View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Publications at Bielefeld University

Thorsten Trippel

The Lexicon Graph Model
A generic model for multimodal lexicon development

Universität Bielefeld
Fakultät für Linguistik und Literaturwissenschaft

Printed on ageing resistant paper according to ISO 9706
Gedruckt auf alterungsbeständigem Papier nach ISO 9706
Published as a PhD thesis at Bielefeld University, Germany
Veröffentlicht als Dissertation an der Universität Bielefeld, Deutschland

This work has been published with the same pagination up to the appendix (p. 255) in a book by the same title published by:
AQ-Verlag, Saarbrücken,
ISBN 978-3-922441-90-8
Additional resources can be found at http://www.lexicongraph.de
Thorsten Trippel
2006

To Christel Trippel †

Preface

At the time of starting my studies, I never thought of writing about lexicons some day. I never imagined that someday I would look at lexicons, or analyze their structures and think of a suitable model for them. Now, a bit later, I can look back on my way towards this field of interest and see, that I really enjoyed the trip into the study of lexicons.

A word of thanks

Obviously a work such as this would hardly be possible without strong support from many different individuals and groups. I would like to say thank you to those who have helped by giving me some form of encouragement, feedback or advice along the way. I would also like to mention some explicitly.

Firstly there are my parents Erwin and Christel, the latter not seeing the completion of this work while dwelling on this earth, but I am sure she is aware of my gratitude in the place she resides now. I am grateful to my sister Birgit, who sometimes experienced my impatience when facing another deadline. A special thanks to the Wächter family, my aunt and uncle, with my cousins, who were always close and offering words of encouragement.

Secondly there is my 'university family', which extends to the *Computational Linguistics and Spoken language* working group at Bielefeld University and the colleagues of the research group *Text Technological Modeling of Information* of the Deutsche Forschungsgemeindschaft (DFG) at different locations. It has been great fun working with you— at least most of the time

— and I hope there is more to come. I owe special thanks to: Dafydd Gibbon for the opportunity, and for supervising me and — besides his words of encouragement and criticism — offering a variety of new ideas, insights and perspectives; Ben Hell who has been around for almost the whole time of my research, for giving numerous pieces of advice in the computational field and even discussing *trivialities* such as running programs remotely; Kathrin Retzlaff for her support on the administrative and human side of the working group. The people working also in the fields of multiple modalities, I have to thank, esp. Alexandra Thies, and the former members of the working group Karin Looks and Ulrike Gut. Others have been working on various aspects of the corpora, and I owe them my gratitude for providing me with material: Maren Kleimann, Morten Hunke, Sophie Salffner, and Sandrine Adouakou. In the computational linguistics working group, I owe special thanks to Alexander Mehler for his encouragement and interesting discussions, providing me with additional ideas and and insights; Jan-Torsten Milde for the TASX-annotator and cooperation in creating the TASX format; Felix Sasaki, who not only enlightened me with language acquisition studies based on a survey of his children; Andreas Witt who constantly made me think of the application of data vs. document centered data modeling. A good 'kickoff' was provided by my former colleagues in the final phase of the EAGLES and Verbmobil projects, who contributed significantly to my decision to start with this project, namely Silke Kölsch, Inge Mertins-Obbelode and Harald Lüngen.

Thirdly there are my friends all over the world, representing different disciplines and backgrounds, who have helped me to reflect upon new ideas. I won't name all of you here, most of you would not even like to be thanked in such a place. However, thank you.

Fourthly — and I should probably stop before listing a phone book — my teachers, educators, and youth leaders who helped me develop my personality and interests. I hope you are aware of the profound impact you have had on me.

Editorial feedback and comments on different issues related to the manuscript were provided by Nigel Battye, Dafydd Gibbon, Mark Mathias, Alexander Mehler, Felix Sasaki, Alexandra Thies, and Daniela Wächter. Thank you for your time and efforts. I owe additional gratitude to my publisher Erwin Stegentritt and the series editor Guido Drexel for their valuable feedback, recommendations and patience. At the end, preparing a manuscript always takes longer than expected.

To all of you, I wish to personally express my gratitude and the way I feel. Although a lot of people contributed in one way or another, any remaining errors, flaws or gaps are solely my responsibility.

This work includes research funded by the Deutsche Forschungsgemeinschaft (DFG) within the Forschergruppe *Texttechnologische Informationsmodellierung*, as well as various other projects such as the BMB+F funded project *Verbmobil* in the Bielefeld lexicon working group, the Volkswagen Foundation project *DOBES* in the EGA sub-project, and in Probral project by the Deutscher Akademischer Austausch Dienst (DAAD). All of them provided funding and research opportunities and collaboration opportunities that influenced this project. Software AG (Darmstadt), provided a licence of the Tamino XML database, that was used for some parts of this project.

Technical remarks

In many scientific articles and books, there is a clear distinction between literature found in the bibliography, software that is mentioned — possibly with reference to the vendor — and standards. However, these resources are treated equally in this work, i.e. the bibliography is a *list of resources cited*, including material from the web, software and corpora, but also literature. This decision was made based on the use of resources and standards which appear in the same way in the discourse as the literature, although the status may be different. For volatile resources, the date of the last check was mentioned. If the document has since changed, the newest available version was reflected in December 2006.

A reference version of this book was handed in at Universität Bielefeld, Fakultät für Linguistik und Literaturwissenschaft as my dissertation.

Contents

List of Figures

List of Tables

Part I

The formal structure of the lexicon

The formal structure of the lexicon

The formal structure of the lexicon is discussed in the first part. It is based on requirements defined for general language resources and lexicons. A formal model of the lexicon is introduced with the Lexicon Graph model, which is also implemented and evaluated.

1

Introduction

The goal of this thesis is to provide a model and framework for lexicons that can be corpus based and contain multimodal information. The focus is more from the lexicon theory perspective, looking at the underlying data structures that are part of existing lexicons and corpora. Before getting into the details of this study, it is necessary to define *the lexicon*.

The lexicon is the place to turn to when unknown words, strange orthography, or pronunciation make one curious. Lexicons are available in different formats and versions, for different purposes and for multiple applications. They cannot be restricted to one medium but lexicons may contain other media. Widely known are pictures and diagrams, but electronic versions also contain audio and video samples. Some users of these lexicons are not even human, but computer programs that need to access lexicon databases for human-machine communication. Talking about lexicons in the field of *Natural Language Processing* requires a definition of both, *lexicon* and *Natural Language Processing*.

The term *lexicon* in linguistics and artificial intelligence is used in different ways, including traditional print dictionaries in book form, CD-ROM editions, Web based versions of the same, but also computerized resources of similar structures to be used by applications. These applications cover systems for human-machine communication as well as spell checkers. The term *lexicon* in this work is used as the most generic term covering all lexical applications.

Another subject field often named lexicon or included in the discussion of lexicons is the so called *mental lexicon*, which is the representation of lexical items in the human brain. The latter is excluded from the research here, not because it seems uninteresting, but because it is not in the center

of the structures used for the representation of the lexicon as discussed here. Also it is not clear what mental lexicons look like and what structures they have. For a detailed discussion of the *mental lexicon* see Handke (1995).

The primary focus of this research is the lexicon used in *Natural Language Processing* (NLP) and the generic structure of lexicons. NLP refers to the field of processing language intended for human communication, contrary to computer programming languages or languages of logic. The lexical information processed is strongly related to language as it appears in natural contexts, such as a person writing or speaking to another person. Information on the items used in communication can be found in print dictionaries, as well, as in automated tools implemented in computer programs. However, the required information can be the same.

1.1 The problem of the lexicon

Some questions in the lexicon context have not been solved previously, among them are:

- How is it possible to combine lexical resources, even if they do not describe similar lexicons?
- What structures are necessary to account for ambiguity in the lexicon on different levels, such as meaning, surface representation, grammatical form, etc?
- How can multimodality be represented in the lexicon?
- How can a lexicon be created based on multimodal data?
- How can all data categories of a lexicon be used as headwords for search and retrieval in a lexicon without implementing new lexicons?

The research described here starts from the premise that lexicons are different in content and structure but can be analyzed in the same way. Using this description, a unification process can be thought of without losing information from any of the original lexicons. The resulting lexicon is a declarative lexicon as used in Feature Structures (Shieber (1986)) or modeled by DATR (Evans and Gazdar (1996)). The underlying structure of these lexicons are directed graphs, and a generic model for describing lexicons is the Lexicon Graph Model introduced in here. This Lexicon Graph is rather similar to the Annotation Graph Model as formally introduced by Bird and Liberman (2001) and hence can be treated in similar ways. In the corpus based creation of lexicons there is an implied relation, which also relates to the formal Annotation Graph, but the formal relation needs to be made explicit. The

connection of lexicons with other language resources such as corpora is evident in modern lexicon development. Gibbon (2000) (p. 30f.) for example discusses lexicon development based on corpora. He points out that corpora are used as the input to a lexicon system and that this results in different lexicon formats. Examples for different lexicon formats are printed lexicons, hypertext lexicons, pronunciation or context lexicons (such as *n-gram* lists or concordances).

The graph structure in annotations, described by Annotation Graphs, as well as the graph structure in the lexicon is especially important when other modalities and multimedia events are included. The reason for this is that non-written items for a lexicon such as audio signals or concrete objects other than written characters require a connection by pointing to abstract or other concrete items.

The present work discusses lexicon development. It integrates established lexicon data formats, lexicon structures and corpora. Corpora cover textual corpora and multimodal corpora, i.e. related to audio and video recordings of spontaneous speech, as described by Gibbon et al. (1997a) and extended to non-audio modalities in Gibbon et al. (2000a). This work does not address a specific strategy for avoiding and disambiguating problematic cases but provides a model of integrating and representing lexical relations regardless of a possible ambiguity. Problematic cases in traditional lexicon theory are addressed, for example, by Pustejovsky (1995) for polysemy resolution in the *Generative Lexicon*. The Lexicon Graph Model, however, allows the inclusion of language information as it appears in natural language use, without having to distinguish cases and avoiding pitfalls of ambiguity resolution. This work also does not intend to explain some of the syntactic concepts within the lexicon as in LFG (Kaplan (1995)) or including grammatical information in the lexicon as in HPSG (Sag and Wasow (1999)) but allows to include knowledge about lexical items by means of knowledge representation and metadata specification. What this work does provide is a way of representing ambiguous information.

1.2 Methodology

For solving the problems mentioned before the following methods are used:

1. Existing lexicons are analyzed and described (Chapter 2). This includes different kinds of lexicons, both from the field of machine applications and human usable lexicons.

2. The Lexicon Graph Model is introduced by discussing problematic cases of lexicography and offering a model solving these problems. This model is then implemented and applied to different lexicons and a lexicon workbench is developed (Chapter 3). The Lexicon Graph also allows for using all different sorts of lexical information as the headword for the rendering of individual lexicon entries. The Lexicon Graph Model is evaluated (Chapter 4).
3. Annotation Graph in the form of concrete annotations and the Lexicon Graph are compared (Chapter 5). Part of this comparison is also their descriptions in the form of metadata, which is discussed in Chapter 6.
4. The lexicon model, together with existing annotations and metadata descriptions are used in a sample application, a multimodal concordance. This multimodal concordance includes the representation of a simple lexicon and a relation to other media (Chapter 7). Another application is the automated extraction of complex lexical generalizations from multimodal annotations (Chapter 8).

This work is divided into two parts, the discussion of the formal structure of the lexicon, covering the first two list items, and the application covering the remaining two list items. It constitutes a work of computational applied linguistics, rather then an application of computational linguistics.

2

Describing different lexicons

In this chapter existing lexicons are described and analyzed by the means of structural descriptions of these lexicons. They are also evaluated according to a number of requirements to language resources in general and lexicons specifically. This serves the purpose of finding relevant structures for a more generic lexicon description which is described in later chapters. The structure of lexicons can be discussed from different perspectives, e.g. from the perspectives of lexicography, lexicology and lexicon theory. Gibbon (2000) defines these areas as follows:

Lexicography deals with the creation, i.e. the design and construction of lexicons for practical use, for example, writing dictionaries and encyclopedias. Lexicography is a part of *applied linguistics*

Lexicology is the field of interest of studying and describing lexical information as part of *descriptive linguistics*

Lexicon Theory is the study of universal properties, such as the formal properties. Lexicon theory is part of *theoretical linguistics*.

The present work approaches the lexicon from the lexicon theory perspective, though sometimes these fields of interests overlap and are not clearly distinguishable. For example, the detailed analysis of existing lexicons conducted in this Section is also connected to the field of lexicology. The lexicon analysis here is based on the differences of lexicons used in different areas. Starting from the requirements for different lexicons, a lexicon structure description is introduced, followed by a detailed analysis of some lexicon use cases.

2.1 Objective

Lexicon design is usually driven by two main forces: firstly lexicon development for human users, for example, by dictionary publishers, and secondly the development for computer programs, here summarized by the term *system use*. Systems in this sense are computer programs that process natural language, for example machine translation, spell checkers or human-machine dialogue systems.

2.1.1 Lexicography for the human user

One approach to the lexicon is driven by human user's needs to have a lexicon at hand. Among the lexicons for human users there are

- Print dictionaries
  - Translation dictionaries, such as bilingual or multilingual dictionaries
  - Encyclopedias, which present different senses for lexical entries (Pustejovsky (1995) groups them into the class of sense enumeration lexicons (SEL))
- Electronic dictionaries
  - CD-ROM dictionaries, which are mostly print dictionaries in electronic form (see Schmidt and Müller (2001))
  - Dictionaries on the web, which are not web accessible CD-ROM dictionaries but, for example, dictionaries designed for the use on the web and which can be edited interactively (see for example `http://www.wikipedia.org`)

These lexicons all share the feature of being intended for human use:

1. Their layout and presentation is intended for human readability, i.e. the layout uses typographic aids for reading structure and comprehension
2. Their information is presented in a human readable way, i.e. sentences and words instead of abstract or symbol driven data structures, and
3. Their content is intended for human perception, i.e. human interpretable information such as semantic concepts according to basic knowledge.

Lexicons in an electronic format are seen as an extension of human readable lexicons in this area (for example, by Storrer (2001)). As these electronic lexicons are barely more than a different edition of print dictionaries, they are processed in the same way, including human editing.

The field of human directed lexicons is most widely spread in the linguistic community, resulting from a long tradition of producing dictionaries

both for research purposes and for commercial use by publishing companies. These dictionaries contain elaborated structures on different levels such as the entry level, the organization of entries and the cross reference structure. The structures are summarized in the microstructure and macrostructure, as used for example by Hartmann (2001) for certain characteristics and a structure referred to as mediostructure, mesostructure or reference structure (see for example Gibbon (2002b)).

2.1.2 Lexicography for system use

Modern computer applications based on human machine interaction use lexical models for interpreting commands, recognizing words and patterns. Examples for these range from grammar-checkers and spell-checkers in text processing systems, speech recognition and synthesis systems, to complex dialogue systems combining features of all of them.

Lexicons for systems can be distinguished into different classes on procedural grounds:

Recognition based lexicons: lexicons that are used in recognition based systems, for example speech recognition. These are based on parameters from signal processing and have no resemblance to human usable information. They are subject to the signal processing units, which might, for example, be based on statistical models as Hidden Markov Models, formant measurements, etc.

Word form based lexicons: lexicons that are based on some form of word for example:

- Spell checking lexicons consist of a list of known words. In the case of a word not being in the wordlist a spell checker can provide alternatives. One way of providing possibly correct forms in a context is by algorithms calculating the distance between known words and existing words, either based on phonetic similarities or other distance operators. Instead of these distance operators, statistical processing depending on character sequences or based on word distributions would be possible. These two components — the wordlist and the distance measure or statistic rules — are part of a spell checking lexicon for system use. Although they could be human readable in principle, the rules might not be human readable.
- Inflectional lexicons consist of a list of possible inflections for a given lexical item, allowing for computer systems to access the different forms according to their predefined contexts. One example of such a lexicon is the Bielefeld Verbmobil Lexicon (2000).

Synthesis based lexicons: lexicons as one of the two components of a system which generates language, which are described for example by Chomsky (1970) as

1. *Categorical component* rules of a context free grammar, and
2. A *lexicon*, consisting of lexical entries with specified features.

2.1.3 Combining approaches

Lexicons for human users and for computer systems are not as different as it may seem. At least structurally they can be described in similar terms and information can be shared among both if the information is represented in a suitable form.

Lexicon synthesis in a combined approach uses two rules:

1. Lexicon synthesis based solely on information from a corpus: this is created from the
   a) Distribution of annotation items, for example words. This is a statistical approach.
   b) Relation of annotation levels, for example relating word level and phonetic level annotations to gain transcriptions for words in a time logic approach, cf. Section 8.2.1.
2. Lexicon synthesis with the help of expert provided rules consisting of
   a) Lists of known units, e.g. known morphemes for morphological parsing and automated extraction from words
   b) Compositionality rules of certain annotation levels, e.g. *words* combine to *phrases*
   c) Semantic analysis by lexicographers in a stand-off fashion (see McKelvie and Thompson (1997)).

A post-editing and evaluation phase, both making use of automatic procedures and experts, add to this strategy. By a combination of corpus based lexicon generation, expert rules and post editing the advantages of all strategies can be taken into account and a maximum of accuracy and completeness can be achieved.

Additionally, the lexicons can receive different ‘views’ (see Schmidt and Müller (2001)) in order to provide for different human users and their requirements. Lexicon used by systems can be seen as another of these ‘views’.

Schmidt and Müller (2001) mention at least three different views:

1. The *lexicographer’s view*, which is the full structured view for the lexicographer working on the lexicon;

2. The *publication view*, being the layout and visual presentation of the lexicon product;
3. The *typographic view*, which does not correspond to the actual layout but to the typographic conventions used to represent the lexicon.

2.1.4 Requirements for a lexicon

A combination of automated and manual approaches and the openness to different views results in a number of requirements for lexicons. In print dictionaries requirements can be used to describe the expected content of a lexicon. However, the requirements are only a first component of an approach to a lexicon model.

Coverage: Corpus based lexicography is intended to include all levels of linguistic information available in the annotation. Consequently, a lexicon model in itself needs to be independent of annotation classes. The independence of annotation classes also covers the fundamental openness to modalities other than written words, both on the input and output level of the lexicon.

Corpus induced lexicons cannot have a wider coverage of a language and its registers than the underlying corpus. The different coverage will therefore not be discussed further in the present work, although the size of the corpus is discussed. The lexicon which is discussed is not generative in the sense of a generative grammar, because it is intended to be corpus based.

Nevertheless, it is intended to be usable in generative contexts such as synthesis systems.

Reusability and portability: A major requirement for state of the art language resources is reusability for later uses and portability to different contexts. The term *language resources* covers corpora, lexicons and applications for natural language processing (see for example Fontenelle (2000), p. 229). For human usable lexicon systems Büchel and Schröder (2001) define requirements for reusability. They restrict reusability to portability onto different computer platforms, which could be in succession to each other. Büchel and Schröder do not address portability in the context of different programs (referred to as *interchangeability*) but seem to refer only to backwards compatibility with older versions of proprietary operating systems and applications.

Concept vs. sign orientation: Terminologists and translators develop their dictionaries of languages for special purposes according to *concepts* (see

Wüster (1991), Sager (1990)). Concepts correspond to mental images of real *objects* according to Saussure's sign model (1916). Lexicographers on the other hand use *lemmas* (see van Eynde and Gibbon (2000)) as abstract representation of a class of *signs* corresponding to grammatical forms of a word.

Concept orientation seems suitable for highly restricted languages but problems with general language in machine translation show that it is not possible to define a general concept system that is suitable for every language and context (see Melby (1995)). Nevertheless, the problematic cases of polysemy, homonymy, homographs and synonymy could be tackled easily by referring to concepts. The problematic cases are defined as follows:

Homonyms: two different words share both orthography and phonology, but not the semantics and distribution. They are distinguished by enumeration or other distinguishing markers. In pronunciation lexicons this distinction is irrelevant. In general the distinction is not relevant for lexicons where the distinctive values do not occur.

Homophones: two different words share the same phonology, but not necessarily the orthography. In the other areas they can be treated just as homonyms.

Homographs: two words share the orthography, but not necessarily the phonology. This is a special case of homonyms.

Synonyms: two words share the semantics but not orthography and phonology; in concept based lexicon bases these are not distinguished.

A formal lexicon model should provide for all these hard cases and for human and system use.

The previously mentioned cases of polysemy, homonymy, homographs, and synonymy require special structures. Polysemy, homonymy and homographs are not distinguished here at the moment as the line between them is not always clear cut and all refer to semantical differences that share an orthographic form. In a system that is concept-oriented, all synonyms are grouped under one concept, while in a system that is form-oriented all polysemous words, homonyms, and homographs are grouped either in the same entry or close to each other.

This affects the problem of disambiguation both in contexts of human-usability, where users may guess from a context given in the lexicon entry what kinds of ambiguous meanings could be required and for systems where from a given structure the suitable use has to be extracted. For example *recognition systems* which are trained according to a suitable cor-

pus reflect the amount of training and adjustment of the corpus in the lexicon, i.e. structures that are given prominence due to corpus analysis are preferred to others if a full set of constraints cannot be provided for disambiguation. Adjustment for a specific situation on the other hand is not what is wanted for a lexicon. In speech recognition and machine translation it is widely known that the application of new contexts/corpora for words result in unsatisfying outcomes (see for example Melby (1995)), though the usability of a lexicon should be general enough to allow its use also for word forms and meanings that were not directly included in the corpus. This form of robustness and generality is a prerequisite for a lexicon. Robustness here refers to the possibility of using a lexicon even in the case of incomplete or deviating input.

Coding of lexicons : Independent of the sign or concept orientation is the lexicon encoding. Lexicon coding can be accomplished in numerous formalisms. Three of them are sketched out here, which have been implemented.

XML coding: Representing lexicons in a reusable way has been attempted in different contexts, both for terminological applications and for lexicography. This — somehow artificial — distinction results from different traditions. Terminologists belong to a tradition of technical writers and translators, while lexicographers come from a tradition of language teaching and description. For terminological lexicons and lexicographic lexicons there are proposals for encoding them in an XML based syntax.

The *Text Encoding Initiative* (TEI) (see Sperberg-McQueen and Burnard (2001c)) define a way of coding existing printed lexicons but their proposed encoding merely tries to resemble the typographic structure. Schmidt and Müller (2001) agree that it is unsuitable for content representation and structurally inappropriate as a lexicon formalism. For terminology there is the MARTIF standard (ISO 12200:1999 (1999)) that is much more suitable for the document grammar and which has resemblance to the TEI recommendations on terminology coding (Sperberg-McQueen and Burnard (2001a))

Coding in relational databases: Relational databases provide a way of efficiently storing large amounts of data in a structured way. This data can be retrieved according to application needs. For this purpose, data items are stored in tables, each data item being a vector of attributes to an identifier. The data items can be accessed by the name of the table and the identifier to retrieve the attributes. The strength of the relational database is that different tables can be merged by a

common attribute such as a common name. As Codd (1970) points out, every relational database can be joined into one huge table.
In a lexical database, every line represents one lexical entry. This structure is relatively flat and connections from one column to another cannot be expressed formally, neither in such as flat table nor in fully redundancy-free systems of tables, which are called Category 4 databases in Codd's terminology. Although the efficiency of these databases in an application is quite high, the linguistic relations between some of the columns constitute a major drawback because they are not explicitly defined, and all columns appear to have the same status if they are not used as primary, i.e. identifying fields or obligatory vs. mandatory fields. Hausmann and Wiegand (1989) point out that there is an isomorphism for hierarchical *microstructures*, which are rather abstract, to concrete flat *microstructures* that can be stored in relational databases. The structures involved are the focus of the whole Section 2.2 and hence will not be discussed any further at this point.

Inheritance hierarchy coding: There are different ways of coding inheritance hierarchies. DATR (see Evans and Gazdar (1996)) is one of these formalism providing for lexicon coding with inheritance hierarchies, defaults and overriding of defaults. Complex lexicon structures can be modeled using this formalism, and implementations such as *zdatr* (Gibbon and Strokin (1998)) enable the use of the system in lexicon systems. It is based on an attribute-value formalism for lexicon entries and allows references to other lexicon entries.

