eighth International Workshop LOPSTR'98, Springer.

[Glossar IfI 2004] Glossar IfI (2004). Ein Glossar der computerlinguistisch relevanten Begriffe. Institut für Computerlinguistik, Universität Zürich. `www.ifi.unizh.ch/cl/Glossar/glossary.html` Stand 24. August 2004.

[Glossar liOn, Linguistik Online 2004] Glossar liOn, Linguistik Online (2004). Glossar der Fakultät für Linguistik und Literaturwissenschaft der Universität Bielefeld `luna.lili.uni-bielefeld.de/lion/glossar` Stand 24. August 2004.

[Hartmann 2001] Hartmann, R.R.K. (2001). *Teaching and Researching Lexikography*. Applied linguistics in action. Longman, Harlow, England.

[Herbst und Kerstin 1999] Herbst, Thomas und P. Kerstin, Hrsg. (1999). *The Perfect Learners' Dictionary (?)*. Lexicographica. Series maior ; 95. Niemeyer, Tübingen. Symposium entitled „The Perfect Learners' Dictionary (?)", which was held at the University of Erlangen-Nürnberg in April 1997.

[Hess 2004a] Hess, Michael (2004a). *Einführung in die Computerlinguistik I*. Vorlesung WS 2003/2004. `www.ifi.unizh.ch/CL/hess/classes/ecl1/ecl1.0.pdf`, Stand 20. August 2004.

[Hess 2004b] Hess, Michael (2004b). *Einführung in die Computerlinguistik II*. Vorlesung SS 2004. `www.ifi.unizh.ch/CL/hess/classes/ecl2/ecl2.0.pdf`, Stand 20. August 2004.

[Hudson 1999] Hudson, Scott E. (1999). *CUP User's Manual*. `www.cs.princeton.edu/~appel/modern/java/CUP/manual.html`. Modified by Frank Flannery, C. Scott Ananian, Dan Wang with advice from Andrew W. Appel.

[LDOCE, Lisp version 1978] LDOCE, Lisp version (1978). Informationen über die maschinenlesbare Version des *Longman Dictionary of Contemporary English*, der 1978 publiziert wurde: `www.longman.com/dictionaries/research/reslisp.html`, Stand 12. August 2004.

[LDOCE3 1995] LDOCE3 (1995). Informationen über die maschinenlesbare NLP Version der dritten Ausgabe des *Longman Dictionary of Contemporary English*: `www.longman.com/dictionaries/research/resnlapp.html`, Stand 12. August 2004.

[Lieske et al. 2001] Lieske, Christian, S. McCormick und G. Thurmair (2001). *The Open Lexicon Interchange Format (OLIF) Comes of Age*. Proceedings of the Machine Translation Summit VIII. `www.eamt.org/summitVIII/papers/lieske.pdf`.

[Macleod et al. 1998] Macleod, Catherine, R. Grisham und A. Meyers (1998). *COMLEX Syntax Reference Manual. A Specification for a Lexical Knowledge Base. Version 3.0*. Computer Science Department, New York University. Prepared for the Linguistic Data Consortium, University of Pennsylvania. `nlp.cs.nyu.edu/comlex/refman.ps`, Stand 26. August 2004.

[McCormick 2002] McCormick, Susan (2002). *The Structure and Content of the Body of an OLIF v.2 File*. OLIF2 Consortium. `www.olif.net/documents/specificationFeb2002.pdf`.

[McCormick et al.] McCormick, Susan M., C. Lieske und A. Culum. *OLIF v.2: A Flexible Standard for Language Data Exchange*.

[McLaughlin 2001] McLaughlin, Brett (2001). *Eine Einführung in XML*. `www.oreilly.de/artikel/xml_einf.html` Stand 12. September 2004. Dieser Text ist das einleitende Kapitel aus dem Buch „Java und XML“ (Brett McLaughlin, Deutsche Übersetzung von Kalle Dalheimer 1. Auflage März 2001) in aktualisierter und leicht gekürzter Form.

[Näf 2002] Näf, Michael (2002). *Einführung in XML, DTD und XSL*.

[Ooi 1998] Ooi, Vincent B.Y. (1998). *Computer Corpus Lexicography*. Edinburgh textbooks in empirical linguistics. Edinburgh University Press, Edinburgh.

[Parsons 1994] Parsons, Terence (1994). *Events in the Semantics of English. A Study in Subatomic Semantics*. Nr. 19 in *Current studies in linguistics series*. The MIT Press, Cambridge, Massachusetts; London, England.

[Ritchie et al. 1992] Ritchie, Graeme D., G. J. Russell, A. W. Black und S. G. Pulman (1992). *Computational Morphology. Practical Mechanisms for the English Lexicon*. ACL-MIT Press series in natural-language processing. MIT Press, Cambridge, Massachusetts; London, England.

[Rohen Wolff et al. 1998] Rohen Wolff, Susanne, C. Macleod und A. Meyers (1998). *COMLEX Word Classes. Manual*. Computer Science Department, New York University. Prepared for the Linguistic Data Consortium, University of Pennsylvania. `nlp.cs.nyu.edu/comlex/manual.ps`, Stand 26. August 2004.

[Sevenich 1999] Sevenich, Richard A. (1999). *Compiler Construction Tools*. `www.linuxgazette.com/issue39/sevenich.html`.

[Thurmair und Lieske 2002] Thurmair, Gregor und C. Lieske (2002). *Lexical Exchange Formats – DXLT and OLIF*. The Burlington Hotel - Dublin (Ireland). Workshop of the Localisation Research Centre (LRC), in co-operation with the 21st International Unicode Conference. `lrc.csis.ul.ie/IUC21Web/Presentations.htm`.