2.1.5 Excursus: Portability of language resources

One requirement for language resources such as a lexicon was defined in the previous section under the term *reusability and portability*. The field of reusability is rather complex. As it is a key feature of the present approach to a generic lexicon model it is necessary to address the issues that are part of it. Bird and Simons (2002) and Bird and Simons (2003) define 7 dimensions of portability, namely

Content: for working on more than one resource, researchers rely on comparable structures. As long as the structures are very similar this does not result in major problems, but data categories and descriptions of language resources can be the same in different resources, having different meaning, or, vice versa, having different names but identical meaning.

To prevent this, either a *structured semantics* of encoding of resources is needed or a restricted set of possibilities for resource encoding can be defined. The latter is called *controlled vocabulary*. The structured semantics on the other hand can be based on a common ontology.

Format: working on resources requires tools operating on the data structures of the resources. If these data are stored in proprietary data structures, the data are unusable without specialized software.

Discovery is the first requirement in the reuse of existing resources as they need to be found, located and evaluated for appropriateness.

Access refers to actually getting and using the resource.

Citation: language resources sometimes do not receive a proper citation as they are not as widely used as other types of reference materials, such as articles and books.

Preservation: in the electronic medium, data are extremely volatile in comparison to books printed on paper. Networks are volatile per se, hard discs have a life expectancy of few years, CD-ROMs and DVDs of a few decades, storage media get out of use and hardware becomes unavailable.

Rights: intellectual property rights and ethical issues restrict the use of language resources in some contexts.

The requirements for language resources implied here are manifold. The portability of content, for example, should not be underestimated. Different theoretical backgrounds provide a major obstacle for controlled vocabularies, and generic or common ontologies are not available. If there is an ontology, it is subject to change over time, or some areas are missing. With the *General Ontology for Linguistic Description* (GOLD, see Farrar and Langendoen (2003)), an ontology has been designed for linguistics, which is promising but currently not fully specified for all linguistic areas. The problem of content description is related to the XML-world problem of the semantics of XML, i.e. the meaning of XML-tags and structures in themselves, and how this meaning is specified. A more detailed discussion can be found in Renear et al. (2002).

Another portability area is the data format. A solution could be to use only open formalisms such as XML. However, this can be only part of a solution, because it is not sufficient to use an open syntax if it is not documented. In the XML context, documentation of the syntax means to provide and specify the document grammar and to describe the elements sufficiently for a researcher with the background information necessary to create their own tools.

The discovery of language resources is possible using a description of the resource in terms of content, type and format. To enable the discovery allow-

ing a description of the resources it is to be described according to metadata standards (see Section 6.3.2) which is to be provided for public use.

Accessing data offers a problem for researchers to get a hold of language resources, for example, for restrictions due to ethical reasons. Problems with accessing a resource can be avoided providing full documentation of the resource structure and making the resource accessible for all user needs within the limits of access restrictions. The same applies to copyright restrictions.

A technical problem is the issue of preservation, which can hardly be solved in a non technical way, for example by copying media regularly or by establishing persistent and portable media and data formats. This could allow reusability over time and system boundaries.

Reusability is a major reason for using an open approach. An open approach to language resources is one that allows accessing the resources without special proprietary tools and programs. This includes the use of publicly available specifications and non-restricted access to the required information for accessing the resources. It only assumes the implementability of algorithms and a corpus which needs to be transferable into an interchange format. An open interchange format provides an open standard which is easy to implement and to document.

Reusability requires:

- Open and easy to process data formats, i.e. character based data formats which are well documented in order to enable users at a later point of time to adopt it to their specific needs;
- Open and well documented algorithms for processing the data, which enables skilled programmers — even if the original software does not work on a specific computer system — to re-implement the system if required, in a very short period of time;
- Open and freely available software, which can be run and if necessary adapted for the use on different computer systems;
- Description of the content, structure and property rights;
- Storing the resources and descriptions in accessible and reliable repositories, such as libraries or archives.

Any lexicon developed should provide for being reused by other systems and lexicons by adhering to existing standards, which results in a character based data format such as ASCII or Unicode. Büchel and Schröder (2001), Storrer (2001) and others stress the value of XML in this context, without distinguishing the character of XML as being a *document syntax* and not a *data format*. Although for certain reasons XML is the format of choice, the document grammar is arbitrary as long as it is well documented and the data

are accessible from a system in a well defined way. If this is true it should be possible to transform any such file into an XML file that provides for the same structures, at least as an interchange format.

2.1.6 Seven requirements for modern lexicography

Storrer (2001) defines seven requirements for modern lexicography based on the assumptions of hypertextuality in a lexicon. Although, she refers to dictionaries in the sense of human processed ones from a hypertext perspective, most requirements are useful for general computational lexicon development:

Data modeling according to linguistic units and structures: the data model explicitly contains information on *surface structure*, *syntactic structure* and *semantics*.

Data collection according to lexicographic work flow: traditional publishers work flow for dictionary collection is oriented towards sets of initial letters, especially if an existing dictionary is to be revised.

Transparent relation between lexicographic sources and description: computational lexicons can be combined with concordances.

Flexible user interfaces according to user groups: as different uses of lexicons require different data, computational lexicons can provide the inherent information according to the applied needs.

Extensibility: printed lexicons describe a lexicon database at a given time, while the extension by the lexicographer is not limited by time constraints.

Inclusion of multimedia: certain media may be useful for the information retrieval by certain users.

Assessment by backchanneling: related to the extensibility issue, but listed separately is the issue of assessment. A computerized lexical database which is available on a central architecture can be assessed by a backchannel from the user to the lexicographer or lexicographic system, providing corrections and extensions.

The requirements can be discussed in the context of corpus based lexicon creation. The data modeling according to linguistic units and structures is not highly formalized in conventional print dictionaries. A reason for this is that human users can retrieve information from less structured data.

The data modeling according to lexicon data collection and work flow has some drawbacks:

- Words occurring in the process of data collection cannot be inserted at the work flow position if a particular word does not fit into the order — which is rather likely.
- Changes are not accomplished in short time range, e.g. upon the discovery of an error, but according to the position in the work flow

In an automated lexicon collection based on corpus information this is not true as the work flow in the lexicon synthesis is related to the sequence of words which is sent to the processor. Furthermore, for corpus induced lexicons real-use examples can be taken from the corpus for every lexical item; the transparency of the relation of sources and description is maintained. Additionally, by means of computational lexicons more source data can be processed on a shorter timescale.

A structured data storage also allows the flexible use of the lexicon data by defining flexible access structures. For human agents this may result in fully readable texts, while for systems only specific structured query results are necessary. A central database architecture additionally enables the extension of lexical databases according to the lexicographer's requirements for the dissemination of the data. Extending the database is also possible to different media. Storrer includes the aspect of multimedia, such as pictures, sound, and video. The use of these media types seems to be restricted to the human user but in fact applications of system dictionaries can use applicable media as well, practiced, for example, in text-to-speech system with diphone bases (see Dutoit (1997)) — which are lexicons of prerecorded diphones which are headworded by phoneme combinations.

An example of the integration of the backchannel from the start of a lexicon is the so-called Wikipedia (`http://www.wikipedia.org`). Wikipedia is an interactive lexicon project based on the Wiki technology. In the Wikipedia every user is allowed to add and modify lexical entries, which can be reviewed by other authors. This can be changed to validate the user (system or human) assessment by a system and/or a lexicographer.

2.2 Lexicon Structure Description

Describing and comparing lexicons requires the analysis of common structures and other features of a lexicon, besides the evaluation according to the requirements specified for a given lexicon.

According to Hartmann (2001) the following structural features can be distinguished:

1. The lexicon *microstructure*, which is the description of the lexicon entries, sometimes these are called lexicon articles or records;
2. The lexicon *macrostructure*, defining the sequence of lexicon entries and the access to the entries;
3. The lexicon *megastructure*, which is the lexicon body with all the lexicon entries in their microstructure, ordered according to the macrostructure plus what he calls the *front matter* and *back matter*.

The terms *front matter* and *back matter* refer to additional information such as metadata, grammatical rules in appendices, etc. The definition of megastructure is hence on a different level as the other structures: the megastructure defines not an inherent structure but a structure that is only expressed by the linear representation of a lexicon, i.e. upon bringing the macrostructure together with the front and back matter, unlike the microstructure and macrostructure. Hence megastructure here is only used in terms of the rendering of a dictionary.

Hausmann and Wiegand (1989) distinguish the following three structures:

Macrostructure: the set of ordered headwords
Microstructure: the structure of the entry
Mediostructure: the ways of defining cross-references.

A structure related to the mediostructure is the lexicon *mesostructure*, which defines the interrelation between the lexicon entries and between lexicon entries and other information not coded in lexicon entries. The mesostructure covers the information of the megastructure explicit in structural terms, wherever it is coded. The mesostructure is, for example, introduced by Gibbon (2002b). The mesostructure covers aspects of Hartmann's megastructure such as grammatical rules, which can be part of the mesostructure and are sometimes found in the front matter of a lexicon. Though this is a special type of reference, other cross-references, such as references to other lexicon entries, are also included which are covered by the mediostructure in Hausmann and Wiegand (1989). The mesostructure is therefore more general than the mediostructure.

As the most general structure description without reference to specific renderings and restrictions, the *microstructure*, *mesostructure* and *macrostructure* are taken into consideration for further analysis. These structures partially depend on each other, though they can be discussed separately.

2.2.1 Microstructure

Lexicon databases can contain a lot of information, for example, on orthography, pronunciation, definition, morphology, and word class. Although the number of different information classes in a lexicon can be large and — especially in some print lexicons — the information classes are used inconsistently, there is a structure describing the *lexical data categories* called the *lexicon microstructure*.

The microstructure defines:

Lexical data categories, i.e. all categories that can be used inside a lexical entry. In some cases the values of categories result in the existence of other categories, for example the categories for `auxiliary verbs` are only filled if the word class is `verb`. This relation is part of the mesostructure discussed below, but this structure depends on the existence of the relevant data categories;

Order of data categories, which describes the sequence— if any — of data categories. In most cases, the data categories are given in a sequence, i.e. in a lexicon vector which helps potential users to locate a required data category. If the data category of the lexical information is typed or marked, a sequencing is not obligatory as an application or user can identify the relevant information by the tag;

Content model of data categories defines restrictions to the content represented in a data category. The content restrictions can be applied both in terms of encoding standards and the use of a controlled vocabulary or predefined structures. For example, if one data category contains a calendar date, the format of these dates is usually standardized, using a language specific standard or a predefined convention. Nevertheless, the structure of the content is defined. The same applies to the encoding of special information, such as the use of a special character encoding for information — for example IPA symbols for phonemic transcriptions — or the use of a controlled vocabulary, i.e. a closed set of options such as language or country names according to some convention.

The data categories are usually given as a list, which often implies the order. The content model is only sometimes formally defined; mostly it is given in prose description or assumed to be obvious in a language context, such as the format of dates. Dates are already differently encoded in closely related communities such as German, British English, American English. As these categories are usually not really fixed if extended to other areas, they need to be defined explicitly.

2.2.2 Mesostructure

The data categories represented in the microstructure do not exist independently of each other but refer to other lexical entries. For example, categories for `synonyms` and `antonyms` refer to other entries of the lexicon they constitute cross-references.

Some data categories are related to each other, i.e. they belong to a similar domain or subfield. In addition to the specification of these relations there are some references to information not in the lexicon, such as bibliographical references and links to corpora, which are all part of the *mesostructure*.

The mesostructure of a lexicon consists of the following components, which describe the relations:

1. Between data categories, allowing possible dependency and hierarchical relations. As used before in the discussion of the microstructure, an example is the data category `auxiliary verb` being relevant only for `verbs` in languages such as English or German and other Indo-European languages. The mesostructure describes that auxiliary verbs are a subtype of verbs, so these data categories are related.
2. Between lexical items which comprise the lexical relations constituting cross-references to other items in the lexicon. However, cross-references are not only available in form of synonyms, antonyms, etc., but also in other forms, such as explicit, defined references, thematic similarities (*semantic fields*), or any other reference.
3. To the *front matter*, often containing sketch grammars and inflectional tables, which are referred to using some sort of reference marker.
4. To external information, such as reference to a corpus or to further sources.

A special case of the relation of metadata categories can be seen in *inference lexicons*, where lexical information is deduced from other lexicon entries. Beside inference lexicons, the mesostructure is usually described only in prose and latently in data categories describing lexical relations or by marking references in prose with the help of keywords such as the following: `see also`, `comp.`.

2.2.3 Macrostructure

Locating lexical items in the lexicon is described by the *lexicon macrostructure*. This structure relies on the data categories as defined in the microstructure as cross-references of the mesostructure rely on the possibility of using the macrostructure to access the referenced terms.

The macrostructure covers the treatment of spelling variants, handling of numbers, and the distinction between lexical entries and can be described by the sorting of lexicon items, headword selection, and access structure as discussed in the following paragraphs.

Print dictionaries are usually sorted according to the order of an alphabet, at least in European languages. Although it is possible to define conventions for character based sorting also for other writing systems, the order is not conventionalized. One example are Japanese systems that do not use a character based order for sorting, but have different sorting strategies such as by the number of strokes needed for drawing a character.

The character order is not the only sorting criterion, but also the direction of applying the sort algorithm. While most dictionaries start from the beginning — may it be a script system from left to right, right to left or top down — in reading direction, other lexicons, such as end-rhyme lexicons, could use a different strategy. For modern computer applications the sorting is irrelevant for storing because the access structure is independent of a sorting strategy, which depends on sorting algorithms.

Defining which lexical category to use for sorting and accessing the lexical entry is the concern of the headword selection process. Usually the orthographic representation is used, but it is also possible to use a sorting strategy according to the position in a concept hierarchy, i.e. according to a numerical or alphabetical identification represented as a lexical category.

Two headword selection processes are especially prominent, namely

Semasiological lexicons which aim at the semantics and properties of a word, starting form a surface representation such as its orthography. Most print lexicons — especially from a European language background — are semasiological lexicons, which are alphabetically sorted.

Onomasiological lexicons, which are sometimes called the *writers lexicon*, starting from the semantics and mapping meaning to a surface representation. Onomasiological lexicons are sorted by semantic fields or concepts. Typical examples are thesauruses and terminological dictionaries. These usually contain an alphabetical index for initially locating the concepts, for example, by means of a synonym or by a word in a different language.

These headword selection processes are rather prominent, resulting from a tradition of card based lexicon acquisition and print lexicons, using only semantics or orthography as headword data categories. In a computer based system, any data category can be used as a headword, resulting in a larger variety of different types of lexicons, generated from the same initial data structure.

The *access structure* to a lexicon describes possible ways of locating and selecting lexical entries. In traditional lexicons this is usually predefined by the sorting strategy, but other aspects aside from sorting are relevant as well. These cover

Handling of inflections and derivations: in highly inflected languages usually one form per wordclass is used to access a lexicon entry, for example, first person singular indicative present tense for Latin verbs, or nominative singular for German nouns. In languages with inflectional prefixes, one could as well sort the entries by the morphological stem. In rich full form lexicons the handling of word formation processes is strongly related to the headword selection.

Number-character combinations in words can either be accessed by the spelled out number, or by the position of the digit in the order of the characters. In character based access structures of computational applications, the treatment of number-character combinations is more relevant than the sorting structure, as the application needs to locate the appropriate lexicon entry, or users need to know about the treatment of number-character combinations to find them in a given lexicon.

Non-standard representation, such as typing errors which can be used for the location of lexical items. The number-character issue is a special case of a non-standard representation. Another is the handling of spelling errors or uncertainties. A list of constraints for a lexical item can be used by the access function to cover deviating representations. These constraints themselves are a special kind of a lexicon, containing spelling and representation variants.

The access structure in computer based applications for human users usually provide an inexact or approximative search based on a replacement method or what is called *phonetic search*, for example, the SoundEx system. The SoundEx system is a method which identifies all vowels replacing it by a digit, the same with all spelling forms of bilabial consonants, using a different digit as for the vowels and applying the same method to labiodental, dental, alveolar, velar, and glottal consonants, respectively. This is not strictly speaking a phonetic search but a search by classifying letters into these seven classes and searching a database using these classes instead of the original characters. Another possibility is the identification of a word not quite matching a search key using a distance measure, such as the Levenshtein measure, in which a search key gets transformed into a known key. The one with the least steps of transformation is presented as the result.

2.2.4 Summary of lexicon structure descriptions

Three different structures for comparing lexicons were introduced, namely the microstructure, mesostructure and macrostructure. These structures are used for a qualitative comparison of lexicons stored in lexicon databases. Without the qualitative lexicon description a quantitative comparison by, for example, the number of words is hardly possible. A reason for this is that the selection of headwords, handling of variations and sorting strategies results in differing numbers.

2.3 Analysis of structures of existing lexicons

Existing lexicons can be described according to the structures introduced in the previous sections. In this section some prototypical lexicons are analyzed. As lexicons appear in different formats and are available in different forms, a distinction of *lexicon database*, a *database format* and a *product* will not be made. These are different things, as the lexicon database contains the lexical data categories directly, hence the analysis of the structure is straight forward. The same is true for a database format that allows certain structures. A product, on the other hand, is always based on lexical structures, although they may not be explicit in the representation. Usually it is not even possible without insider knowledge to access the structures of products directly. Nevertheless, the model of describing general lexicons is supposed to cover these structures as well; hence they are included in the analysis.

The criteria for the analysis of all lexicons are the same. Bell and Bird (2000) use a similar technique for comparing lexicons, however, they primarily focus on lexicons for different language areas. They focus only on lexicons for human use, not taking into account any lexicons for system use. Their aim is to provide a generic lexicon structure covering most existing bilingual lexicons. Their analysis of lexicon microstructures distinguishes three major classes in a lexicon entry, namely the *pronunciation*, *morpho-syntactic information* and *sense definitions*. This corresponds to similar categories used in feature structure representations such used by HPSG. The microstructure analysis here will impose a simple structure like this to the lexical categories, which are the

Surface categories, covering the appearance in orthography or pronunciation representation;

Morpho-syntactic categories, including inflectional classes and syntactic paradigms;

Semantic categories, where everything concerning meaning is subsumed.

This categorization of lexical data categories is not introduced for theoretic reasons but rather for easier use and access. For instance a translation in a bilingual lexicon will be subsumed under *semantic category*. This can be controversial as it is only a different form on the surface. However, for further discussion of the structures involved, the categorization is irrelevant but introduced for convenience in parallel with the cited sources.

Each individual lexicon used in the comparison is analyzed first for its microstructure, looking at all possible lexical data categories that can be identified. The microstructure description is followed by an analysis of the mesostructure, i.e. whether there are any cross-references, pointers to grammars, inference rules, or what information can be inferred from the front matter concerning a structure description and other metadata. The structural description then contains the macrostructure analysis, based on primary and secondary sorting criteria and headword selection.

As the lexicons are supposed to be used for other purposes and used in the development of a lexicon formalism, the final test will be how modern and portable the lexicon is in terms of the requirements defined by Storrer cited in Section 2.1.6 and by Bird and Simons cited in Section 2.1.5.

A quantitative comparison of these lexicons will not be included. This can be justified by the approaches of these lexicons and their coverage of different data categories. And even in the case of two lexicons appearing to be relatively similar, the problem remains of what a lexicon entry is. For example, the treatment of homographs and polysemy in the lexicon can make a difference in counting the number of headwords, i.e. if homographs and polysemous words are not distinguished, but each variant is granted their own lexicon entry, the number will be different from one where only polysemous words are distinguished, even if the same amount of data is covered. A quantitative classification will be rough and can only serve as an indicator of the size.

2.3.1 Overview of lexicons

The description of lexicons for human use is characterized by an explanation of layout conventions to indicate the lexicon structure. These lexicons nowadays exist also in electronic formats, but the visual representation remains the user's only way to access a lexicon entry. As lexicons for human use have different layout and use different media, the lexicons for human use are subclassified by the media used for their presentation.

Printed lexicons share the feature of having a fixed layout at the time of printing, also resulting in a fixed structure which is almost impossible to update. A theoretically possible way is covering original lexicon entries by adhesive labels with corrections, costly replacements of whole pages or adding pages or even volumes. Another possibility is reprinting from a modified source. However, the nature of print dictionaries is that they are fixed on paper and therefore stable.

This has certain consequences in terms of portability and the criteria defined by Storrer described in Section 2.1.6:

1. Due to the limited space on paper, a number of compression techniques are used, for example to include morphological information in the standard orthography or to avoid repeating a morphological root by using a placeholder or wildcard character. This data modeling does not always respect linguistic structures.
2. For large print lexicons it cannot be determined in which order the data was collected. This is not true for large encyclopedias and lexicons in different volumes appearing at different times.
3. References to the lexicographic resources are rarely included. It is not distinguished if this is due to a different tradition of defining ad hoc examples or if it is due to the limited space.
4. The user interface is fixed and not flexible
5. Other media to be included are restricted to printable pictures.
6. The process of backchanneling is only available by commenting to the editor for future revisions.
7. The data is fixed on paper. This means that the preservation is usually quite good, depending on quality of the paper as well as the user treating it.
8. The lexicon's copyright is held by a publishing house or an organization.
9. The citation tends to be simple by the standard procedure of citing books.
10. Accessing the lexicon in a library or by the owner of a lexicon is hardly a problem, but locating the lexicon if it is not locally available is almost impossible.
11. The underlying lexicon base is *closed source*, i.e. the underlying data and the format are not available.

Hence, print dictionaries are neither portable nor do they constitute a modern lexicon according to Storrer's criteria. The same is true for some electronic lexicons, especially if they are fixed on a storage medium such as a CD-ROM. In the case of a lexicon deviating from these general restrictions a comment will be issued in the lexicon analysis.

Electronically available lexicons can be distinguished by the intended application. Either they are intended for an application providing a user interface for humans, i.e. for human use, or they are intended for driving an application such as a research oriented application like a parser or tagger, or in a productive environment like a dialogue system, machine translation application or (speech) synthesis system.

Electronic lexicons share a few characteristics. They are usually built on top of a single source, which is present in a database. The status of this database at a given time can be saved and transformed to be used by an application running statically, although the real advantage of an electronic lexicon is the ease of maintenance if the database is not only stored centrally but is also remotely accessible. In this case adjustments can be performed on the central lexical database, being effective immediately after applying the change. A centralized structure requires a network connection and therefore an appropriate infrastructure, both on the user side and lexicon side of the system. A precompiled, static system focuses on the requirements on the user side, as the lexicon side is controlled in full by the lexicon maintainers. A flexible lexicon system on the other hand can be adjusted both on the lexicon base side of the system — for example for performance — as on the user interface side, such as size of fonts and colors.

The layout and structure of the individual entry of a human usable electronic lexicon can be compared to printed dictionaries as the structure is highlighted using typographical conventions. However, there are certain limitations that do not apply, e.g. the presentation space, which is limited by the number of pages of a lexicon, and the size of the characters and figures, which is not as restricted in the electronic medium, and references to other entries can be made explicit using technical interpretation functions such as hypertext links. Especially the lexicons available in networks are candidates for both portable lexicons in the sense of Bell and Bird (see Bell and Bird (2000)) and modern lexicons in the sense of Storrer (see Section 2.1.6).

Similar to these human usable lexicons are system based lexicons. Many computer based systems rely on some sort of lexicon. Computer based systems cover areas from text processing — e.g. text classification, text mining, spellcheckers — to speech systems, such as speech recognition systems, speech synthesis systems, dialogue systems. All these systems include a lexicon component; for example, a spellchecker compares a given word in a text with lexicalized words from a wordlist. If a word does not exist in the lexicon wordlist, another word from this wordlist is offered, according to constrained similarity relations. A text mining system filters for keywords, neglecting stop words, both part of a lexicon database; speech synthesis systems — at least

most modern ones — are based on a database containing prerecorded phonemic patterns as values for a phoneme based lexicon.

Lexicon systems for computer systems share some common features. To allow for machine processability, the structure of the lexicon needs to be fixed and well defined, the encoding of the structure needs to be clear and the content has to be interpretable, i.e. the content types have to match the system's required formats, either by using a restricted vocabulary or by constraining the structure. The content restriction is part of the document grammar of a lexicon rather than the presentation structure as the layout is irrelevant to a system as long as the structure of the content is well defined. Applications themselves can present the structure in various ways, i.e. a presentation structure can be imposed by an application.

The system based lexicons are consequently further candidates for modern lexicography as defined by Storrer and can be portable if defined appropriately.

Special cases of highly structured, usually electronically available, lexicons exist in the linguistic context. On the one hand, these lexicons are not oriented towards being published in book form. On the other hand, they are not directly included in applications. The size of lexicons from linguistic contexts is comparatively small, not having a wide coverage, but being rich in information. Some of these lexicons are only available in a printed model format, i.e. the complete lexicon is not available at all, but some examples are used to illustrate the lexicon structure in technical and scientific texts. The reasons for different linguistic theories are not under consideration here, but the structure of the information represented in the lexicons in linguistic theories are. The investigation of lexicon structures is based on the translation of existing models into some form of data representation, where no data structure is available in electronic readable format.

Modern linguistic theories show a different strategy from former Chomskyan models (see Chomsky (1995), p. 235), where rules are part of the grammar while the lexicon contains only idiosyncratic information. In modern theories such as the ones mentioned here, the difference between *grammar* and *lexicon* diminishes in some areas, especially as lexicons also allow generalizations and some, formally considered grammatical constructs, are approached from the lexicon side.

2.3.2 Monolingual Dictionary: Oxford Advanced Learners Dictionary

The Oxford Advanced Learners Dictionary of Current English (OALD, Crowther (1995)) is characterized as a monolingual dictionary for learners of English, restricting the number of words used in definitions.

Microstructure:

The OALD shows 18 different lexical data categories in the microstructure, shown in Table 2.1.

Table 2.1. Microstructure of the Oxford Advanced Learners Dictionary(Crowther (1995))

Surface categories	Morpho-syntactic categories	Semantic categories
headword	inflectional category	prose description
prototypical entries	word class	example
related idioms	derivation	synonyms
phrasal verbs		controlled vocabulary reference
pronunciation		usage information
orthographic variants		contextual information

Mesostructure:

The mesostructure has many features and contains different areas:

Cross-references: cross-references to other lexicon entries are included, using typographical highlighting

Pointers to grammars: a sketch grammar is included which is referred to by conventional abbreviations

Inference rules: no explicit inference rules are included, but some grammatical rules can be interpreted as such

Information in the front matter on the structure: a prose description of the microstructure, including compression conventions and typographical conventions. A prototypical lexicon entry is included. A brief description of the macrostructure is also included in prose.

Other metadata: Metadata on the whole lexicon, such as editor and edition is included. Additional editorial information — such as the time of insertion of a lexical item, language, author of a lexical entry — has to be inferred from the more general description on the book, with all the limitations this provides.

Macrostructure:

Primary sorting key: alphabetic sorting according to a headword.
Secondary sorting key: intuitive semantic substructure for homographs vs. polysemous words.
Further sorting key: there is no further sorting key
Headword selection: the authors of the lexicon take an orthographic prototype of a morphologic stem as the headword, i.e. a meaningful unit of language which can stand on its own. Words containing non letter characters such as numbers are not included. For words containing numbers only a spelled out variant can be found. Spelling alternatives are included inside of the lexicon entry, they do not receive individual entries.
Macrostructure class: The OALD is a semasiological dictionary.

Sample entry

> **unit** /ˈjuːnɪt/ *n* **1** a single thing, person or group that is complete in itself, although it can be part of sth larger: *a family unit* ○ *a course book with twenty units.* **2** a fixed amount or number used as a standard of measurement: *an unit of currency* ○ *The metre is a unit of length.* ○ *a bill for fifty units of electricity.* **3** . . .

2.3.3 Bilingual Dictionary

Another class of printed dictionaries is available for locating translation equivalents for words. An example of this type is the Collins German Dictionary (Terrell et al. (1995)), which contains translation equivalents, with the source and target languages being German and English.

A bilingual dictionary usually consists of two parts, one with the first language as the source and the second as the target language, and another which is organized vice versa. This bilingual dictionary is a semasiological dictionary.

Microstructure:

The microstructure is not very consistent. Definitions and sample uses are not included besides the translation and sample phrases. The presence of lexical categories varies a lot, although the headword and the translation equivalent seem to be mandatory.

The microstructure is listed in Table 2.2.

Table 2.2. Microstructure of the Collins German Dictionary(Terrell et al. (1995))

Surface categories	Morpho-syntactic categories	Semantic categories
headword	wordclass	sense enumeration
orthography	gender (relevant for German nouns and pronouns)	additional context in the source language
pronunciation		translation equivalent
syllables		phrases used with the headword in source language with a translation equivalent
stress		
pronunciation		

Mesostructure:

The mesostructure is again elaborated:

Cross-references: references to other lexical entries are given inside of the entries

Pointers to grammars: this lexicon contains a short sketch grammar, which is referred to from the lexicon entries

Inference rules: The same as for the Oxford Advanced Learner's Dictionary (see 2.3.2).

Information in the front matter on the structure: structural information is missing in the front matter

Other metadata: a prototype reference to the pronunciation is included in the front matter. The compression techniques used in the lexicon by abbreviation and typography are explained as well.

Macrostructure:

Primary sorting key: language of the headword
Secondary sorting key: alphabetically by headword
Further sorting key: semantic subclassification, the same as in Section 2.3.2
Headword selection: the same as in Section 2.3.2
Macrostructure class: The Collins German dictionary is a semasiological dictionary.

2.3.4 The case of a lexicon including different writing systems

Much more complex is lexicographic work in a lexicon for different writing systems, e.g. mapping a language written in one writing system onto another language using a different writing system. An example of this class of lexicons is a bilingual Japanese English lexicon (Halpern (1993)), which is a lexicon of Kanji characters, intended to assign 'meaning' to the characters of the Japanese Kanji writing system by giving an English gloss.

The lexicon consists of different parts, i.e. the part containing lexical entries and different indices for accessing them. To access lexical entries the indices have to be consulted. Halpern (1993) has four different indices:

1. An index by semantic field, identified by an English semantic field description, directly giving the Kanji characters and identification numbers for the lexical entry.
2. A Hepburn-System index, which is a standard transliteration system for representing Japanese in Latin characters, including an implicit grapheme-phoneme description, providing all Kanji characters that share this Hepburn-System realization.
3. An index of *radicals*, i.e. a standardized representation of Kanji character components, giving the number of strokes needed to write this radical and ordered by this number. This index starts from the notion of a *central part* of the character, carrying the core meaning. The definition of a central part is conventionalized.
4. An index by division of characters, for example characters can be divided by right and left part, top or bottom part, outer or inner part.

To find the Japanese equivalent for *word*, starting from the pronunciation and spelling of the word, the following steps have to be performed:

1. Find the word in the Hepburn-System index, i.e. the Latinized character encoding, where a variety of Japanese characters is given sharing the

same Hepburn-System representation. As the Japanese script system is syllable based, these are not complete words. Otherwise the same Hepburn representation would mean them to be homophones, informally they might be called *homophonic syllables*. By knowing the radical of the word the appropriate symbol can be selected, using the number of strokes needed to draw the symbol without the radical. This number then has the reference number of the lexicon entry.

2. The lexicon entry can be selected by the reference number.

Microstructure:

The microstructure for the Japanese-English lexicon is shown in Table 2.3.

Table 2.3. Microstructure of a Japanese English lexicon (Halpern (1993))

Surface categories	Morpho-syntactic categories	Semantic categories
character		notes on usage (rare)
identification number		frequency number
character variants source (e.g. origin of the variant Chinese writing system)		list of compounds and combinations with other symbols to form words
drawing model for the sign		synonyms
Hepburn-System		homophones
radical		
number of strokes needed for the character		
classification on character segmentation of the character		

Mesostructure:

The mesostructure is very simple:

Cross-references: only to synonymous Japanese characters and the pointers from the index to the entries

Pointers to grammars: –
Inference rules: –
Information in the front matter on the structure: reference to the indices
Other metadata: editorial information, see OALD.

Macrostructure:

Indexing by character division and by radicals is related to the writing system, resulting in a complex macrostructure, which is exemplified by the process of finding a word in the lexicon above.

Primary sorting key: the division of characters into right, left, top and bottom part
Secondary sorting key: radicals, i.e. central character features
Further sorting key: Hepburn system as tertiary and last by semantic fields
Headword selection: character based
Macrostructure class: The Japanese-English dictionary is a semasiological dictionary.

2.3.5 Orthography Dictionary: Duden

A specialized lexicon intended for standardizing orthography for the German is the Duden (for example Drosdowski et al. (1996)), which intends to be a normative reference for German orthography. Derivations are omitted and inflections are only included to define a reference form, for example, nominative singular for nouns and infinitive for verbs. As German uses a number of characters that are not part of the standard Latin alphabet, namely the umlauts `ä, ö, ü` and the `SS`, the treatment of these is defined as being sorted synonymously to the corresponding standard letter combinations for the umlauts (`ae, oe, ue`), and as a double `s` character for `SS`.

Microstructure:

The microstructure is very complex, as shown in Table 2.4, resulting from embedding as much information as possible into the orthography of a word. This embedding is accomplished by typographic highlighting, such as underlining and special symbols. 12 different lexical data categories, all indicated by typography, do not increase the readability, and the context dependent use of typographical conventions for different purposes does not allow a context free approach to the lexicon entry.

Table 2.4. Microstructure of Duden (Drosdowski et al. (1996))

Surface categories	Morpho-syntactic categories	Semantic categories
orthography	grammatical gender (if applicable)	notes on usage
syllable length (by underlining or printing a dot under long or short syllables)		explanations
syllable segmentation		etymology
trademark identification (icon)		
pronunciation (in special cases, using IPA symbols)		

Mesostructure:

The mesostructure is rather flat:

Cross-references: indicated by an icon with the cross-referenced word
Pointers to grammars: –
Inference rules: –
Information in the front matter on the structure: a list of abbreviations used in the lexicon
Other metadata: there are guidelines on orthography listed in the front section.

Macrostructure:

Primary sorting key: morphologically related (grouping related words in one entry)
Secondary sorting key: alphabetic sorting of headwords
Further sorting key: –
Headword selection: headwords, which are selected from a group of words with the same morphological root
Macrostructure class: –

Sample entry

Ein | heit; Tag der Deutschen - (3. Oktober);
Ein | hei | ten | sys | tem; . . .

2.3.6 Pronunciation Dictionary

Another specialized lexicon is a pronunciation dictionary, such as Wells (1990), which is intended to function as a normative reference to the pronunciation of English words. In contrast to the German orthography lexicon mentioned before the pronunciation dictionary is restricted to the pronunciation of words. The authors avoided other lexical categories if they do not relate to pronunciation.

Microstructure:

The simple microstructure for the pronunciation dictionary is illustrated in Table 2.5.

Table 2.5. Microstructure of a pronunciation dictionary (Wells (1990))

Surface categories	Morpho-syntactic categories	Semantic categories
orthography	proper nouns	keyword or keyphrase
British pronunciation variant		
American pronunciation variant		
general pronunciation variant		
old style pronunciation variant		

Mesostructure:

In contrast to the simple microstructure, the mesostructure has different options:

Cross-references: to orthographic variants in the lexicon

Pointers to grammars: –
Inference rules: grapheme-phoneme rules
Information in the front matter on the structure: explanation of phonetic symbols by prototypes; schematic pronunciation guide is given for vowels prototypes
Other metadata:

Macrostructure:

Primary sorting key: morphologically related (see Duden)
Secondary sorting key: alphabetical by headword
Further sorting key: –
Headword selection: prototype of morphologically related words
Macrostructure class: The pronunciation dictionary is a semasiological lexicon.

Sample entry

unit ˈjuːnɪt † -ət ∼s s
,unit "trust

2.3.7 Thesaurus

Roget's Thesaurus, first published in 1852 (see, for example, the more recent edition by Dutch (1962)), is a differently structured lexicon. Contrary to the other dictionaries mentioned here, it is structured by *concept*, called *ideas*, in the introduction rather than form.

Microstructure:

For the thesaurus two different microstructures have to be described, namely the main body of the thesaurus (Table 2.6) and the structure of the alphabetical index (Table 2.7).

Mesostructure:

The names of the data categories in the thesaurus are ordered hierarchically, which constitute the mesostructure.

Cross-references: by reference number to a concept

Table 2.6. Microstructure of the main body of the thesaurus (Dutch (1962))

Surface categories	Morpho-syntactic categories	Semantic categories
headword of superordinate category	wordclass	words that fall into this semantic field
identification number		

Table 2.7. Microstructure of the alphabetical index of the thesaurus (Dutch (1962))

Surface categories	Morpho-syntactic categories	Semantic categories
headword	wordclass	name of the superordinate category
		identification number

Pointers to grammars: –
Inference rules: –
Information in the front matter on the structure: hierarchy of concepts, distinguishing six divisions into 39 sections, each of the sections being (sub-) classified into subsections with individual *head*s, serving as the second lowest superordinate categories for the individual words.
Other metadata: –

Macrostructure:

Primary sorting key: numeric in the main part and alphabetic in the index of the concepts
Secondary sorting key: —
Further sorting key:
Headword selection: numeric identifiers of concepts
Macrostructure class: A thesaurus is the prototype of an onomasiological lexicon.

2.3.8 Monolingual electronic lexicon on the web

The use of modern electronic media has been part of a new development with publishing houses. An example of a monolingual lexicon originally published in print is the electronic version of the Oxford English Dictionary (see for example Murray et al. (1970)) which can be found on the World Wide Web (see OED online (2004) in the bibliography).

The online version is intended for faster and more efficient access than the print version. Access to lexical entries is granted via a search form, where a user can insert a word he/she wants to query for.

Microstructure:

The microstructure of the electronic OED (Table 2.8) is simple and similar to the print version of the lexicon. The lexicon is clearly biased at the semantics of the word, though the semantics is not as elaborate as in modern semantic theories. This bias can be seen by the extend of the semantic description and the sparse inclusion of other types of information.

Table 2.8. Microstructure of the Online Oxford English Dictionary (Murray et al. (1970))

Surface categories	Morpho-syntactic categories	Semantic categories
orthography	word class	semantic field
		definition or explanation
		source reference
		time identification for the source
		context of use
		bibliographical information for the source

Mesostructure:

Cross-references: explicit hypertext links to other entries
Pointers to grammars: –
Inference rules: –
Information in the front matter on the structure: –
Other metadata: hardly, only a brief description of the project.

Macrostructure:

Primary sorting key: edition of the print version

Secondary sorting key: semantic differences (representing homographs in different entries)
Further sorting key: alphabetically (to some extend as the search functionality does not require it)
Headword selection: orthographic prototypes
Macrostructure class: The online OED is a semasiological lexicon.

Sample entry

Figure 2.1 shows an example entry of the online version of the OED. The left frame shows an alphabetical wordlist, which has some double entries, the central frame shows a definition and example usages with the year of use. The top frame is part of the general navigation.

2.3.9 Bilingual electronic lexicon on the web

Some bilingual lexicons are available on the web, too. One of them is the *English <-> German* dictionary of Richter (2004), for English and German. Strictly speaking this is not quite true, as it serves as an interface to different lexicons, as well, allowing for monolingual search and specialized queries, for example, for proverbs and idioms. However, the lexicon maintained by Richter is the bilingual one.

Microstructure:

The microstructure is simple, based on translation pairs, i.e. a German term is linked to an English term in a table. The additional information is not included consistently, but as prose text, resulting in inconsistent usage restrictions and examples. Table 2.9 shows the lexical categories in this lexicon.

Mesostructure:

The mesostructure is built into the system. However, it is rather limited:

Cross-references: synonyms are realized as cross-references issuing a search for the synonymous term
Pointers to grammars: –
Inference rules: –
Information in the front matter on the structure: –
Other metadata: –

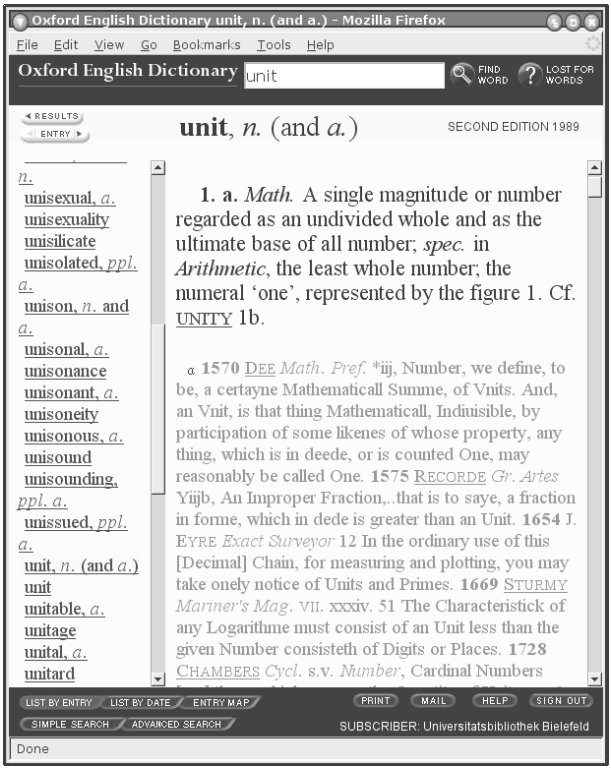

Fig. 2.1. Web version of the Oxford English Dictionary, query for *unit*

Macrostructure:

Primary sorting key: language (not required)
Secondary sorting key: –
Further sorting key: –
Headword selection: orthography (fuzzy search possible)
Macrostructure class: The *English* <-> *German* dictionary is a semasiological lexicon.

2.3.10 Terminology lexicon: Eurodicautom

Terminology dictionaries are derived from a different tradition than other lexicons, namely from technical writers and translators. In theory, the entries in

Table 2.9. Microstructure of the English <-> German dictionary (Richter (2004)); the semantic categories cover pragmatic categories as well

Surface categories	Morpho-syntactic categories	Semantic categories
headword	number (for plural words)	translation equivalent
pronunciation	gender for German words	usage restrictions
		regional variant
		semantic field
		usage examples
		synonyms

a termbank are created according to the semantics, a term being the verbal representation of a concept (see for example Wüster (1991)).

An example of such a terminology lexicon is the term database of the European Commission. This termbank, called Eurodicautom, is available on the web (see Eurodicautom (2004) in the bibliography). It contains terms in the official languages of the EU.

An interesting feature of Eurodicautom is a way of contacting the maintainers for including corrections directly. One reason for this is the special content of a termbase, i.e. the terms included are restricted to a knowledge domain, have a specialized usage that is more likely to change with new developments, but at the same time the terms are rather infrequent in general corpora. Hence locating terms requires other and larger corpora.

Microstructure:

The specialized field of terminology has been rather advanced in the use of electronic media and databases, trying to distinguish all data categories from each other, resulting in a transparent and consistent microstructure (Table 2.10).

Usually a termbank records editorial information on the authorship of a lexicon entry as well as a change-log, but Eurodicautom does not offer these in the user interface.

Termbases are assumed not to require information on word class and morpho-syntax, as most terms are nouns naming an item or idea in a subject field. Hence the need for grammatical information does not arise because the intended user group consists of language professionals. As the users of technical terms are supposed to be competent in using the language a term is used for, information on formal aspects are minimal and not treated with

Table 2.10. Microstructure of the Eurodicautom (Eurodicautom (2004))

Surface categories	Morpho-syntactic categories	Semantic categories
orthographic representation		language
		semantic field called subject field
		definition
		reference to a source of the definition
		notes

immense care, resulting in a sometimes lax treatment of them as recorded by Herbert Hasenbein (2002) in a discussion of weaknesses of online dictionaries.

Mesostructure:

The mesostructure is very simple:

Cross-references: by the ordering synonyms are presented. The structure of semantic fields provides for related terms
Pointers to grammars: –
Inference rules: –
Information in the front matter on the structure: –
Other metadata: –

Macrostructure:

Primary sorting key: language, not required for the access
Secondary sorting key: 90 semantic fields, not required for the access but allow a differentiation of subject fields.
Further sorting key: not transparent
Headword selection: concept name in one language
Macrostructure class: onomasiological lexicon (?)

Sample entry

Figure 2.2 shows a query result from Eurodicautom. It shows a query result for the English term *unit* with the target language German, showing the data

categories *definition*, *Reference* for the definition, the *Term* with certain notes by the terminologist and the German equivalent with a note, in this case it is empty, as the concept described by the definition does not have a German expression.

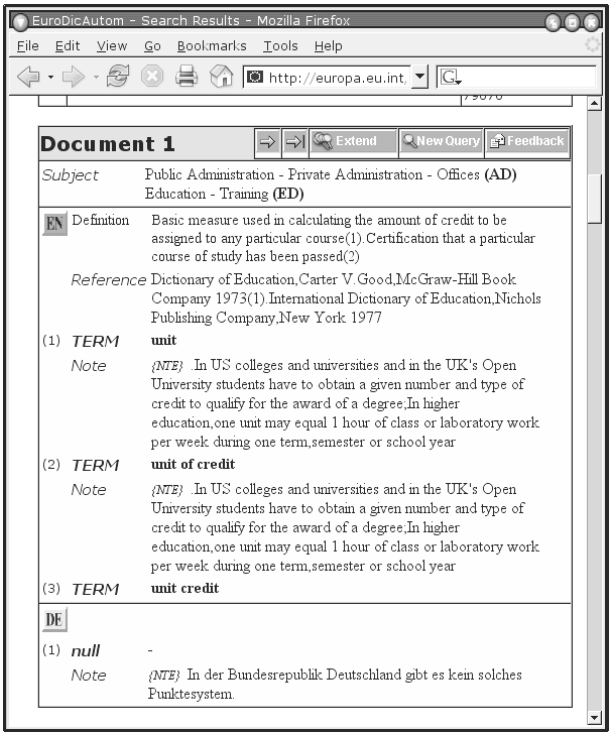

Fig. 2.2. Result for the query for *unit* in Eurodicautom, source language English, target language German, all data categories

2.3.11 Computer readable glossary: The Babylon GLS format

The Babylon *GLossary Source format* (GLS, see Babylon (undated)) is used for the commercial system Babylon-Builder (2004) to build lexicons called *glossaries*. These are used both from within an application, for example, for simple machine translation, or with an interface as a translation lexicon.

Hence it is not only a human readable lexicon but also a system targeted format.

Microstructure:

Babylon glossaries have a simple microstructure (Table 2.11), consisting of a two column structure of lexical items. The first column contains a headword in a list of alternatives, delimited by a vertical line, a pipe sign | . The second column contains a lexical property, such as a translation equivalent, definition, or lexical value. Lexicon entries are delimited by an empty line, i.e. two line-break characters; the columns are delimited by the first single line-break character after the first column.

Table 2.11. Microstructure of the Babylon *GLossary Source format* (Babylon (undated))

Surface categories	Morpho-syntactic categories	Semantic categories
headword		lexical property (?)

Mesostructure:

The mesostructure is restricted to cross-references as defined in the technical documentation:

Cross-references: optional cross-reference to other entries and to external sources
Pointers to grammars: –
Inference rules: –
Information in the front matter on the structure: does not apply; the structure is specified in a technical document
Other metadata: –

Macrostructure:

As a data format the Babylon GLS format does not specify a macrostructure. This is left to an actual application, though the headword seems to be intended to be a form of orthography.

Primary sorting key: –

Secondary sorting key: –
Further sorting key: –
Headword selection: free, orthography can be assumed
Macrostructure class: semasiological lexicon

Sample entry

The following are two entries in the specified format, created after the specification in Babylon (undated). Interestingly, the structure defined by the lexicon format is mixed with formatting information in HTML style.

```
anfallen|attackieren|angreifen
to attack someone<br>grammatical features see
<a href="bword://fallen">fallen</a><br>
picture illustration <img src="fallen.png">

fallen
to fall <br>
verb<br>
```

A table representation of the structure of the same entry is available in Table 2.12, again showing the remnants of formatting information.

Table 2.12. Babylon GLS lexicon entry in a table format

Headword	Synonym list	Properties
anfallen	attackieren angreifen	to attack grammatical features see ...
fallen		to fall verb

2.3.12 Wordlists as lexicons

The simplest lexicon, if not only a 'protolexicon', that can be defined are wordlists, serving as the basis for many applications, for example, for spell checkers. The microstructure is simple; only words are recorded. Wordlists are not connected to one application or use, as there are many applications and wordlists available that can be created based on a text. Although wordlists are

often alphabetically sorted or sorted by frequency, a macrostructure cannot be assumed. Mesostructures cannot be applied and the microstructure is trivial.

A sample lexicon of the wordlist type is the vocabulary list of the project *Deutscher Wortschatz* (Quasthoff (1997)), containing more than 2.4 million word forms. This lexicon is not lemma based and no other information is part of the lexicon, but it is a list of word forms extracted from large corpora as the basis for the identification of new words in texts. For more efficient processing, a second lexicon is part of this system, which is a stop wordlist, a list of more than 500 very frequent words that are not to be further processed.

Some extensions to wordlists exist, resulting in more and more complex structures. One wordlist extension is a frequency lexicon, where a second value is added, namely the frequency or relative frequency of a word in a corpus. A relative frequency is the number of a word divided by the total number of words.

Frequency lexicons serve as the basis for frequency studies, stylistic studies, but also for spell checkers and other applications based on a word form or frequency. A frequent word with similar features may be offered by a spell checker in the case of a questionable orthography.

Microstructure:

Table 2.13 shows the simple microstructure of frequency wordlists.

Table 2.13. Microstructure of a frequency wordlist

Surface categories	Morpho-syntactic categories	Semantic categories
orthography		frequency

Mesostructure:

The concept of mesostructure is not applicable to frequency lexicons.

Macrostructure:

Although sorting is not required for frequency lexicons, they usually are sorted.

Primary sorting key: either the numerical value of the frequency or the alphabetical order is used

Secondary sorting key: if the frequency is used as a primary sorting key then the alphabetical order can be the secondary sorting key
Further sorting key: –
Headword selection: either the frequency or the orthography is used as headword
Macrostructure class: semasiological lexicon

2.3.13 Computer readable semantic lexicons: the Multilingual ISLE Lexical Entry

Network based semantic lexicons such as *WordNet* rely on a concept hierarchy, usually a taxonomy, or multiple interwoven semantic hierarchies, sometimes called heterarchies. In semantic hierarchies a concept — in WordNet terminology a *synset* — is connected to other concepts, using lexical relations such as homonymy or meronymy relations. The term synset in WordNet is motivated by a composition of *synonym* and *set*. A concept can be identified by all terms in a set used to denote this concept. Terms identifying the same concept are synonyms. Hence the set of synonyms identifies the concept.

Frame based semantic lexicons constitute another representation originally derived from psycholinguistics (see for example Handke (1995), p. 100 for details). Based on a microcosm of prototypes, new concepts are defined in relation to the known concepts. The description of new concepts relies on the compositionality of existing concepts, i.e. one concept can be described by a number of other, closely related concepts . The result is again a network of interrelations between terms. A modern representation of such a lexicon is *FrameNet* (see Johnson et al. (2002)).

The *Multilingual ISLE Lexical Entry* (MILE, see Atkins et al. (2002) and Atkins et al. (ated)) is intended for a multilingual representation, based on the idea of allowing reusability of lexical resources for different purposes, sharing resources and adapting the model to user requirements. It was created within the context of the *International Standards for Language Engineering* (ISLE) project, which targeted best practice in the field of language and speech technology.

The MILE lexicon format intends to define both a standard for content and representation of a lexicon and refer to the inclusion of lexical semantic information. It explicitly allows for semantic, syntactic and morphological relations. Different representations are possible, of which one is based on XML and the Resource Description Framework (RDF, see Lassila and Swick (1999) and Ide et al. (2003)). As MILE allows for multilingual entries with complex structures, it allows for WordNet and FrameNet lexicons to be

transformed into its format. Lenci (2003) for example presents a mapping for WordNet lexical entries in MILE format.

The MILE lexicon entry is therefore taken as a prototype for this kind of lexicon.

Microstructure:

The complex microstructure for the MILE lexicon is given in Table 2.14.

Table 2.14. Microstructure of the MILE lexicon

Surface categories	Morpho-syntactic categories	Semantic categories
headword	wordclass	semantic specification
phonemic transcription		lexical relation to other lexical entries
		examples

The sample implementations in Ide et al. (2003) and Atkins et al. (ated) do not show explicit data categories for orthography and phonemic transcription.

Mesostructure:

A mesostructure is explicitly included, namely for cross-references.

Cross-references: to synonyms
Pointers to grammars: by wordclass (only implicit)
Inference rules: –
Information in the front matter on the structure: –
Other metadata: –

Macrostructure:

A macrostructure does not apply to a data structure, though a physical representation may be sorted by an application for efficient storage. In practice the MILE lexicon will be sorted alphabetically due to the workflow in the creation process.

Sample entry

The example entry is taken from Atkins et al. (ated).

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Sample ISLE lexical Entry for EAT (transitive),
SynU only Abbreviated syntax version using no pre-
defined objects 2002/10/23 Author: Nancy Ide -->
<rdf:RDF
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:mlc=
"http://www.cs.vassar.edu/~ide/rdf/isle-schema-v.6#"
xmlns=
"http://www.cs.vassar.edu/~ide/rdf/isle-schema-v.6#">
 <Entry rdf:ID="eat1">
<!-- The SynU for eat1 -->
  <hasSynu rdf:parseType="Resource">
   <SynU rdf:ID="eat1-SynU">
    <example>John ate the cake</example>
    <hasSyntacticFrame>
     <SyntacticFrame rdf:ID="eat1SynFrame">
      <hasSelf>
       <Self rdf:ID="eat1Self">
        <headedBy>
         <Phrase rdf:ID="Vauxhave">
          <hasSynFeature>
           <SynFeature>
            <hasSynFeatureName rdf:value="aux"/>
            <hasSynFeatureValue rdf:value="have"/>
           </SynFeature>
          </hasSynFeature>
         </Phrase>
        </headedBy>
       </Self>
      </hasSelf>
      <hasConstruction>
       <Construction rdf:ID="eat1Const">
        <slot>
         <SlotRealization rdf:ID="NPsubj">
          <hasFunction rdf:value="Subj"/>
          <filledBy rdf:value="NP"/>
         </SlotRealization>
        </slot>
```

```
        <slot>
         <SlotRealization rdf:ID="NPobj">
          <hasFunction rdf:value="Obj"/>
          <filledBy rdf:value="NP"/>
         </SlotRealization>
        </slot>
       </Construction>
      </hasConstruction>
      <hasFrequency rdf:value="8788"
         mlc:corpus="PAROLE"/>
     </SyntacticFrame>
    </hasSyntacticFrame>
   </SynU>
  </hasSynu>
 </Entry>
</rdf:RDF>
```

2.3.14 Machine Readable Terminology Interchange Format (MARTIF)

The *MAchine Readable Terminology Interchange Format* (MARTIF, also ISO 12200:1999 (1999)) is another lexicon data format. It is intended for the interchange of terminology data between different applications, such as terminology management systems and translation memory systems.

As an interchange format, MARTIF has been designed on the ground of standards, originally using SGML. In the meantime ports to XML have been suggested, with few consequences regarding the actual representation.

Microstructure:

The XML representation implies a tree structure on the representation level. Trippel (1999) discusses, for example, the transformation of a table based termbank into the tree structure of MARTIF. The MARTIF format contains many data categories available in print dictionaries, as well as editorial information which is usually found in the introductory sections of dictionaries, but here they are specified for every lexicon entry, see Table 2.15.

By using the term notes and description notes, the data categories can be extended. As a data format intended for termbases, the same restrictions apply as for Eurodicautom mentioned above.

Table 2.15. Microstructure of MARTIF

Surface categories	Morpho-syntactic categories	Semantic categories
concept name		subject field
		language
		reference to related concepts
		date of entry creation
		person creating the entry
		term notes
		description notes
		definition

Mesostructure:

The mesostructure defined by MARTIF is variable as the information is optional.

Cross-references: explicit references to related concepts are possible, as well as pointers to external information
Pointers to grammars: –
Inference rules: editorial information can be inferred using pointers to authors and resources used for definitions
Information in the front matter on the structure: a document grammar is referred to in the XML format
Other metadata: very detailed information on authors, editions, and cited resources are intended.

Macrostructure:

A macrostructure does not apply for a data format.

2.3.15 Integrated corpus and lexicon toolkit: the Field Linguist's Toolbox

Especially for linguists in a fieldwork situation, integrated tools allowing for synchronous use of corpus and lexicon are state of the art. These allow the creation of linguistic resources, inserting a textual corpus and, based on the wordlist, creating a lexicon structure.

A toolkit for this integrated corpus and lexicon processes is the *Shoebox program (Shoebox (2002)), the newer version published as the* Field Linguist's Toolbox*Shoebox* (Toolbox (2004)). This program allows the insertion of texts, either by import or by an editor, and the creation of a wordlist from this text. This wordlist is used for the basic lexicon creation; words already in the lexicon are automatically glossed with the lexicon entries.

Microstructure:

The microstructure of the lexicon contains a number of different lexical categories, taken from a list of predefined lexical data categories, but allowing for the free definition of further categories. The order of the lexical categories is free, they are marked up by an idiosyncratic abbreviation.

Mesostructure:

Generalizations over lexical categories in a mesostructure are not represented; the integrated lexicon builder, however, can be used for the definition of typographic highlighting, which can be used for an implicit mesostructure.

Macrostructure:

The macrostructure is freely definable by selecting the data category for sorting the lexical entries.

2.3.16 Verbmobil lexicon

The Bielefeld Verbmobil lexicon (Gibbon and Lüngen (2000)) is the German reference lexicon created for the Verbmobil project, dealing with speaker independent automatic translation from German into English and Japanese; the Verbmobil system is restricted to the domain of appointment scheduling and and travel booking. At the same time, the Verbmobil lexicon is a prototype for inheritance lexicons.

Inheritance lexicons cover the class of lexicons that allow inheriting lexical information on one lexicon entry from another lexicon entry. Ideally, lexical information is only defined once in the dictionary avoiding all redundancies. Kilgarriff (2001) describes *compact lexicons* which allow the inheritance of generalized lexical information, exemplified by valency of verbs and

semantic knowledge. Inheritance lexicons as discussed in this section are lexicons that are computer processable, i.e. based on the lexical information a program processes.

In inheritance lexicons the microstructure can be freely defined by the lexicographer, i.e. lexical data categories can be defined, according to the linguistic framework used by the researcher. The order of lexicon items does not need to be fixed. One formalism to encode inheritance lexicons is DATR (see Evans and Gazdar (1996), with its implementations, such as Gibbon and Strokin (1998)).

The macrostructure of an inheritance lexicon is usually arbitrary. The lexicon entries do not need to be sorted.

Inheritance lexicons have an elaborated mesostructure, referring to other entries and rules not only for reference, but for applications to interpret and use the information referred to.

The microstructure and the mesostructure, however, are fixed for a specific application such as the Bielefeld Verbmobil Lexicon.

Microstructure:

The microstructure of the Bielefeld Verbmobil lexicon depends on the wordclass, i.e. different wordclasses show a different static vector of data categories. The data categories are listed in Table 2.16.

Table 2.16. Microstructure of the Bielefeld Verbmobil lexicon

Surface categories	Morpho-syntactic categories	Semantic categories
lemma	wordclass	
syllabification	syncretism	
phonemic transcription	morphologic decomposition	
	derivational class	
	inflectional information	

Mesostructure:

Starting with a lemma based lexicon, a full form lexicon, i.e. a lexicon containing all possible inflected forms for a given wordclass, can be inferred, using the information from the syncretism in connection with the other bits

of information provided in an individual lexical entry. Part of the lexicon system are the rules that are used for this full form generation. Hence the mesostructure is rather elaborated.

Cross-references: –
Pointers to grammars: full inflection tables included with classifiers
Inference rules: full forms inferred by the rules based on the classification
Information in the front matter on the structure: inference rules, structure description
Other metadata: –

The representation of the Verbmobil lexicon is a Prolog feature vector, with an arity that depends on the wordclass.

Macrostructure:

The sorting is arbitrary, though different word classes are stored separately.

Primary sorting key: word class
Secondary sorting key: –
Further sorting key: –
Headword selection: lemma for the reduced version, inflected form for the full form lexicon
Macrostructure class: semasiological lexicon.

Sample entry

The following shows the entry *Einheit* (unit) in the German Verbmobil lexicon in the Prolog representation (the line-break was added for printing).

```
mor_noun_stem_lemma('Einheit', 'Ein+heit',
'?''aIn.+haIt','N', 'Nomen_Frau', fem, nonumlaut, _).
```

By omitting the Prolog specific syntax, the following ASCII based version of the same entry can be created:

```
'Einheit','Ein+heit','?''aIn.+haIt','N','Nomen_Frau',
fem,nonumlaut,_
```

The same entry with an explicit representation of the data categories in XML can be represented in the following way:

```
<entry type="noun">
 <lemma>Einheit</lemma>
 <stem_morph_segm_orth>Ein+heit
    </stem_morph_segm_orth>
 <stem_morph_segm_phon>?''aIn.+haIt
    </stem_morph_segm_phon>
 <syn_cat>N</syn_cat>
 <inf_class>Nomen_Frau</inf_class>
 <gram_gender>fem</gram_gender>
 <umlaut>nonumlaut</umlaut>
 <marker>_</marker>
</entry>
```

2.3.17 Speech synthesis lexicons

In modern speech synthesis, speech is usually taken from recordings (see Dutoit (1997), Chapter 1.4) , i.e. a speaker is recorded and this recording is processed in a way that parts of it can be reused by concatenation, resulting in new utterances.

The production of synthesis lexicons is based on a very detailed, time-related annotation of the recording, and a signal processing in which the signal is split up into appropriate units that are defined based on the annotation. One frequently used form is based on *diphones*, i.e. the period of transition from the center of one phoneme to the center of the next phoneme; the diphones are segmented automatically based on a phonetic annotation. The resulting diphones can be normalized for length and pitch and stored in a diphone base, which is a database optimized for speech synthesis allowing efficient access (see Dutoit (1997), Chapter 10).

Larger units than diphones increase the naturalness of the speech. Hence a tendency for speech synthesis goes in the direction of *unit selection*, i.e. taking the largest matching unit for the synthesis that can be found in the database. As the danger of this always is that a word to be synthesized is not in the database, it is not based on sentences or words alone, but also on diphones to allow a fall back to these in a case where no larger unit is available. The corpus and annotation for this kind of synthesis needs to be larger, to cover a sensible number of larger units. However, including larger units in a speech synthesis database has the effect of creating a larger database which takes longer to be accessed. An optimization of size and quality has to be taken into consideration in the building. A unit selection based Text-To-

Speech (TTS) system is available with the Festival system (see Black et al. (2004)).

Microstructure:

A speech synthesis lexicon, i.e. a sound unit database with corresponding signal chunks, is a simple lexicon system (Table 2.17), including more than one modality. In this case these modalities are speech in signal chunk form and text in phoneme representations, linked to each other.

Table 2.17. Microstructure of a speech synthesis lexicon

Surface categories	Morpho-syntactic categories	Semantic categories
unit transcription		signal chunk

The structure of the comparison table is maintained though it is rather doubtful, in which of the categories a signal fits.

Mesostructure:

The concept of mesostructure does not apply to the speech synthesis lexicon.

Macrostructure:

As a sorting strategy is not implied, a macrostructure cannot be postulated. However the phoneme can be seen as the headword of this type of lexicon.

2.3.18 Lexicon for part-of-speech tagging and automatic phonemic transcription

Programs for part-of-speech tagging and automatic phonemic transcription rely heavily on the underlying lexicons. An example of such a system is the Stuttgart Tree Tagger (see TreeTagger (1996) in the bibliography), which for German relies on a lexicon such as the *BOnn Machine readable Pronunciation dictionary* (BOMP, Portele et al. (1995))[1].

[1] The version that was available for investigation is the one originally defined for the Hadifix speech synthesis system, called *hadi-bomp* in the version 010802, created 2002.

Microstructure:

The sample lexicon database contains a wordlist database of more than 140.000 word forms, with a part of speech tag according to the Stuttgart Tübingen Tag Set (STTS, see Schiller et al. (1995)) and phonemic transcription with superimposed syllabification represented by the pipe symbol, the phonemic transcription using the SAMPA transcription system (see Gibbon et al. (2000b)). The data categories are listed in Table 2.18.

Table 2.18. Microstructure of the BOMP lexicon

Surface categories	Morpho-syntactic categories	Semantic categories
word form	part of speech tag	
phonemic transcription		
syllabification		

Mesostructure:

Metadata for the lexicon are available in an accompanying file, naming author, format and purpose of the lexicon as well as contact information, copyright, etc. This metadata are not coded explicitly according to a standard format but given in prose. Cross-References or other generalizations over the microstructure are not included.

Macrostructure:

The macrostructure of the lexicon should be arbitrary, but the text database representation is alphabetized by the word form.

Sample entry

```
Einheit NOM     '?aIn|haIt|
```

2.3.19 Feature structure lexicons

A very powerful lexical representation is available with *feature structures* or *typed feature structures* as described by Shieber (1986). Feature structure lexicons describe lexical entries using a microstructure with a superimposed hierarchy of the microstructure such as a tree structure. This hierarchy can be used for inheriting information. Feature structures are, for example, used in the *Head driven Phrase Structure Grammar* (HPSG, see Sag and Wasow (1999)).

HPSG lexicons are usually represented in a attribute-value-matrix form similar to the example shown for the generative lexicon in Figure 2.5, but there are formal representations in textual form. For example, with the emerging standard developed by the *International Organisation for Standardisation* (ISO) [2]. The structure is an embedded AV-format, which can be represented as a *Directed Acyclic Graph* (DAG, see for example Shieber (1986), section 2.4). These are trees with the possibility of reentrancy, i.e. some nodes can have more than one root node. These DAGs are interpreted from the root.

Microstructure:

The microstructure of an HPSG lexicon includes data categories from semantics and syntax, but also from a surface representation [3]. The root of the DAG is the headword of the lexicon entry, which is in general compared to the lemma concept. Though Shieber (1986) (Chapter 3, Section 3.1) states that there is not a principal reason for interpreting a Feature Structure bottom-up (that is in the example from right to left) or top-down (left-to-right, resp.), the headword is fixed by the root node. A lemma base approach, however, presupposes a fixed macrostructure, which is semasiological. For a lexicon formalization that is intended for the most abstract representation, this is a major design flaw.

For the application of feature structures in the context of syntactic theories this approach is suitable and allows the encoding of many features of a lexical

[2] With ISO CD 24610-1:2003 (2003) and ISO DIS 24610-1:2004 (2004) ISO TC 37 SC 4 has issued a number of related documents, but non of them is in a final phase yet. See http://www.tc37sc4.org for the related documents.

[3] Sag and Wasow (1999) and others using their work include a category *PHON* which is supposed to mean *phonemic representation* or something similar, but they fill this category with orthographic representations only. However, *surface representation* here is intended to bridge this schism allowing a modality specific representation, such as phonemic or phonetic transcription for speech and orthography for textualized language.

item. Depending on the application and linguistic theory, the data categories can vary.

Mesostructure:

The surface representation of an HPSG lexicon entry already displays some portions of the mesostructure: the data categories are classified and grouped according to their linguistic fields. Another possibility is the abbreviation of some features where the values are the same, which means that some features can be inherited. Additionally by the application of certain rules different grammatical forms can be inherited in the sense of inflectional morphology. Sag et al. (2003) (Appendix A.5) contains 17 such rules, applicable for English. These rules can be applied to specific word classes only.

Metadata descriptions and specific cross-references to other lexicon entries are usually not included, as HPSG structures are usually only applied to one lexicon item at a time.

Macrostructure:

In Sag et al. (2003) (Appendix A.6) a basic lexicon in the HPSG paradigm is given, interestingly this lexicon is sorted by wordclasses, i.e. by one property other than the headword. The reason for this is obviously as this lexicon is part of a syntactic theory where wordclasses are of special interest. A secondary sorting strategy is not apparent. However, this lexicon only consists of 34 words. These 34 words were easily ordered by hand in the lexicon. The sorting according to POS would otherwise not be the obvious one according to the microstructure, which presupposes a headword based sorting strategy.

Primary sorting key: word class
Secondary sorting key: –
Further sorting key: –
Headword selection: lemma
Macrostructure class: semasiological lexicon.

Sample Entry

Figure 2.3 shows a Feature Structure lexicon entry in the HPSG paradigm for the English word *dog*.

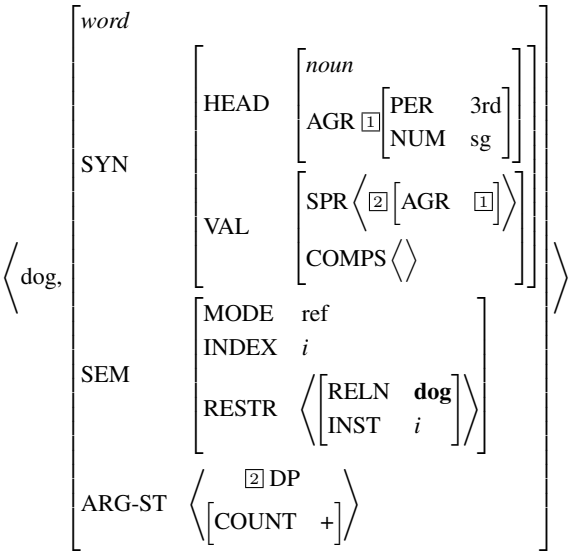

Fig. 2.3. HPSG lexicon entry in Feature Structure form for the English word *dog*, taken from Sag et al. (2003), p. 254

An XML representation according to ISO CD 24610-1:2003 (2003) is the following complex feature structure[4].

```
<fs>
  <f org="list" name="dog">
     <fs>
       <f name="orth">
          <str>dog</str>
       </f>
       <f name="word"/>
       <f name="syn">
         <fs>
             <f name="head">
                <fs>
                    <f name="noun"/>
                   <f name="agr">
                      <fs id="one">
```

[4] Some features and elements from this encoding are still experimental and subject to change in the process of standardization. However, the structure used here is the one that will be used for the final version of the standard.

```
                    <f name="per">
                      <str>3rd</str>
                    </f>
                    <f name="num">
                      <str>sg</str>
                    </f>
                  </fs>
               </f>
            </fs>
         </f>
          <f name="val">
            <fs>
               <f name="spr" org="list">
                  <fs>
                      <f name="null" fVal="two"/>
                     <f name="agr" fVal="one"/>
                  </fs>
               </f>
               <f name="comps" org="list"/>
            </fs>
          </f>
      </fs>
   </f>
   <f name="sem">
      <fs>
          <f name="mode">
            <str>ref</str>
          </f>
          <f name="index">
            <str>i</str>
          </f>
          <f name="restr">
            <fs>
                <f name="reln">
                   <str>dog</str>
                 </f>
                 <f name="inst">
                    <str>i</str>
                 </f>
            </fs>
          </f>
      </fs>
   </f>
```

```
        <f name="arg-st" org="list">
           <fs id="two">
               <f name="dp"/>
           </fs>
          <fs>
               <f name="count">
                 <plus/>
               </f>
           </fs>
        </f>
      </fs>
  </f>
</fs>
```

Very similar to this entry is the entry shown in Figure 2.4, which shows the plural form *dogs*. The only deviating parts of these lexicon entries are the word and the number, the earlier showing *sg* for singular, the latter *pl* for plural. These entries show the problem of redundancy in a lexicon. In fact Sag et al. (2003) give the inheritance rule for producing the plural entry from the singular entry. Only semantic, syntactic and argument structure features are inherited.

2.3.20 The Generative Lexicon

Related to feature structure representations of lexicon entries are many semantic lexicons used in the research community. In the *Generative Lexicon* (Pustejovsky (1995)) a method for semantic description is introduced which can serve as an example of the class of semantic lexicons.

The Generative Lexicon resolves polysemy and allows the introduction of intentionality, and creative use of words into the lexicon. The generative lexicon approach has been criticized frequently, such as by Frath (2003), who claims that the basic notions of the Generative Lexicon are problematic inasmuch as intentionality cannot be adequately represented in a lexicon and that the system involves circular rule application. The criticism of Kilgarriff (2001) is based on the problem that the Generative Lexicon cannot be used for productive and creative uses of words, but allows only for the description of a small number of words in a corpus that are not classifiable using existing lexicons. However, scrutinizing the Generative Lexicon is not the purpose of the present work.

$$\left\langle \text{dogs}, \left[\begin{array}{ll} \textit{word} & \\ \text{SYN} & \left[\begin{array}{ll} \text{HEAD} & \left[\begin{array}{ll} \textit{noun} & \\ \text{AGR } \boxed{1} & \left[\begin{array}{ll} \text{PER} & \text{3rd} \\ \text{NUM} & \text{pl} \end{array}\right] \end{array}\right] \\ \text{VAL} & \left[\begin{array}{l} \text{SPR} \left\langle \boxed{2} \left[\text{AGR } \boxed{1}\right] \right\rangle \\ \text{COMPS} \left\langle \right\rangle \end{array}\right] \end{array}\right] \\ \text{SEM} & \left[\begin{array}{ll} \text{MODE} & \text{ref} \\ \text{INDEX} & i \\ \text{RESTR} & \left\langle \left[\begin{array}{ll} \text{RELN} & \textbf{dog} \\ \text{INST} & i \end{array}\right] \right\rangle \end{array}\right] \\ \text{ARG-ST} & \left\langle \boxed{2}\,\text{DP}, \left[\text{COUNT } +\right] \right\rangle \end{array}\right] \right\rangle$$

Fig. 2.4. HPSG lexicon entry in Feature Structure form for the English word *dogs*, taken from Sag et al. (2003), p. 254

Microstructure:

Pustejovsky describes a word in four components, i.e.

1. The logical arguments with a typisation and required number, termed *argument structure*,
2. The typisation of an event, called *event structure*,
3. The relation to constituents, distinguishing factors to other concepts in the domain, function, and origin of the concept, the *qualia structure*, and
4. The *lexical inheritance structure*, which is the relation to a hierarchy, i.e. the relation to the mesostructure[5].

Mesostructure:

The feature structures can use inheritance rules, as indicated by the numeric placeholders in the following example. The hierarchy of the microstructure

[5] Interestingly this structure is not included in the lexicon samples shown in Pustejovsky (1995).

also is part of the mesostructure. Metadata descriptions, however, are not included.

Macrostructure:

The macrostructure is arbitrary again and not specified as the usual specification is based on one entry at a time.

Sample entry

Figure 2.5 shows a sample lexicon entry from the generative lexicon.

$$\left[\begin{array}{l}\textbf{build}\\ EVENTSTR = \left[\begin{array}{l}E_1 = e_1 : process\\ E_2 = e_2 : state\\ RESTR = <_\alpha\\ Head = e_1\end{array}\right]\\ ARGSTR = \left[\begin{array}{l}ARG1 = \boxed{1}\left[\begin{array}{l}\textbf{animate\_ind}\\ FORMAL = \textbf{physobj}\end{array}\right]\\ ARG2 = \boxed{2}\left[\begin{array}{l}\textbf{artifact}\\ CONST = \boxed{3}\\ FORMAL = \textbf{physobj}\end{array}\right]\\ \text{D-ARG1} = \boxed{3}\left[\begin{array}{l}\textbf{material}\\ FORMAL = \textbf{mass}\end{array}\right]\end{array}\right]\\ QUALIA = \left[\begin{array}{l}\textbf{create-lcp}\\ FORMAL = \textbf{exist}(e_2, \boxed{2})\\ AGENTIVE = \textbf{build\_act}(e_1, \boxed{1}, \boxed{3})\end{array}\right]\end{array}\right]$$

Fig. 2.5. Sample entry from the generative lexicon, taken from Pustejovsky (1995) (p. 82)

This lexicon entry shows a hierarchical organization of the lexicon microstructure, which consists of the categories E_1, E_2, $RESTR$, $HEAD$, $ARG1 : FORMAL$, ... These are classified in a hierarchy according to the event structure, argument structure and, qualia structure.

2.3.21 Summary of the lexicon analysis

Table 2.19 shows a detailed summary of the lexicon analysis using the following columns:

Lexicon: the lexicon under consideration
DatC: number of data categories identified in the microstructure
Refs: existence of cross-references
Gram: existence of reference to grammars
Inf: existence of inference rules
Meta: existence of metadata
Struct desc: existence of a structure description
Front: completeness of information in the front matter
Prim primary sort key
Sec secondary sort key
Headw: headword selection
Mod: ranking according to the seven requirements defined by Storrer: one point for each fulfilled requirement
Port: ranking according to the seven portability criteria raised by Bird and Simons: one point for each fulfilled requirement
Class: part of the semasiological or onomasiological class of lexicons.

Table 2.19: Lexicons compared (summary)

Lexicon	DatC	Refs	Gram	Inf	Meta	Struct desc	Front	Prim	Sec	Headw	Mod	Port	Class
OALD	15	typographically	sketch gr.	–	–	prose	yes	orthography	semantics	orthographic word	0	3[6]	sem
Collins German	> 12	explicit	short sketch gr.	–	–	prose	yes	language	alphabetically by headword	orthographic word	–	3[7]	sem
Japanese-English	12	synonymous characters + index	–	–	–	prose descr. of index	yes	division of characters	radicals	character	–	3[8]	sem
Duden	9	typographically	–	–	–	list of abbreviations + orthographic rules in prose	yes	morphology	alphabetically by headword	orthographic word	–	4[9]	sem
Longman Pronunciation	7	orthographic variants	–	prose grapheme-phoneme rules	–	prose	yes	morphology	alphabetically by headword	orthographic word	–	4[10]	sem
Thesaurus	14	to concept	–	implied in concept hierarchy	hierarchy of concepts	–	yes	numeric concept identifier	–	concept number	–	3[11]	onom

[6] Citation, preservation and rights are clear.
[7] Citation, preservation and rights are clear.
[8] Citation, preservation and rights are clear.
[9] Citation, preservation and rights are clear.
[10] Citation, preservation and rights are clear.
[11] Citation, preservation and rights are clear.

Table 2.19: Lexicons compared (summary)

Lexicon	DatC	Refs	Gram	Inf	Meta	Struct desc	Front	Prim	Sec	Headw	Mod	Port	Class
OED online	8	hyperlinks	–	–	–	–	see text	edition	semantic differences	ortho-graphic word	3[12]	2[13]	sem
dict.tu-chemnitz.de	10	hyperlink	–	–	–	–	–	language	–	ortho-graphic word	4[14]	1[15]	sem
Eurodicautom	> 5	implicit by semantic field	–	–	–	–	–	language	semantic field	concept name in source language	6[16]	5[17]	onom (?)
Babylon GLS	2	hyperlinks and explicit cross-references	–	–	–	technical documenta-tion	–	–	–	ortho-graphy	6[18]	2[19]	sem
Wordlist	1	–	–	–	–	–	–	–	–	word	-	-	sem (?)
MILE	vari.	explicit	–	–	–	–	–	–	–	identifier	6[20]	2[21]	sem (?)

[12] Data collection depends on the organization on the publishers side; source is transparent; the system is extensible.
[13] Defined access and rights by the publisher.
[14] Work flow; limited adjustment of the user interface (with or without phonemic transcription); extensible
[15] Access only; if access to the sources can be provided, rights, content and format can be assumed.
[16] Other media are not included.
[17] Citation: no standard reference; format unclear because of lacking access to sources; preservation assumed due to the reliability of the maintainers.
[18] Backchanneling does not apply to the structure.
[19] Reliable structure; open format.
[20] Backchanneling does not apply to the structure.
[21] Reliable structure; open format.

Table 2.19: Lexicons compared (summary)

Lexicon	DatC	Refs	Gram	Inf	Meta	Struct desc	Front	Prim	Sec	Headw	Mod	Port	Class
MARTIF	vari.	pointers to editorial data and resources	–	editorial information inherited	editorial information per entry	DTD	–	–	–	concept identifier	6[22]	2[23]	onom
Toolbox	user defined	can be implied by user defined datcat	–	–	–	–	–	–	–	word	7	7[24]	sem
Verbmobil	> 8[25]	–	–	inference rules for inflections	–	prose	technical documents	word class	–	lemma	5[26]	6[27]	sem
Speech Synthesis	2	–	–	–	–	–	–	–	–	diphone	4[28]	2[29]	sem
BOMP	3	–	–	–	–	prose	in an extra file	–	–	orthographic word	4[30]	6[31]	sem
HPSG	vari.	placeholders included	part of syntactic theory	implied	–	–	–	–	–	word (?)	(?)	(?)	sem

[22] Backchanneling does not apply to the structure.
[23] Reliable structure; open format.
[24] Portability relies on annotator consistency and reliability of the archive
[25] Depending on wordclasses subcategories are defined.
[26] Project inherent corpus as source; no other media; assessment discontinued.
[27] Discovery depends on the archive.
[28] Data collection according to work flow; data modeling according to (linguistic) structures; other media implied; transparent relation between source and description.
[29] Rights and structure are clear.
[30] Relation of source and description is unclear; no other media involved; backchannel does not apply.
[31] Discovery is problematic.

Table 2.19: Lexicons compared (summary)

Lexicon	DatC	Refs	Gram	Inf	Meta	Struct desc	Front	Prim	Sec	Headw	Mod	Port	Class
Generative Lexicon	> 4	–	(?)	implied	–	by model specification	–	–	–	word (?)	(?)	(?)	sem

2.4 Summary

This chapter discussed the structures of lexicons, and existing lexicons were analyzed. The lexicon microstructure, mesostructure and macrostructure were introduced and identified in existing lexicons. The lexicons were evaluated on the basis of portability and requirements for state of the art lexicography. No lexicon and no lexicon model were fully specified in terms of structure, metadata description, portability, and modern requirements. Inference was part of some lexicons, especially of automated applications. It was shown, that existing lexicons do not provide the structure to satisfy the requirements defined and that a number of problems especially related to ambiguity are not solved sufficiently. The open problems and structural requirements form the basis of the next chapter.

3

Introducing the Lexicon Graph: Combining different lexicons

A formalism allowing the combination of different lexicons in a generic representation model is the focus of this section. The previous sections describe the structures of lexicons and mention requirements for lexicons. These requirements are based on portability and use of modern infrastructure, but also cover a mapping of the lexicon structures. A description of lexicons is also included. From the implementation side the formal model is not approached yet, only traditional techniques such as representing a lexicon as a hierarchical tree (as in HPSG, see Section 2.3.19), and inference based lexicons (see Section 2.1.4) were described. Problematic issues such as *homonymy*, *polysemy*, and *synonymy* were mentioned but not discussed in detail as they are included in the literature rather frequently (see for example Sager (1990)).

A solution to the problematic cases, besides other pending issues in lexicography is presented with an implementation of a *Lexicon Graph* (LG) as described in this section. The model is applied to existing lexicons using samples of different lexicon entries to be represented.

3.1 Pending issues in lexicography

Problems of ambiguity in the lexicon have been addressed frequently in the literature. Cases of ambiguity are not the only problems not solved in the literature, they cover for example:

- Special interrelations between lexical items, such as homonyms, polysemous words, synonyms. These are essentially special cases of the duplication of lexical information which basically can be reduced to the case of duplication of lexical knowledge, which is described in Section 3.1.5.

- Inclusion of special data items such as audio and video information, illustrations, special characters, which is a special case of non-textual data in a lexicon, treated in Section 3.1.7.
- Sorting of lexical items, which is not only a question of sorting according to which sequence in non ASCII codings, such as where to position the German umlaut characters, but according to which data category to sort, such as semantic (termbases), morphologic (morpheme lexicon), graphemic (Japanese lexicons: number of strokes). This issue is related to the headword selection, see Section 3.1.2.

The rather special problems that are treated by lexicographers working on dictionaries are related to more general problems. These general problems occur especially when lexicons are processed that were created for different purposes originally. Examples for lexicons created for different purposes are spellchecker lexicons to be used for speech synthesis, or full form lexicons to be used for encyclopedia, etc. The more generic problems are:

- Combining existing lexical resources
- Free headword selection to allow
  - Different sorting strategies
  - Different access structures
  - References
- Use of different, purpose dependent presentation structures,
- Use of inference
- Prevent multiplication of information items
- Allow automated consistency check for references and structures
- Complete representation of information contained in the source corpus
- Inclusion of non-textual data.

These questions require a detailed discussion. Hence the open issues are discussed in the following sections.

3.1.1 Combining lexical resources

The combination of lexical resources is intended to provide a possibility of reusing existing ones, ensuring portability and use of the available structures. The intention for the combination of lexicons is therefore to combine two or more lexicon databases for the low cost creation of a new and larger lexicon with a broader coverage and an extended usability. For the creation of the new lexicon the following major problems need to be solved:

- Different structures: depending on the purpose of the lexicon the microstructure, macrostructure and mesostructure (see Section 2.2) differ from each other. This means that different data categories and different structures of the data categories, for example, in trees, graphs or table structures, are needed. One example is the structure of an orthographic lexicon which differs significantly from a speech synthesis lexicon in terms of the data categories.
- Different coverage: coverage in this context does not only include the number of lexical items but also their area, such as the domain and the coverage within a domain, for example, a general lexicon might have some technical terms of an area that is almost completely covered by a second one which happens to be a technical dictionary.
- Overlapping lexical information: any two lexicons to be combined can have a share of common lexical information, the combination therefore can result in the multiplication of information with all problems of maintenance which are caused by it.

In terminology science there is an established tradition of interchanging terminological databases, which resulted even in international standards such as MARTIF (ISO 12200:1999 (1999)). For other lexical resources standardization is intended but not advanced yet.

The international standard for the terminology interchange does not answer any of the problems sufficiently. This has a simple reason: it assumes that the structure of the terminological lexicon is similar if not congruent and therefore can be mapped onto each other in a one to one relation. This starts from the following assumptions:

Every lexical entry represents one concept only (see for example Sager (1990), Wüster (1991) and Arntz and Picht (1989)). This concept is represented usually with a name that is similar to the lemma concept in traditional lexicography but which is only used as an identifier.

Some structures in every term entry are similar, for example, every entry contains (at least one) orthography, semantic and structural information in different complexity. Heinrich (2004) describes the process of double entry recognition based on partial similarity.

The combined termbase has a similar structure, for example, there is not a typological shift from a concept to a lemma based lexicon involved in the interchange.

The domain is restricted and defined resulting in a common feature of all items.

Interchange is restricted to applications, i.e. it is intended for the reuse of terminological data by different programs. This means especially that the development and maintenance of the lexicon is not handled on distributed systems.

Based on these assumption a core structure of terminological databases can be assumed. The combination of distinct databases with empty intersections can then be reduced to the addition of subtrees in a large target tree structure. Nevertheless the issue of duplicate entries, i.e. entries that were present in more than one source termbase is not addressed at all and is not only a problem of termbases but of databases in general.

3.1.2 Headword selection

The problem of headword selection is not the most obvious one for the traditional human user trying to find some information on a lexical item (see Section 2.2.3). This is because the user assumes the item in question to be the headword, which is due to his or her training in the use of lexicons. A different example, which is still based on a standard graphemic representation was given with the Japanese lexicon in Section 2.3.4. Nevertheless the headword selection has a fundamental influence on the final product.

In an article about computational lexicons, Byrd writes:

> "It is clear, however, that random access to dictionary entries by headword is not sufficient by itself to meet the needs of human users of dictionaries. It is well known that people require access to word information by sound, meaning, and other possibly *unpredictable aspects of word's representation*." (Byrd (1991), p. 117; emphasis added)

Classifying a lexical category as the *headword* can hardly be accomplished if the users headword selection is *unpredictable*. One specific result of this undetermined access structure is that every lexical category potentially can serve as a headword. For the use of any lexical category as a headword a way needs to be provided. For example, a dictionary of *parts of speech* (POS) may have not more than a few entries (i.e. *nouns*, *verbs*, *adjectives*, *pronouns*, etc.), each having a huge number of lexical information, such as the words in the lexicon that belong to these word classes. A *lemma* based lexicon on the other hand gets the information on wordclasses as values, resulting in multiple occurrences of the parts of speech. Both lexicons have their use and may contain the same information, but not only are the presentation structures

different from each other but the lexicon microstructure, and macrostructure differ significantly.

Selecting all lexical items which share certain characteristics, for example being of the same type is another issue of headword selection. For example, a phonological transcription in an ASCII representation might have the same appearance as the orthography but both are of different type, one might be used as an orthography in a spell checker, while the other is used in speech synthesis.

If a lexicon database contains information on the phonology of a word, the orthography, a definition, etc., this database should be usable for a lexicon with each of these categories as headwords, even if this was not the intention of the original author of the lexicon. For this purpose a way needs to be found to represent the information structure of the lexicon explicitly.

3.1.3 Single source lexicon publishing

The lexicon source is based on the lexicon structure, which can be used for renderings for different purposes. One example was mentioned before with the POS lexicon and the lemma based lexicon containing wordclass information, but other examples are possible as well: A speech system based lexicon such as the Bielefeld Verbmobil lexicon contains information for the included words that are also contained in a lemma based orthography lexicon, pronunciation lexicon or morphologic lexicon or any combination of them. The presentation structure is usually only seen as another rendering of one particular machine readable format. The machine readable format as such is used as the structure for an application such as the Verbmobil lexicon being the structure used by some tools of the Verbmobil project. The structure can be used for generating a visual output for the other types, either for print or for electronic rendering.

The use of the same lexicon base for different editions is practiced in print dictionaries where one publisher has different series for different target groups. This results in the need for a generic information structure to be defined. One attempt on defining a generic structure for a lexicon was proposed by the TEI (Sperberg-McQueen and Burnard (2001c)), defining a formalism for the encoding of information from print dictionaries. This TEI encoding for print dictionaries is not suitable for other types of lexicons, because it does cover all possible data categories for lexicons and does not allow the class of data categories to be open for different types of lexicons.

3.1.4 Inference in a lexicon: using knowledge

In machine applications lexical knowledge is used to infer additional information for lexical entries. Human users of lexicons also infer lexical knowledge, this procedure is sometimes described as *experience* or even *language competence*, which is used, for example, in highly inflected languages to synthesize the correct inflected form in a given context even if the word itself might be unknown. Another use is in the derivation of new words from existing ones. The knowledge that is learned by the human user needs to be coded for machine readability if a computer program is supposed to do the same inference.

A general lexicon model needs to provide a way to include lexical knowledge as well, and an appropriate representation needs to be found.

3.1.5 Duplication of lexical information as a problem of maintenance

The duplication of lexical information was addressed in Section 3.1.1 in the context of the merging of lexical resources. The problem of duplicates is not only a problem there but also in more general: if a lexical item is included within a lexicon database more than once at least two problems occur:

- When querying the database, the duplicates could be found and need to be handled. Traditionally, they have unique identifiers, which results in the coexistence within the database, but when queried the processing engine needs to decide which variant, if any, is to be returned. The default could be to present every distinct entry but this is not the desired effect especially if an application needs to decide which one to take. Ambiguity in the lexicon exists also as a problem for a human user searching for a term in a dictionary and finding more than one such as in translation dictionaries, finding two *translation equivalent*. Problems that apply here are explained by Heinrich (2004).
- When updating the database, for example in order to correct an orthographic error, the change is either applied to all duplicates, in which case the system has to find and recognize them as duplicates and contain routines of handling them, otherwise only one is altered resulting in inconsistencies with unpredictable side effects. These inconsistencies could be that two words, originally duplicates, are differentiated after updating due to different orthography but crossreferences could point to either the altered or non-altered variant.

An even larger problem are partial duplicates, i.e. the case in which one lexical item has different properties, for example, due to a merger of two different lexical databases, where the same lexical item is present twice with

different, possibly contradicting values. In traditional lexicography ambiguity applies, for example, to the problem of homonyms, homophones, homographs, synonyms and polysemous words. The lexicon graph however allows ambiguity as this is part of the real world.

Whenever two lexical entries share a number of features they can be grouped together and might be unified, i.e. combined into one entry, if it is in fact the same feature. Unfortunately, the examples from traditional lexicography show that the entries might be distinct from each other in other respects. Some data categories could have the same name but different content, such as the citation or examples, hence they cannot easily be unified by appending information from data categories available in one of the entries to the non redundant information of the other. The same is ture for data categories not available in one but in the other, which could be added. Nevertheless, the problem of maintenance remains with the features they share. One possible example could be a case where different systems for naming parts of speech exist. A solution would be to rename the parts of speech, which could result in a huge number of changes to preserve consistency. In principle this is the case for all redundancies in the lexicon. A lexicon model has to answer the problem of redundancy to maintain consistency.

3.1.6 Consistency of a lexicon and preserving corpus information

The maintenance issue points directly to the problem of maintaining consistency within a lexicon. From the information structure the lexicon has to be consistent for

- Applying appropriate tools for the transformation into required data formats for applications
- Creating consistent layouts for human users
- Return reliable data — both in structure and content — to tools and users working on the data.

Consistency in this context is related to the structure of the lexicon, while reliability refers to the content and homogeneity, i.e. that a user can assume the correctness of the content of the lexicon. Therefore it is required that the lexicon is *consistent* and *reliable* in terms of:

- The data type of the lexical items.
- The information structure.

By the definition of a fixed set of *data types* the first can be automatically addressed, by a document grammar the latter can be assessed. Data types in

this context refers to a grammar of the content, e.g. that the content has to be composed of a number-dash combination to represent a date, or a sequence of letters to be a name.

The preservation of lexical information of a corpus in a lexicon seems to be a reasonable requirement for a lexicon format. The transformation of corpora into different data formats and interchanging data for various programs requires the processing to be lossless (see also Sections 4 and 5.1), i.e. to preserve all bits of information contained in the source format. Although the goal for lossless representation seems to be obvious, it is not easy to perform. One reason is the use of different tools. The use of a variety of applications can lead to problems as described in Section 5.6.2 for corpora, i.e. differences on a level which is hard to perceive but which is different for automatic applications. The question remains on how to represent the information from the corpora in the lexicon and how to make sure that all information is covered. This is addressed later in Section 8.2.1. The basic lexicon representation nevertheless should contain all available information which can be filtered for the different uses, see also Section 3.1.3.

3.1.7 Including non-textual and other special data

The use of figures and photos in lexicons of the encyclopedia type is rather frequent in printed works. With the introduction of CD-ROM based lexicons (e.g. the OED, Murray et al. (1970), or the Microsoft encyclopedia, see Encarta (2004) in the bibliography) and other hypertextual lexicons the inclusion of multimedia events such as video sequences and animations has become equally as frequent. These sequences with an annotation can serve as a corpus, which itself can be the base of a simple lexicon, a concordance, which can be implemented using the temporal annotations. This will be described in detail in Chapter 7.

For a generic formalism a way of representing non-textual and other special data needs to be found.

3.2 A strategy for the definition of a generic lexicon

The strategy for the representation of a lexicon that fulfills the conditions of the previous section is based on the distinction of the different lexicon structures.

For the microstructure the modeling strategy is based on a dual concept, describing general relations between sets:

Sets of lexical items: the traditional data categories from a table-structured lexicon data format are transfered into sets of lexical items. Each set is classified and typed according to its purpose.

The set of relations between lexical items: by splitting the data categories into individual sets of lexical items the relation between the lexical items implicitly included in the table representation is made explicit. In the case where the columns are the data categories each column becomes one set of lexical items. The rows of this lexicon then contain one lexicon entry, i.e. the relation of the specific lexical items in this row. To maintain this information and to include it explicitly the relation between the lexical items is defined with reference to the set or even the different sets of lexical items.

The mesostructure is described by a knowledge component, which contains knowledge about the sets of data and about the relations. Lexical knowledge is given independently, for example, by an ontology or inference hierarchy.

Lastly, a mechanism for sorting and accessing needs to be identified, which is the macrostructure of a lexicon. It may seem odd again for a computational application to define a sorting strategy, as all sets of the microstructure can be used for sorting. However, the macrostructure definition relies on the selection of a headword, i.e. the restriction of the lexicon set to one subset used as headwords.

Figure 3.1 illustrates the relations between the sets of lexical items, relations and knowledge. It shows the three sets of `lexicon items`, `knowledge` and `lexical relations`, which are each interconnected in both directions. These connections are the references to the items. For example it is possible to define knowledge items for relations or for lexicon items. Lexicon items on the other hand can be related to other items, for which a relation is needed, or there can be knowledge associated with a lexical item. The relations have lexicon items as their components and there can be some knowledge about a type of lexical relation.

The lexical items need to be assigned a type to create, for example, a lemma or an orthography based lexicon. This typing is required for maintenance and updating as discussed below. The set of lexical items therefore need to contain typed items. Typisation can be accomplished using different strategies, first of all including a type hierarchy, second by attributing the type to the individual items. Hierarchical typing and type attributing are equivalent as there is a one-to-one relation between them. Grouping by the type allows hierarchy creation, while the inclusion of the hierarchy identifier as attribute

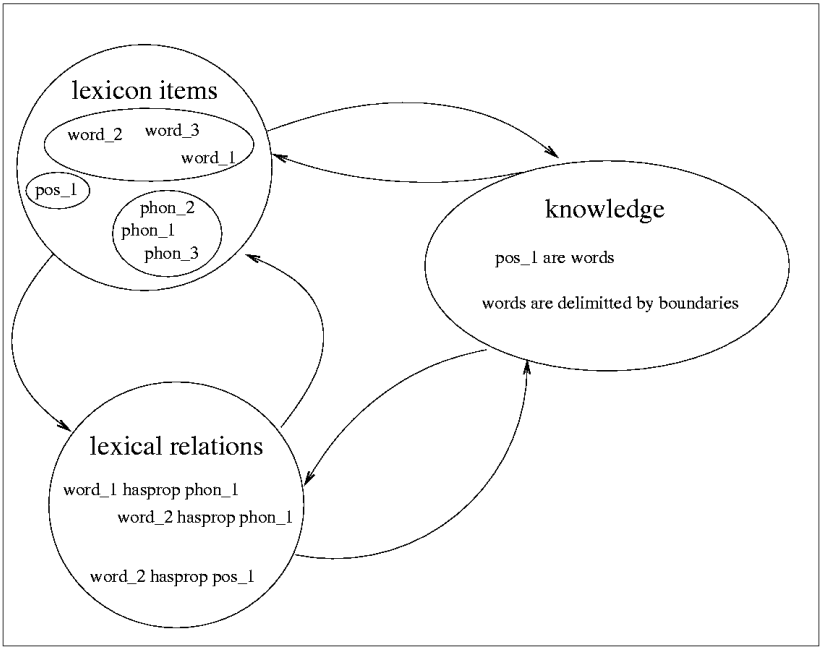

Fig. 3.1. Representation of the sets in a formal Lexicon Graph (LG)

creates other structures such as the vector of a microstructure. For the design of a lexicon this might be relevant, as a flat structure appears to be more generic in the conceptual design, because the type information is included only when required by an application.

In databases data is differentiated according to properties such as memory space allocated, allowed content, etc. The typisation of the content of the lexical items does not specify the data type in the database sense and the syntactically acceptable content of the lexical items. As lexical items can be both, textual or non-textual, the content needs to allow for both. For applications this means that lexical items have to be included in a way that they can be differentiated. For consistency the differentiation of lexical items requires a formal definition of the implied data structure. The formal lexicon model consists of:

- A formal lexicon description, which is the lexicon grammar. This is also the grammar for the lexicon information structure. It contains the following modules:

  - A structural description of the lexicon with the lexical items, relations and knowledge components, which is the *lexicon structure grammar*. In an XML context this can be modeled with a schema language such as DTD.
  - A description of the lexicon item data types, which is the *lexicon content grammar*. This definition of a type system is complex and cannot be solved as easily, because a type hierarchy could be involved. In an XML context, a grammar is therefore required which allows for more specific data types, for example, by using XSchema (Thompson et al. 2001 (2004b)).
  - A definition of the type system for the lexical items, which is either a closed vocabulary or at least a data type system for lexical type descriptions.
- A lexicon instance which conforms to the formal description, containing
  - Lexicon items, for example, a wordlist
  - Optional relations between lexicon items
  - Optional lexical knowledge
- An interpreting functionality, e.g. a description of a computer program allowing the use of such a lexicon.

The implementation will be based on the lexicon structure. The coverage by a lexicon content grammar would go beyond the focus of the research conducted here.

3.2.1 Microstructure in the Lexicon Graph view

The definition of different structures results in different sets of items, first of all the set of lexical items, which can be subdivided into subsets, for example, according to the type, lexical relations and knowledge. This section discusses their representation.

3.2.1.1 Representation of lexical items

The mapping of lexical items from a table structured representation model into a set structure is easy and straight forward as illustrated in Figure 3.2. The bar at the top is the list of lexical data categories, i.e. the column headings of a lexicon in table form. Each column in a lexicon table where the columns represent the data categories is mapped on one set of lexicon items. These sets are independent of each other, i.e. the relation that was implied by the table rows is lost in this representation so far. Each set of lexical items contains a

class of items, i.e. the set is described by distinctive features, for example, one set contains items in the *orthography* of a given *language*. The classification and distinct features of these sets are instances of metadata for lexicon microstructures as discussed in Section 6.3.3. The classification enables clustering of different lexicon sets, for example, for evaluation purposes. Specific examples of the application of this model to concrete lexicons are given in Section 3.4, when the model is applied.

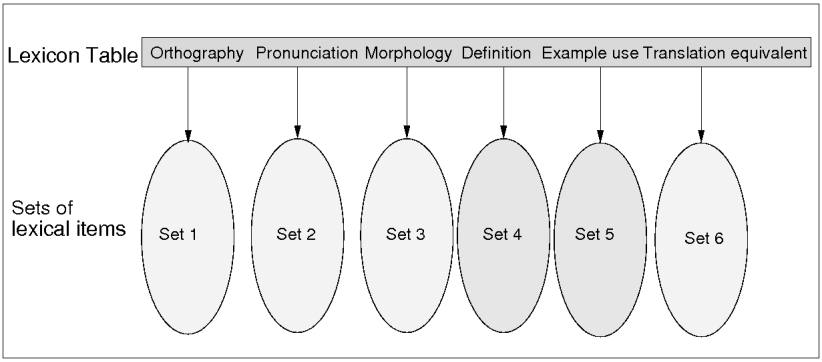

Fig. 3.2. Mapping of table structured lexicons on lexical item sets

In the implementation, a lexical item can be fully defined by a value, an identifier and a type, for example, a lexical value, such as a word in an *orthography*, a unique identifier, and the type, which is *orthography* in this case. The triple structure of this lexical item can then be referred to from lexical relations.

All lexical items here have the same status, which results in the same generic way for querying the lexicon. It is possible to create any subset of this lexicon with the same algorithms and methods, i.e. if the access is defined for one type of information, the same method can be used for other types by replacing the types.

As the lexical items are organized in a set, ordering is irrelevant and duplicates cannot exist. Lexical items that have the same type and the same value are identical. For the merging of a number of lexical databases, appropriate tools need to be provided, such as a method for comparing the values and types, i.e. if an item is supposed to be added to the lexicon, the first lookup checks if the item is with the content and type is already in the set. If not,

an identifier is created and the item is inserted. Otherwise the insertion is restricted to the definition of a relation.

3.2.1.2 Representation of lexical relations

By representing the lexicon data categories in separate sets the relation between them is not implied but needs to be represented explicitly. Figure 3.3 illustrates these relations. It shows three sets of lexical items, each with some elements. Some elements of the first set are related to elements of the second set, some of the third set, maybe also to both. As these relations differ in their quality and meaning, it is not sufficient to claim to items to be related, but this relation needs to be explained, i.e. typed.

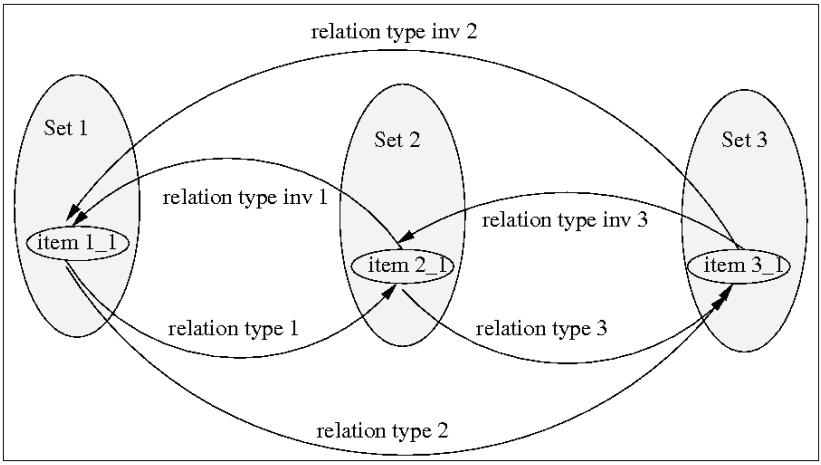

Fig. 3.3. Relation between lexical items of different lexical sets

Every relation of lexical items from a lexical entry is combined and represented by

- A source lexical item, which could serve as a lexical headword in a lexical relation
- A target lexical item, which is the assigned lexical value
- A type of lexical relation to classify the relation between the lexical items; this is typically a hierarchical relation of some kind
- An identifier, to directly address a relation.

The result is a directed graph, which allows every lexical item to serve as a source and as a target item; even self reference is possible. The self reference may result in an additional challenge in the processing, but this is irrelevant for the fundamental model. Assigning identifiers to a relation may not seem to be important, as the recorded items in a lexicon are the items and not the relations. In spite of that, an important application of identifiers for relations is in debugging where relations can be addressed directly. For practical reasons there is no reason not to include a mechanism for the direct identification of relations, as this does not increase the physical size of the database significantly.

The typisation of lexical relations needs to be discussed in further detail, hence it is included in the next section.

3.2.1.3 Typing lexical relations

The type of lexical relation is most crucial in the description of the lexicon microstructure as this is the assignment of data categories and their relation to each other. As different lexicons have different microstructures a simple way of creating new relation types needs to be part of the implementation. The most generic one, of course, is to allow for all lexical relations that are coded in a specified format to be allowed values. An example of such a format is a character string. This approach seems to be straight forward and simple, though it has one major disadvantage: lexical relations might have homonyms, i.e. lexical relations originating in different lexicon bases can have the same name but different coverage.

The problem of naming lexical relations seems to be unproblematic only at a first glance. The reason for this is that names can be easily interpreted, such as the relation named *IS-A*, which (usually) describes a transitive hierarchical hyponymy relation, or *PART-OF*, which (usually) describes a meronymic relation. The usage of both can vary, for example for some applications all hyponymy relations are *IS-A* relations, others differentiate between transitive and intransitive relations.

As the number of relation types is fixed and small for each lexicon (the number of columns in a table representation) the only solution for this problem that allows sufficient extensibility to new relation types is to include a manual process of relation type mapping when combining different lexicons or interpreting them. The only necessity in these cases is the uniqueness of relation type names in every lexicon, different depth of hierarchies in the data categories can be covered by this approach as well. The hierarchies of relations and the interpretation of the relation types is part of the knowledge

component in order to enable querying for further information. This information can then be richer than a simple flat table structure.

3.2.1.4 Resulting structure of the microstructure representation

The resulting representation of the lexicon model as described in the previous sections is a graph structure, with the nodes being the lexical items and with the relations being edges between the nodes. In a multi dimensional representation every type of relation constitutes a connection into a different dimension, which is rendered in a 2 dimensional space by labeled edges.

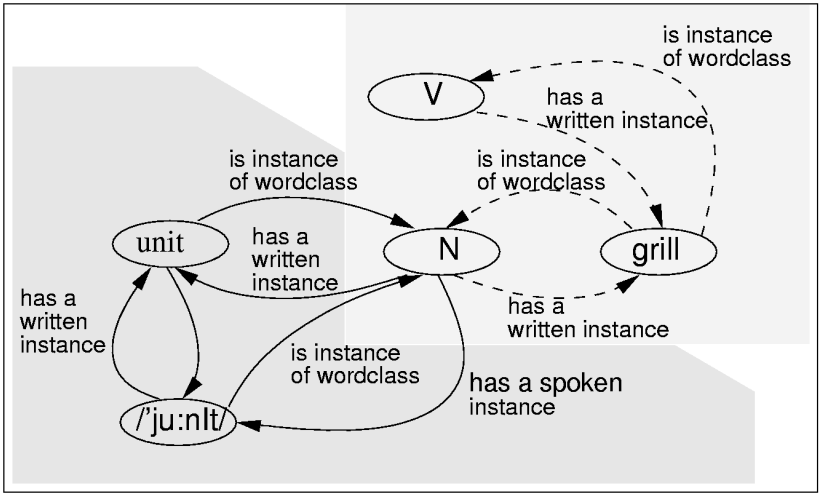

Fig. 3.4. Sample Lexicon Graph with 5 nodes; the dashed lined relations are from one lexicon, the solid ones from another

Figure 3.4 shows a 5 node graph with labeled edges. The graph is based on two lexicon entries, one for the word *grill* which is ambiguous in terms of wordclass, and the word *unit* with its pronunciation and wordclass. This example shows that ambiguity is not the exception but the rule as can be seen in the *N*-node. The graph also indicates the procedure of extending the Lexicon Graph by adding additional nodes and relations.

3.2.2 The mesostructure in the Lexicon Graph view

For the representation of lexical knowledge a further analysis of this knowledge is necessary. In existing applications this knowledge is represented either in relational structures or hierarchies. The first can be implemented in inflection tables or simple references, which can be interpreted as tables with one column and line. The latter can be exemplified by an ontology as described by Farrar and Langendoen (2003). As the table representation can be represented in a hierarchical form, the hierarchical representation will be taken into consideration. A procedure for the representation of tables in hierarchical form, called *shredding* can be found in the representation of relational databases in XML as described in Draper (2004)(p. 335ff).

Another way of representing lexical knowledge relies on inference techniques, which can, for example, be represented in DATR. In DATR three different inheritance rules are distinguished (see Gibbon (2002a)) which are:

1. Sequences and atoms: the value is directly given by a terminal symbol, no further inheritance needs to be processed. Example: in a translation based dictionary for English and German the English word *unit* receives the value *Einheit* as a simple terminal value.
2. Local inheritance: the value is directly given at a different position of the knowledge base. Example: in a monolingual English dictionary with part of speech information the English word *unit* is classified as *n* which is a reference to the list of abbreviations where this is evaluated as *noun* (e.g. see Crowther (1995))
3. Global inheritance: a part of the value is given at a different position of the knowledge base but this information needs to be combined with either a terminal value or a locally inherited value. Example: in an English monolingual dictionary the word *relate* is classified as *Vn*; this is evaluated as a *transitive verb* (e.g. see Crowther (1995)); verbs are inflected and can be marked for third person, using an unmarked form (i.e. in English the *infinitive*) and the appropriate affix (in English *-s*). A query for the third person singular of the word *relate* therefore infers from its class the inflectional information that is recombined with some local, initially available terminal symbol.

3.2.3 The macrostructure in the Lexicon Graph view

The macrostructure in the Lexicon Graph view is rather simple in comparison to the other lexicon structures. It consists of defining the primary sorting strategy and the headword. The first step is to define the lexical headword

required for an application. For the headword selection the lexicon graph has to be restricted to the node of a specific type and with the appropriate lexical value, i.e. the concrete lexicon item. Based on the headword a sorting strategy can be implemented, such as alphabetically by the orthography, or by semantic properties according to a given ontology. Each data category can be used for sorting, either as part of a primary sorting strategy or as a secondary or even n-ary sorting strategy taking into account different classes of lexical properties.

3.2.4 Summary: Lexicon Graphs as a generic model for lexicons

The Lexicon Graph provides a model that allows a generic representation of lexical structures, with the bits of lexical information for each data category forming nodes in a graph related to other nodes, which could be the other bits in the same row of a table representation. The lexicon microstructure, mesostructure and macrostructure can be defined from the Lexicon Graph perspective, using specifications of subgraphs and selecting start nodes for accessing the graph. Lexicons can also be combined, and parts of the lexicons can be selected, which both contribute to an increased portability. The consequent use of pointers avoids duplication of information and is part of a modern approach to declarative lexicons. The indefinite number of lexical data categories allows the use of Lexicon Graphs from different theoretic backgrounds.

3.3 Lexicon format

The strategy for the creation of lexicons is based on standard data representations and can be modeled according to a syntax in XML. Figure 3.5 shows the structure of the lexicon grammar with the element names and corresponding attributes in a simple tree. The three sets of lexical items, relations and knowledge with their individual elements are not connected, as they are different branches of a tree grammar. Each lexical relation even has a subtree with their source and target.

Although the Lexicon Graph seems to contain a flat hierarchy, the structure is complex due to the interrelation of the different elements and subtrees. Figure 3.5 illustrates the relation of the lexicon structures. The lexical items are connected to the source branches and to the target branches of the relation subtree, the relation type is connected to knowledge items, which are used for the inclusion of knowledge about the knowledge type.

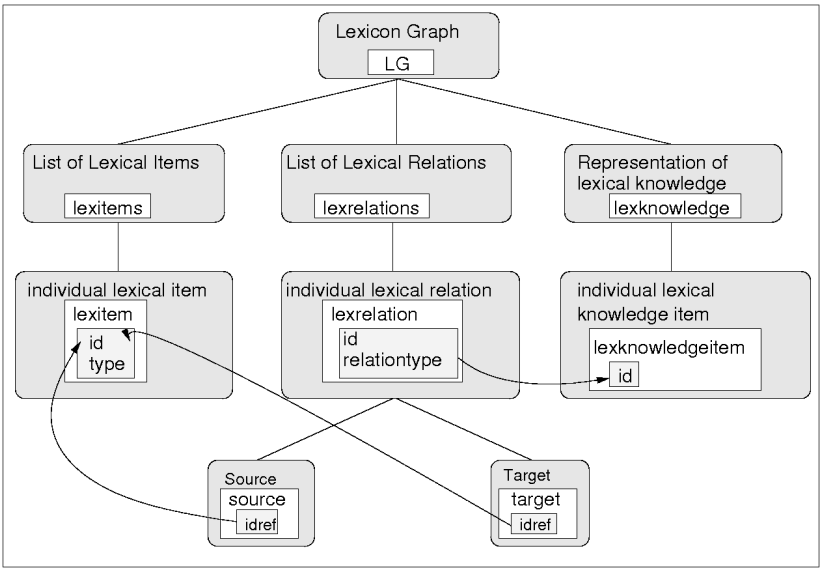

Fig. 3.5. Grammatical components of a Lexicon Graph implementation, the structure is named, the element name is given with corresponding attributes, relations in the Lexicon Graph grammar included by the curved lines

The implementation of the Lexicon Graph in XML results in a data centered XML structure, rather than a document centered one (see Kay (2004b)). This is due to the fact that every bit of information is included in specifying elements. Data centered XML could be represented in the form of tables of relational databases using a technique called *shredding* (Draper (2004), p. 335ff). Nevertheless, there are good reasons for not using *Relational Data Base*s(RDB) but rather XML:

- The structure of the lexical information can be validated according to XML standards, e.g. for structural properties instead of the strict data types definition of RDBs as strings, integers, floats, etc.
- Standard XML tools can be used for manipulation, validation, and retrieval
- The format is portable to other platforms and tools
- Coherent storage of a lexicon, not in many different, unrelated tables but in one document.

Lexical Relations in an XML based implementation of a Lexicon Graph are therefore inherent and explicit and can be automatically validated and checked for consistency, completeness and uniqueness.

The relation of the lexicon structures is modeled with the XML inherent *id-idref* mechanisms, where elements receive a document internal unique identifier which can be used in addressing the element from other elements using the identifier reference. The relation of identifiers references to identifiers is an interpretation of the id and idref attributes already included in the XML specification to define a way of linking and referencing within a document. Although this reference is not obvious from the data structure, the interpretation as a relation is fixed and can be evaluated.

3.3.1 Lexicon Graph element definition

The distinction between lexical items, relations and knowledge illustrated in Fig. 3.5 anticipates the elementary units of the Lexicon Graph, which are defined in a document grammar. The elements of the Lexicon Graph are:

Lexical items which contain the individual items, such as orthography, definitions, phonemic transcriptions, etc.

Lexical relations which define the relation of lexical items, referring to the items and classifying the relation

Lexical knowledge encodes a representation of knowledge items in attribute value form

Linking to special data is an exceptional pseudo elementary value for lexical items which allows the inclusion of non textual data such as speech signals and pictures among the lexical items. The intended procedure is similar to the one defined for web pages: non textual resources are stored externally and pointers from the textual representations refer to these external sources. Textual elements, even marked up according to other document grammars can be included as an embedded structure, interpreted as an element according to the lexicon graph grammar.

The elementary values are clustered within container elements for each of the three types of elementary values, but are not further structured as this is part of the presentation structure of the individual lexicon output with its microstructure, mesostructure and macrostructures.

The definition of the Lexicon Graph elements does not anticipate the content of the elements besides being of a textual type. This is especially true for the lexical items, which can be referers to other types of data, and lexical knowledge which can contain a complete knowledge representation

nested in the Lexicon Graph XML implementation using the XML namespace definition, where within one element that allows parsable character data (PCDATA), i.e. character strings that are either interpretable by a parser as reserved strings or as character strings. Within PCDATA sections XML elements according to other document grammars can be applied, if the document grammars are identified by a prefix of the element name. This procedure, using XML namespaces (see Bray et al. (1999)) of an element name needs to be defined to refer to a specific document grammar in the document type declaration of the XML document.

3.3.2 Specifying the elements

The elements of the Lexicon Graph lack some additional specification. These specifications are further defined for referring and content type classification where appropriate. In the XML context the question of including information in attributes or elements is sometimes conducted with almost religious eagerness, but this discussion is not conducted here. Speaking of attributing the elements this section here defines the units of information that are required for the elementary types to reference and identify them efficiently. In the implementation this is done in XML attributes.

The following attributes are defined:

Identifiers (id) for all elements;
References to identifiers (idref) for referring from relations to either items or knowledge

The value of the identifiers is almost arbitrary but as the *id-idref* mechanism is part of the XML specification certain restrictions apply. The availability of the reference mechanism can easily be used though it imposes certain restrictions. The XML specification states that the value of an identifier is supposed to be a string with only certain characters and sequences. These restrictions result in an ID being an enumeration as syntactic erroneous, while the same string starting with a letter prefix can be correct. For a formal definition of the requirements see Bray et al. (2000), Section 3.3.1 *Attribute Types*.

In practical applications it is appropriate to select a character prefix, and enumerate the lexical items or relations without space characters. Generating these IDs by an application can, for example, include the node number when generating the identifier, and some random character string. The only additional condition is that an identifier needs to be unique within a document and an identifier reference needs to have a matching identifier. Other schema languages such as XSchema (Thompson et al. 2001 (2004a) and Thompson

et al. 2001 (2004b)) allow further restrictions to the values of an attribute, for example to restrict the possible values for identifiers of a specific type to a defined range of values.

The complete document grammar for a Lexicon Graph in a DTD format is rather simple.

```
<!--
        LG DTD 2003-12-01
        Version 0.1
        Thorsten Trippel

        Universitaet Bielefeld
        Fakultaet fuer Linguistik und
        Literaturwissenschaft
        DTD for the Lexicon Graph Model implementation
-->

<!-- ENTITIES -->

<!-- ELEMENTS -->

<!--  LG is the root element for Lexicon Graph
implementation -->

<!ELEMENT LG            (lexitems,relations,knowledge)>

<!--  Three basic elements per lexicon -->

<!ELEMENT lexitems     (lexitem+)>
<!ELEMENT relations     (relation+)>
<!ELEMENT knowledge     (know+)>

<!-- lexitems -->

<!ELEMENT lexitem (#PCDATA|object)*>
<!ELEMENT relation (source+,target)>
<!ELEMENT know (conceptname*, knowcat+)>
<!ELEMENT object EMPTY>
<!ELEMENT source EMPTY>
<!ELEMENT target EMPTY>
<!ELEMENT conceptname (#PCDATA)*>
<!ELEMENT knowcat (#PCDATA|like)*>
<!ELEMENT like EMPTY>
```

```
<!-- ATTRIBUTES -->

<!ATTLIST lexitem id ID #REQUIRED>
<!ATTLIST relation type CDATA #REQUIRED>
<!ATTLIST source idref IDREF #REQUIRED>
<!ATTLIST target idref IDREF #REQUIRED>
<!ATTLIST know id ID #REQUIRED>
<!ATTLIST like idref IDREF #REQUIRED>
<!ATTLIST object
                type CDATA #IMPLIED
                uri CDATA #REQUIRED>
```

The type of lexical information has to be included to allow the differentiation of different lexical classes that might share the same form. One example of this is the case where an orthography shares the same form with a phonemic transcription. The process of typing lexical classes is rather complex and dealt with in the following section.

3.3.3 Typing items and knowledge

The process of typing lexical classes is complex, though the number of types of items and knowledge is relatively small and fixed for each application lexicon. A controlled vocabulary would be the obvious solution for typing lexical items, knowledge and relations, but fixing the types requires a prior knowledge of the complete and required categories. For different applications, especially while merging lexicons, this can hardly be guaranteed, wherefore at least a principal openness is required. Nevertheless, for individual lexicons a type list is desirable to validate the individual items.

A generic LG approach does not describe the semantics of the categories but allows the free representation of data categories. The lexical types and knowledge categories have to be interpreted as features of a lexicon, therefore they are not to be prescribed but left to be represented.

The complete openness of types raises the question what a *type* is supposed to be. As mentioned before, some lexical items can be typed as *headwords* or *phonetic transcriptions*, *orthography*, *definition*, etc. Some lexical items can be of more than one type at a time, in print dictionaries, for example, the *headword* type and the *orthography* are often identical, usually it is said that the orthography serves as the headword. Inherent to these types can be other information. For example, a monolingual English dictionary may have an orthography type, but of course this implies that the word belongs to the English language.

A type for a lexical item is therefore a characteristic attribute, i.e. a describing feature. Lexical items can be described by different features, such as technical, administrative and linguistic features. Such descriptive features are metadata on the lexical items, which have to be integrated into the Lexicon Graph in some way.

3.3.4 Integration of metadata in a Lexicon Graph

The integration of metadata in the Lexicon Graph Model with the distinction of relations, knowledge and lexical items. The motivation for the inclusion of metadata will be discussed in Section 6 as well as the metadata categories (Section 6.3.3). However, the integration of metadata of existing lexicons in the Lexicon Graph Model is illustrated by Figure 3.6. Metadata items from a metadata set are related to whole sets of lexical items to describe them.

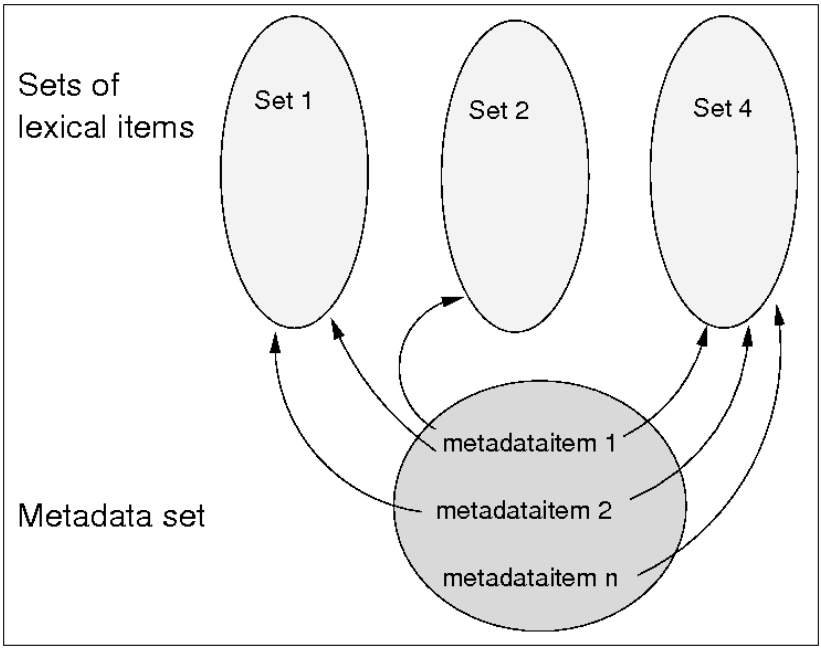

Fig. 3.6. LG metadata — set model

The metadata information is another set of items in the lexicon, where these items can refer to whole subsets of lexical items. By ways of the LG

model this is another relation type, but not another formalism. In a relational model this would be more complicated, as some metadata items refer to more than one set, i.e. column of the database, and some sets receive more specifying metadata relations. Hence, by referring to the sets of lexical items by the metadata, the distinction of sets can be different from the first partition given by the properties of the database. The reason for this is that the properties defined by the metadata can be characteristics of the sets, but as well the properties can be shared characteristics of the elements of the sets. To allow for this differentiation, the implementation allows a metadata description on all levels of the lexicon model, see Figure 3.7. It shows that the metadata elements can be pointing to all levels of the lexicon, to the lexicon as a whole, sets of lexical items, relations, or knowledge or individual items.

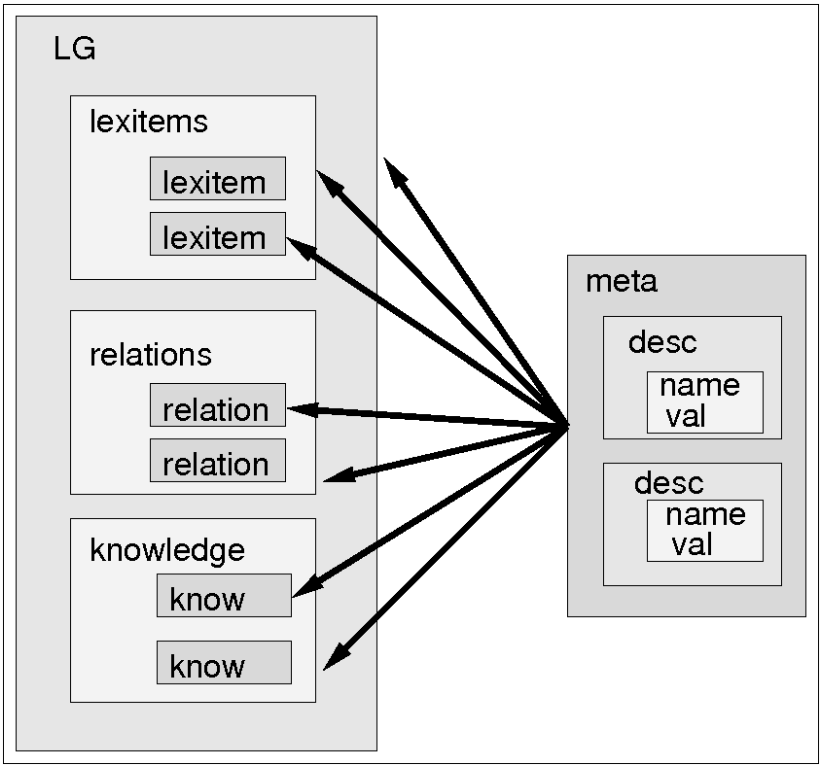

Fig. 3.7. Lexicon Graph (LG) metadata model

The differentiation between the lexicon as a whole (in Figure 3.7 called *LG*) and the individual data types of lexical items, knowledge and relations, each with the individual items, allows to distinguish between information on the elements of the sets and information on the sets. In other words, the sets that are used for the relations of the lexicon can be defined by the metadata. A similar approach to the metadata of corpora can be found in Trippel (2004) and Trippel et al. (2004a).

The question if metadata are different from other data in the lexicon context is not answered, i.e. what is the difference between the relation of a data item to some other item and to its metadata description. As each lexicon item has a unique identifier, no additional metadata are required for addressing a particular lexicon item. The classification could be given as a lexical relation *IsOfLexicalClass* which is the relation to the lexical class. The problem then is the case of homography again, or if one lexical item is the same as another item from a different class in terms of character representation. Although conceptually the solution to this problem would be in an ambiguous classification, i.e. the item is classified for more than one category, in practice the problem arises again when updating or deleting an item. This is briefly discussed in Section 3.5.1. Therefore a *type* is included for every lexical item which is the lexical data category associated with the item. Other lexical metadata are represented as a lexical item with the appropriate relation.

3.4 Representing existing lexicons

The Lexicon Graph model needs to be applied to existing lexicons. The reason for this is obvious: If the model does not suit existing lexicons and if there is no application using the formalism, then it is not necessary to discuss it further. In Section 2.2 a number of different lexicons and their structure was discussed, these will serve as the use case for the the LG paradigm.

The first samples will be taken from machine readable lexicons, as their structure is formally the most consistent. Different types of book dictionaries with partly variable applications are described later.

3.4.1 A system driving lexicon: The Verbmobil lexicon

The Bielefeld Verbmobil lexicon as described in Section 2.3.16 has 11642 lexicon entries which were converted from the relational table representation into an XML format which explicitly relies on the lexicon microstructure, i.e. every column of the lexicon table is decomposed into one element, each

row — being a lexicon entry — is serving as another element in which the microstructure is embedded. This leads to a total of 100994 different lexicon items with 27070 different values and types.

The immense numbers shows that each lexicon item appears per average 3 to 4 times in the lexicon, indicating a great number of redundancy ranging from single occurrences to 3991 instances of the label `N` as a syntactic category label for nouns. The size and massive redundancy of the Verbmobil lexicon cause the real challenge for an implementation that is supposed to represent these relations individually not having redundant lexical information.

The following is a part of the Verbmobil lexicon generated from the microstructure representation. The encoding follows the XML binding of the Lexicon Graph as previously introduced. For simplicity most relations are omitted.

```
<?xml version="1.0" encoding="utf-8"?>
<LG>
<lexitems>
 <lexitem type="aux" lexid="d0e17">haben</lexitem>
 <lexitem type="base" lexid="d0e20">SIMPLEX</lexitem>
 <lexitem type="comment" lexid="d0e44">@</lexitem>
 <lexitem type="compl_base" lexid="d0e26">@</lexitem>
 <lexitem type="ge_pre" lexid="d0e35">ge_pref</lexitem>
 <lexitem type="lemma" lexid="d0e5">"andern</lexitem>
 <lexitem type="nonsep_pref" lexid="d0e23">@</lexitem>
 <lexitem type="particles" lexid="d0e32">@</lexitem>
 <lexitem type="sep_particle" lexid="d0e29">@</lexitem>
 <lexitem type="stem_morph_segm_phon" lexid="d0e11">
         [?''En.d@r]</lexitem>
 <lexitem type="stem_morph_segm_orth" lexid="d0e8">
         ["ander]</lexitem>
 <lexitem type="stem_syncretism" lexid="d0e38">
         FRAGEN</lexitem>
 <lexitem type="suf_syncretism" lexid="d0e41">
         BEGEISTERN</lexitem>
</lexitems>
<relations>
<relation type="has_aux">
   <target idref="d0e17"/>
   <source idref="d0e5"/>
   <source idref="d0e8"/>
   <source idref="d0e11"/>
   <source idref="d0e17"/>
```

```
    <source idref="d0e20"/>
    <source idref="d0e23"/>
    <source idref="d0e26"/>
    <source idref="d0e29"/>
    <source idref="d0e32"/>
    <source idref="d0e35"/>
    <source idref="d0e38"/>
    <source idref="d0e41"/>
    <source idref="d0e44"/>
</relation>

[...]

</relations>
<knowledge>
 <know id="stem_sync_fragen">
  <conceptname>FRAGEN</conceptname>
  <knowcat>
    Knowledge about stem syncretism model FRAGEN
   </knowcat>
 </know>

 <know id="suf_sync_begeistern">
  <conceptname>BEGEISTERN</conceptname>
  <knowcat>
   Knowledge about suffix syncretism model BEGEISTERN
  </knowcat>
 </know>

 <know id="ge_pref">
  <conceptname>ge_pref</conceptname>
  <knowcat>
   Knowledge about prefix class ge
  </knowcat>
 </know>

 <know id="simplex">
  <conceptname>SIMPLEX</conceptname>
  <knowcat>
   Knowledge about SIMPLEX verbs
  </knowcat>
 </know>
```

```
 <know id="aux_haben">
  <conceptname>haben</conceptname>
  <knowcat>
   Knowledge about auxiliary class haben
  </knowcat>
 </know>
</knowledge>
</LG>
```

Figure 3.8 shows an illustration of the graph. Some relations however are not shown in the figure to enable legibility. The restriction to certain relations centering around either the lemma or the pronunciation indicates a possible macrostructure.

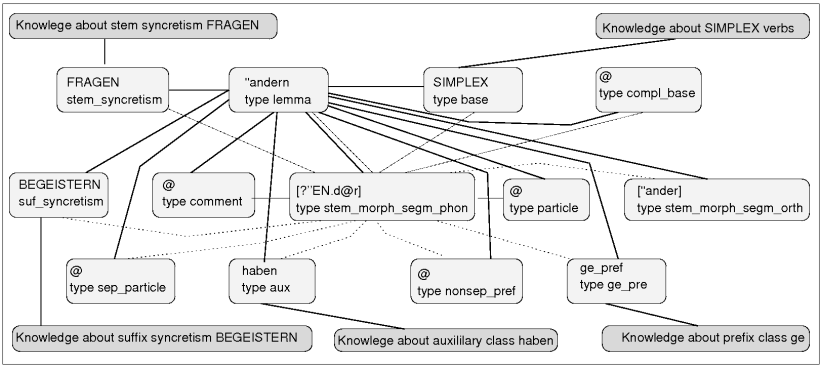

Fig. 3.8. Verbmobil lexicon in Lexicon Graph, resulting from one entry. Labels of the edges are omitted, relations for the lemma (thick lines) and pronunciation (dashed lines) centered views are shown.

3.4.2 A lexicon exchange format: MARTIF

The MARTIF format as introduced in Section 2.3.14 serves as a further source for a hierarchical lexicon which is available in machine readable format. The sample lexicon that was used contains 19 terms from a sample termbase created for Trippel (1999), originally converted from a relational table structure into the hierarchical tree structure. The result is a microstructure with 71 lexical items and 674 relations between them.

Although the conversion is straight forward, the microstructure is explicitly represented first for each concept in the termbase, the additional question is how to represent the hierarchies that are part of the original format and how the original contained information can be maintained. The answer for this is to create an appropriate hierarchy in the knowledge section.

Figure 3.9 shows the microstructure of a small part of the MARTIF coded termbank from Trippel (1999) in a Lexicon Graph. Again, the edges are not labeled in the figure and the knowledge is only partially represented. In this case the knowledge is represented for the author of the entry and a source of the lemma.

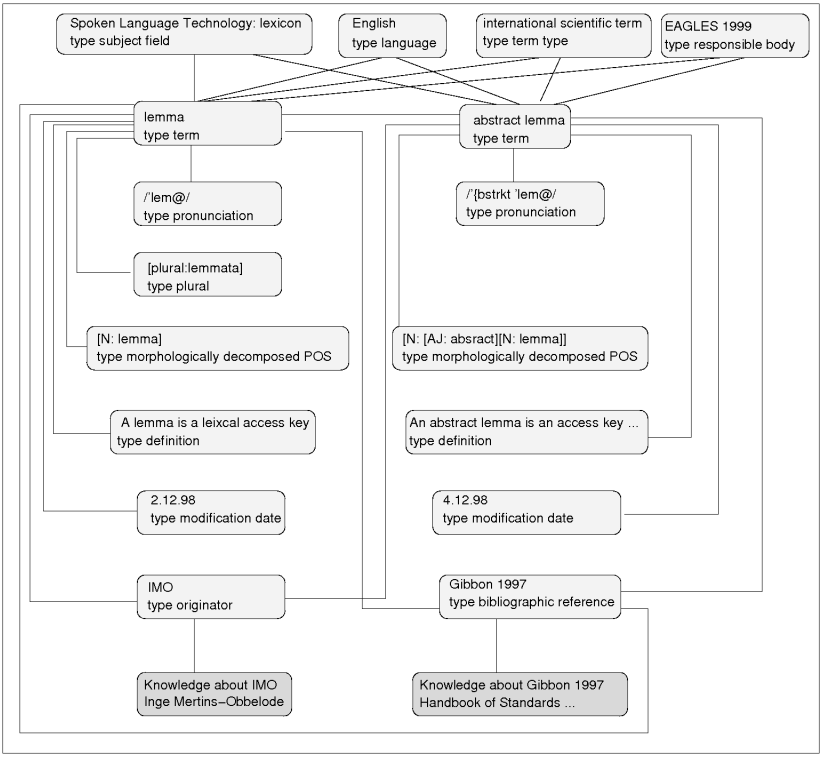

Fig. 3.9. MARTIF coded termbank in the lexicon graph, two lemma based entries are shown

The lexicon graph representation can be generated from the original MARTIF format by extracting a flat table like structure from the MARTIF file using a transformation of formats. From the table structure the transformation into the LG format is the same as for the Verbmobil lexicon. The resulting lexicon graph is represented by the following XML code, which is again abbreviated for legibility. As the MARTIF document provides implicit links to related terms the status of the related terms was adjusted and related terms are treated as the same type, namely that of a term.

```
<LG>
 <lexitems>
  <lexitem type="plural" lexid="d0e12">
   [plural: abstract lemmata]</lexitem>
  <lexitem type="pron" lexid="d0e15">
   /'\{bstr\{kt 'lem@/</lexitem>
  <lexitem type="subjectField" lexid="d0e3">
   Spoken Language Technology: lexicon</lexitem>
  <lexitem type="lang" lexid="d0e6">en</lexitem>
  <lexitem type="term" lexid="d0e9">lemma</lexitem>
  <lexitem type="plural" lexid="d0e11">
   [plural: lemmata]</lexitem>
  <lexitem type="pron" lexid="d0e14">/'lem@/</lexitem>
  <lexitem type="term" lexid="d0e16">
    abstract lemma</lexitem>
  <lexitem type="originator" lexid="d0e17">
    IMO</lexitem>
  </lexitems>
  <relations>
   <relation type="has_subjectField">
    <target idref="d0e3"/>
    <source idref="d0e3"/>
    <source idref="d0e6"/>
    <source idref="d0e9"/>
    <source idref="d0e11"/>
    <source idref="d0e14"/>
    <source idref="d0e16"/>
   </relation>
[...]
   <relation type="has_relatedTerm">
    <target idref="d0e9" orgidref="d0e18"/>
    <source idref="d0e3" orgidref="d0e3"/>
    <source idref="d0e6" orgidref="d0e6"/>
    <source idref="d0e16" orgidref="d0e9"/>
```

```
    <source idref="d0e12" orgidref="d0e12"/>
    <source idref="d0e15" orgidref="d0e15"/>
    <source idref="d0e9" orgidref="d0e18"/>
   </relation>
  </relations>
  <knowledge>
   <know id="imo_originator">
    <conceptname>IMO</conceptname>
    <knowcat>Knowledge about Inge Mertins-Obbelode
    </knowcat>
   </know>
   <know id="know_1">
    <conceptname>
     Spoken Language Technology: lexicon</conceptname>
    <knowcat>
     Knowledge about Spoken Language
     Technology: lexicon
    </knowcat>
   </know>
  </knowledge>
</LG>
```

3.4.3 Print dictionaries: the Oxford Advanced Learners Dictionary

For the human readable dictionaries the procedure is similar to the one used for MARTIF above, but additionally the digitalization needs to be done. The digitalization in itself is not trivial as usually a lot of abbreviations are included as described in Section 2.3.1.

In the digitalization process a flat hierarchy for the lexicon microstructure is created. For each lexicon entry it is possible to create a lexicon table as shown in Table 3.1. The example of Section 2.3.2 was used. This example are represented as two lexicon entries in a sense enumeration lexicon, though those lexicons could take more subentries, which are omitted here. Expanded, these are 10 lexical items and 67 lexical relations.

The following shows a possible implementation of the same structure in XML.

Table 3.1. Lexicon entries from the Oxford Advanced Learners Dictionary (Crowther (1995)).

lemma	unit
pronunciation	/ ju:nIt/
part of speech	n
definition	a single thin, person or group ...
example	a family unit
example	a course book with twenty units.

lemma	unit
pronunciation	/ ju:nIt/
part of speech	n
definition	a fixed amount or number used as a standard of measurement
example	a unit of currency
example	The metre is a unit of length.
example	a bill for fifty units of electricity

```
<lexicon>
 <entry>
  <orth>unit</orth>
  <pron>/ju:nIt/</pron>
  <pos>n</pos>
  <def>a single thin, person or group that is complete
   in itself, although it can be part of sth larger
  </def>
  <example>a family unit</example>
  <example>course book 12with twenty units.</example>
 </entry>
 <entry>
  <orth>unit</orth>
   <pron>/ju:nIt/ </pron>
   <pos>n</pos>
   <def>a fixed amount or number used as a standard of
    measurement</def>
   <example>a unit of currency</example>
   <example>The metre is a unit of length. </example>
   <example>a bill for fifty units of electricity.
   </example>
  </entry>
```

```
</lexicon>
```

A graphical representation of these relations is given in Figure 3.10. The assumed macrostructure for this lexicon, which is classified as a sense enumeration lexicon, is seen from the lexicon item typed as definition. Other relations are omitted for legibility reasons again.

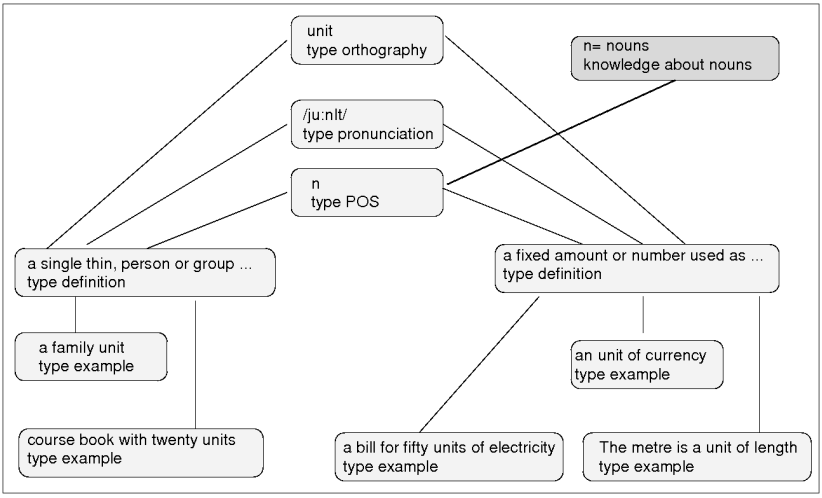

Fig. 3.10. Lexicon Graph entry from the Oxford Advanced Learners Dictionary (Crowther (1995)) for the English word *unit*. The lexicon entry is shown in Section 2.3.2. Relations are shown for a definition based macrostructure

A possible rendering in XML of this graph is indicated in the following XML code. The different entries in a sense enumeration lexicon become obvious from the relations having the different definitions as source.

```
<LG>
  <lexitems>
    <lexitem type="def" lexid="d1e9">
      a single thin, person or group that is complete
      in itself, although it can be part of sth larger
    </lexitem>
    <lexitem type="example" lexid="d1e11">a family
       unit</lexitem>
    <lexitem type="example" lexid="d1e13a">
       course book with twenty units.</lexitem>
```

```
    <lexitem type="orth" lexid="d1e2">unit</lexitem>
    <lexitem type="pron" lexid="d1e4">/ju:nIt/
     </lexitem>
    <lexitem type="pos" lexid="d1e6">n</lexitem>
    <lexitem type="def" lexid="d1e8">
       a fixed amount or number used as a standard of
       measurement</lexitem>
    <lexitem type="example" lexid="d1e10">
                    a unit of currency</lexitem>
    <lexitem type="example" lexid="d1e13">
       The metre is a unit of length. </lexitem>
    <lexitem type="example" lexid="d1e15">
       a bill for fifty units of electricity.
    </lexitem>
  </lexitems>
  <relations>
    <relation type="has_orth">
         <target idref="d1e2"/>
[...]
         <source idref="d1e15"/>
      </relation>
[...]
      <relation type="has_def">
         <target idref="d1e8"/>
         <source idref="d1e2"/>
         <source idref="d1e4"/>
         <source idref="d1e6"/>
[...]
      </relation>
      <relation type="has_def">
         <target idref="d1e9"/>
         <source idref="d1e2"/>
         <source idref="d1e4"/>
         <source idref="d1e6"/>
[...]
      </relation>

[...]
      </relation>
   </relations>
   <knowledge>
      <know id="know_1">
         <conceptname>n</conceptname>
```

```
        <knowcat>knowledge about nouns</knowcat>
      </know>
    </knowledge>
</LG>
```

Again, some relations are omitted for legibility reasons. One problem becomes obvious in this implementation: the sense enumeration lexicon often combines two or more entries into one entry, the examples however are restricted to one definition and not to all of them. The interpretation is nevertheless simple. Starting from the orthography, the different examples can nevertheless be examples for the orthography though they apply to different definitions. The actual entry of the OAD lexicon can be created by first querying for the different orthographies, search for all POS, pronunciation and definition information that are related to these orthographies by being the target of a relation where the orthography is the source and within that process querying for all relations of the type `has_example` where the definition is the source. However this is part of the lexicon model evaluation which is discussed later in Chapter 4.

3.4.4 Feature structure lexicon: HPSG lexicon in LG

*Feature Structure*s (FS) as discussed in Section 2.3.19 can be represented as *Directed Acyclic Graphs*(DAG) as for example Shieber (1986) (Chapter 3, Section 2.4) points out. Section 2.3.19 already shows an XML implementation of an HPSG lexicon entry. This lexicon entry can easily be transformed into a Lexicon Graph format, though some problems have to be dealt with:

- some of the values of the feature structures are sequences, called *lists* in ISO DIS 24610-1:2004 (2004). A general graph, however, does not include sequences. It is also unclear, how sequences should be included in DAGs, but the LG model allows the coding of sequences as knowledge.
- The example FS includes classes highlighted by typographical features with special semantics, such as *word* marked up by italics as a class. This was not represented in the XML format, so in the transformation these have to be dealt with, as they are coded as features with an empty value.
- The LG model uses directed graphs, but as Shieber (1986) (Chapter 3, Section 3.1) states that there is not a principal reason for interpreting a FS bottom-up or top-down, both have to be represented in an LG.

To accomplish the transformation the FS is read for listing and typing the lexicon items, i.e. the values of the terminal symbols of the FS and the names

of the features as representatives for complex or non-terminal symbols. For the extraction of the relations between these lexical items, every lexical item is taken as a target and the lexical item that refers to it is used as the source of the lexicon relation of the lexicon graph. An additional step is required for the reentrancy into the graph, i.e. substructures and values that are declared once and reused at a different position. One way of handling reentrancy is by selecting the reentrancy code, i.e. the position where a feature is referred to, and treat the referenced structure as if it was attached to that position.

The implementation of this transformation relies on XSLT and a simple shell script and is processed in three steps:

1. The lexicon items are listed.
2. The FS is read from right to left, i.e. for each feature this feature is listed as a target of a lexical relation and all features that are grandchildren in the XML structure of this feature are used as a source. In the case of the feature not having grandchildren in the structure, the source is the value of the terminal node.
3. The FS is read from right to left, i.e. each terminal node as a target receives its parent feature as source of a lexical relation, each feature its grandparent feature as the source of a lexical relation. In the case of a feature being part of a reentrancy structure, every grandparent feature is used for this purpose.

The knowledge of the sequences is not maintained in the current implementation as this would extend the focus of this section. The complete code is available, see Appendix 10 for details. One component of knowledge could be the required function to extract the original feature structure from the lexicon graph. This is indeed lexical knowledge as the relation of the features can be found in the graph, but the reentrancy position is not maintained in the lexicon graph, as it is included explicitly. The query used for the generation of a feature structure can also be used to evaluate the transformation. The evaluation can be understood as comparing the initial FS to the generated one. A part of the query producing the sample feature structure from the LG is shown below.

```
let $file :=doc("LGfile.xml")
for $lexitem in $file/LG/lexitems/lexitem
where $lexitem/text()="dog" and $lexitem/@type="orth"
return
 <fs>
  <f org="list" name="{$lexitem/text()}">
   <fs>
```

```
    <f name="{
     for $lexitemtype in $lexitem/@type
     return $lexitemtype}">
     <str>
      {
      for $relation in $file/LG/relations/relation
      where $relation/source/@idref=$lexitem/@lexid
      return
       for $lexitem2 in $file/LG/lexitems/lexitem
       where $lexitem2/@lexid=$relation/target/@idref
       return $lexitem/text()
      }
     </str>
    </f>
    <f name="{
     for $lexitem4 in $file/LG/lexitems/lexitem
     where $lexitem4/text()=$lexitem/text() and
           $lexitem4/@type="FSR_class"
     return
      for $relation2 in $file/LG/relations/relation
      where $relation2/@type="is_instanced_by" and
            $relation2/source/@idref=$lexitem4/@lexid
      return
       for $lexitem3 in $file/LG/lexitems/lexitem
       where $lexitem3/@lexid=$relation2/target/@idref
         and $lexitem3/@type = "FSR_class"
         and $lexitem3/text()="word"
         or $lexitem3/text()="sentence"
       return $lexitem3/text()
       }"/>
   </fs>
 </f>
</fs>
```

Another form of evaluation is a graphical evaluation: a graph drawing program is used to draw a graph from the LG that was generated from the feature structure. For this purpose the LG is transformed into the graph-drawing input format, in this case a stylesheet creating a input file for the *dot* graph drawing software (see Appendix 10 for details). By comparing the FS directed graph created by hand with the automatically generated lexicon graph, the transformation can be assessed. For the reason of complexity the complete graph cannot be printed here. The assessment shows that the represented structures are identical.

3.5 Lexicon workbench for Lexicon Graphs: Adding, changing and correction of lexicon entries

A lexicon workbench based on the Lexicon Graph Model can be implemented using the document grammar shown and used in the previous section. However, an essential component for a lexicon system is not only a way of implementing existing lexicon entries but also to maintain such a lexicon system. Maintenance of a lexicon requires adding, changing and correction of lexicon entries. Accomplishing this is the focus of this section.

3.5.1 Requirements for the lexicon workbench

Maintaining a lexicon requires some actions, namely insertion of entries and updating entries. For classic database management, deletion is another issue, but for a lexicon database, which is created based on corpora and lexicons this is not needed as the only problem that can arise is that an entry becomes deprecated or gets some other form of label. An entry does not cease to exist in the original source. However, as lexicons and corpora are used to enrich each other, errors can occur in either of them resulting in a continuation in the other format. Consequently the need to alter a lexicon entry exists. Two different kinds of changes can be distinguished:

1. A lexical item is erroneous, i.e. either the content or the type is wrong. In both cases the content can be altered. However, it has to be made sure that the altered entry does not already exist in the lexicon
2. A lexical relation or knowledge is wrong, in which case the wrong part can be deleted, and a new entry with the modifications can be inserted into the lexicon base.

Insertion of lexicon entries appears whenever the source database is extended or if two existing dictionaries are merged. The problem with both is that the extension can contain lexical items that are already part of the lexicon base. One could argue that in the case of a differing lexicon microstructure, which means differing lexical information, it is acceptable to have two entries that refer partially to the same lexical items. But the idea of merging of lexicons is not to double the number of entries but to combine the lexical information. Therefore, a mechanism has to be applied preventing duplicates.

The problem of merging complex structures is not new at all. In the context of the unification of feature structures this concept has been discussed frequently (see for example Shieber (1986), Chapter 3, Section 2.3). The unification of two feature structures is the process of creating the feature structure that contains all information that is part of these two structures but no

extra information. Formally the unification of two feature structures FS_1 and FS_2 to a feature structure FS_0 can be expressed by two subsumptions so that $FS_1 \subseteq FS_0$ and $FS_2 \subseteq FS_0$, usually this is written as $FS_1 \cup FS_2 = FS_0$. The unification of lexicon graphs, i.e. the merger of two lexicon graphs, is defined accordingly.

For targeting duplicates, it is necessary to define when two lexical entries are in fact the same. The simplest thing is to say that two items are the same if they are represented by the same character string, leading and trailing space characters and space character clusters excluded. This rather strict definition allows the distinction of case sensitive orthography as well as the representation of different coding systems without intermingling them, for example, different forms of compounding — e.g. with hyphenation, blanks, direct appending — are treated as separate entries.

As lexicons can become rather large with many lexical items, the system has to be robust to allow for different lexicon sizes. Robustness for a lexicon system in this context refers to the performance of a system and the scalability of a system. Scalability is the aspect concerned with the question if the system scales with the size, i.e. if a small lexicon can be processed in a short period of time, that does not mean that the performance is still acceptable for large ones. This issue of efficiency in the implementation however is not the target of the model.

3.5.2 The Lexicon workbench

Lexicon insertion and updating can be addressed as the same issue. The reason for this is that updating lexical items does not include the change of the item but a change in the description of the item, even a typographical error can be maintained in the database as an erroneous variant of another item, for example, a spell checker lexicon could be based on a lexicon of erroneous forms instead of reporting words that are not in the lexicon of correct forms. Hence, lexicon items can be inserted if they are different from the existing ones. Otherwise only the description can be added.

Figure 3.11 shows the architecture of a system inserting and changing a lexicon entry. The system starts with a provisional lexicon entry in the LG paradigm, i.e. one lexicon entry is represented with all of its items, relations and inserted knowledge. The lexicon is then merged one at a time to the existing lexicon.

This happens in three phases, to be realized as different modules, namely for lexical item insertion, lexical relation insertion and lexical knowledge insertion.

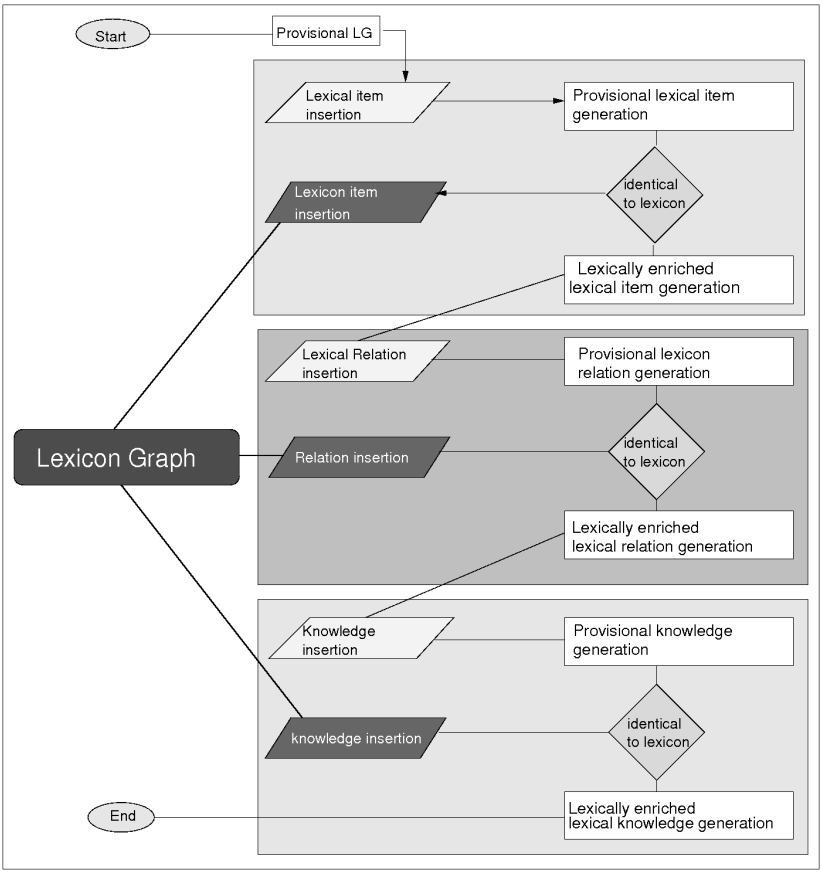

Fig. 3.11. Processing an addition of lexicon entries to a lexicon graph

To insert lexical items, they are treated one at a time, as a very small lexicon. In this case the insertion of a lexicon item is a merging task. This is accomplished by generating a provisional lexicon item, i.e. with its metadata descriptions and identifiers in the lexicon format. This lexicon item is then queried for in the lexicon database. If this lexicon item is not identical to one existing in the lexicon database, i.e. if a query for this item returns zero results, it is inserted into the lexicon database, using a lexicon item insertion module. To ensure robustness, i.e. that the system has a way of handling inconsistencies in the source, the same lexicon entry is then processed again,

resulting in the other possibility, namely that the lexical item is available in the database.

If the lexicon item exists in the lexicon database then only the description needs to be added, i.e. the description items need to be compared, and if they are not identical another description is added. For merging lexicons the source is one of the bits of meta information that can be added. In any case, the provisional lexicon item needs to be altered for the system as it contains the identifiers that are still used by the lexical relations. Consequently the retrieved identifier from the lexical database has to be used with the lexicon item, enriching it from the other lexicon.

Starting with two sets LEX_1 (with an element l_1) and LEX_2 (with l_2 respectively) of lexical items, the merger of these lists means to take one list completely and to take from the second list only the items that are not in the first one.

Different cases can be distinguished for the creation of the union, these can be differentiated by the content of the lexical items. A content here is used in the XML sense, i.e. the content of the element encoding the lexical item, not attributes such as identifier or types. The cases to be differentiated are:

1. If the content of the lexical item is identical, i.e. $l_{1_{content}} = l_{2_{content}}$ there are two cases:
   a) The types $l_{1_{type}} = l_{2_{type}}$, in which case the two items are identical; for further processing the identifiers need to be adjusted and the references.
   b) Otherwise $l_{1_{type}} \neq l_{2_{type}}$, hence the entry l_2 can be added to LEX_1 under the premise that the identifiers are unique, i.e. the identifiers might need to be adjusted.
2. The content is not identical, i.e. $l_{1_{content}} \neq l_{2_{content}}$, then before adding l_2 to LEX_1 it has to be checked if there is another Element l_2' with identical content. If not, it can be added, but if, than the types have to be investigated.

Afterwards the relations have to be handled, though before relations can be compared to existing ones, the identifiers that are referred to in the relations need to be adjusted to the identifiers that are used or that are prepared to be used in the reference lexicon. If the identifiers are adjusted, the comparison can be accomplished, i.e. it can be evaluated if a relation pair exists and if not it can be added to the reference lexicon. The same procedure can be used for the knowledge component, though the difficulty arises here that the content of the knowledge component has a different format depending on the formalism

that is used. The knowledge itself can be an XML tree, and the problem of comparing trees automatically is not solved yet.

For processing the relations, every relation from the provisional lexicon entry with the enriched lexicon items are processed separately. To avoid redundancy again it is checked if the relation already exists in the lexicon database. As the lexical relations contain only references to lexical items, the references to the identifiers have to be altered to the identifiers inside of the lexicon database. The database is then queried for the altered relation, if the relation is not available, it needs to be inserted.

For the insertion of lexical knowledge the same procedure is applied as for the insertion of lexical relations. As the lexical knowledge in the provisional lexical entry includes references to relations or items, these need to be altered into the references to the items in the lexicon, resulting in a provisional knowledge representation. The knowledge is then queried in the reference lexicon database and if not already inside, it is inserted. If it is available, the description is again compared and if deviating, added.

With the insertion of the lexical knowledge from the provisional lexicon entry, all information from this lexicon entry is then in the lexicon database and the insertion is completed.

A more formal description of the merging process is given below in pseudocode, the lexicon Lex_B is merged into Lex_A. The identifiers are abbreviated as ID, the element names are taken from the LG DTD.

```
function generate-new-id ()
   randomly create new ID
   if new ID in Lex_A then
      call generate-new-id
   else return new ID
end function

for each ID in lex_B
do
 if ID exists in lex_A  then
   replace ID and IDREF-references in lex_B
    by generate-new-id
 else next
done

for each LEXITEM in lex_B
do
  if not exists LEXITEM (TYPE and CONTENT) in lex_A
     insert LEXITEM into lex_A
```

```
  else next
done

for each RELATION in lex_B
do
  if not exists RELATION (TYPE and SOURCE and TARGET)
     in lex_A
   insert RELATION into lex_A
  else next
done

for each KNOW in lex_B
do
  if not exists KNOW (CONCEPTNAME and KNOWCAT) in lex_A
     insert RELATION into lex_A
  else next
done
```

The implementation deviates slightly from the pseudocode above, namely the disambiguation of the identifiers is processed as part of the other processes. However, in an implementation without the option of internal recursive programming and with the option of using the insert functionality of database systems such as Tamino (2003), the above pseudocode is more likely to be implemented directly. The identifiers need to be generated randomly to allow the disambiguation of identifiers, but the processes can be a script based application of four XQuery functions, each for the modules described above. These could be implemented in Tamino with the API defined by Hell (2003). The implementation for such a database system is part of a further development when extending the implementation.

The processing is done in two steps, first the lexical items from one lexicon is investigated, and compared to the second one. Afterwards the second lexicon without items from the first one are copied to the resulting lexicon. The processing of the relations, based on the modified lexicon entries of the first lexicon, is simpler. In a database join of the relations only double entries have to be removed. As all relations contain a type and exactly one target having a unique identifier, these can be used for iteration. The implementation has been accomplished using XQuery (see Boag et al. (2003)) using the saxon processor (see Kay (2004a)). The reference to the query that produces the merger of two lexicon graphs is given in Appendix 10.

The requirements for a merger of lexicons in the lexicon workbench are:

1. Maintaining the lexical items, relations and knowledge contained in the source lexicons. A complete evaluation requires that every item and relation from the source lexicons is taken and checked if it is also in the target lexicon.
2. Restricting the lexical items, relations and knowledge contained in the target lexicon to the ones available in the sources, i.e. the merger is not supposed to create additions based on its own functionality, even not using inference.

Testing these requirements is relatively time consuming. The first would demand that every lexical item from the source documents has to be investigated to see if it is already part of the merged one. One way of doing this is to merge the entry to the lexicon for a second time and compare the merged result with the previously merged one. However, the identity of these is not sufficient, as even a systematic error can result in the same thing. Nevertheless, the identity of an identity merger is a necessary condition.

The second requirement is even more complex to test, i.e. for every lexicon entry it has to be checked if a source exists; as the merger involves an adaptation of identifiers, the inverse process would be required, iterating over all source documents. In both cases the performance of the evaluation system would be a major issue, especially as a merged lexicon can be huge, therefore the performance is unlikely to be efficient.

As the merging operation as implemented is already a time consuming task, different strategies were taken.

Rigid syntactic evaluation: An advantage of operating in the XML paradigm is the availability of efficient tools for syntax evaluation. As the source and target lexicons both are from the same paradigm, i.e. are restricted to the same document grammar, these grammars can be used for evaluation. The evaluation includes checks for

- Well formedness, i.e. the proper embedding of structures
- Definiteness of structures, i.e. if the present structures are allowed by the grammar
- Uniqueness of identifiers
- Existing targets for id references, i.e. if for every reference to an identifier there exists a corresponding identifier.

However, this excludes the question of the complexity of such a lexicon and processing time, as the size of a lexicon grows with the size of the corpora and the relations can result in a problematic processing time. The question of complexity in the implementation is beyond the focus of this work.

Identity mapping, which means that a merger of two identical lexicons has to result in the identical lexicon. The lexicon is regarded as identical if it is identical in the order of lexical items, relations and knowledge items, but it is not required that the attributes are represented in the same order as the order of attributes of XML elements is defined as free (see the XML specification in Bray et al. (2000)) and different conforming tools may return different attribute orders. A bitwise comparison of files would be possible if the attribute order was fixed and could be determined and if all space characters such as line breaks are identical. This can be achieved by merging an existing lexicon not once but twice, using the same tools. In this case the ordering of attributes is usually, as it is not artificially randomized, constant. Hence if a lexicon L_1 is merged with itself, and the resulting lexicon L_1' is merged again with itself, resulting in $L_1^{''}$, then a bitwise comparison of L_1' and $L_1^{''}$ can be the same, which would be sufficient to the claim. It would as well be sufficient if L_1 and L_1' are bitwise identical. Both however, would be sufficient to the claim that the identity mapping results in the correct result. Although the condition is not necessary, resulting in the problem that even in the case of a failed test the identity of two lexicons could be true. Possible reasons for this result could be results from differing encoding standards, e.g. Unicode vs. ISO-8859-1. Another reason could be the treatment of white spaces, i.e. clusters of spaces are not interpreted in XML and can therefore be included or removed for readability or processing. The arbitrary ordering of attributes in XML has been mentioned before as a reason for non identity in the bitwise comparison. Using the same tools for performing the identity tests on a number of lexicon files of different sizes was successful.

Sample testing: a number of samples can be used, either investigated manually based on simple individual examples or with a simple program running the same operations, using a small test set. The procedure that was used was based on a small number of individual lexicon entries that were queried using XQuery; with this query the source format was created.

Maintaining the lexicon in this system is easy as long as adding lexical items is concerned. Changing an orthography is possible if the type system is sufficiently supported, otherwise some problems might be encountered, based on the consistency. I.e. it is possible that a change creates an already existing lexicon entry or that relations are not correct any more. These problems have to be foreseen and taken care of.

Conceptually, the solution is to insert a lexicon entry with a property that has a mistake in the entry. Merging this into the existing lexicon adds the relation to the erroneous entry, which marks it as wrong. Selecting all relations and items that are not related to an lexical item that is marked as erroneous then creates a consistent lexicon. While this deletion of an erroneous lexical item is only a logical one, the existence of the lexical item and the use of storage space is not changed.

Wrong relations need to be removed, in order to maintain consistency this is done by selecting a relation pair and if in the relation pair there is an erroneous target the whole relation has to be removed, if there is an erroneous source only the source is removed if there are other sources in the relation definition. The strategy of removing the relation could be handled in the same way as for the lexical items by tagging a relation as erroneous. However, the problem in the implementation here is that this is a property of the relation, i.e. a type of the relation. In the case where a relation does not only contain one but more source elements where not every one is false, this is not wanted. The solution here is to remove the erroneous part physically, i.e. by freeing the storage space.

The physical deletion of items is, as described before, not wanted due to possible side effects. Nevertheless, for the case of erroneous input deletion could be required, either because the description is erroneous, or an item, relation or knowledge is false. One possibility is by changing these errors by hand in the XML-representation of the database. This can be error prone as the consistency of the lexicon requires the existence of identifiers on different levels. Therefore only lexical items are allowed to be physically deleted that are not referred to. Consequently, the first step in deleting an item is to delete the relation and knowledge referring to it.

If a lexical item is to be deleted physically, the procedure requires first to delete the knowledge that points to this item. As knowledge points to relations or items only, no side effects are possible as the reference is unique. The relations pointing to the item to be deleted physically are deleted then by first deleting the relations pointing to this item. As afterwards no other part of the lexicon points to the item, it can safely be removed from the database and the storage can be freed.

As relations can only be referred to from the knowledge section, and knowledge can only be referred to from other knowledge items or relations for both of them the deletion is similar: A query has to be performed for items which refer to the item to be deleted, and these knowledge items have to be deleted first.

3.6 Gain by the Lexicon Graph implementation

The Lexicon Graph implementation in the previously described manner involves the idea that the lexicon microstructure is effectively a graph structure, with the set of lexicon items being the nodes and the relation to other lexicon items being the edges of the graph. The source and target sets are identical, i.e. the set of lexical items.

The following relations exist for all lexical items:

- Each item is related to itself, which is an identity relation
- For each relation there is an inverse relation, i.e. it is possible for a relation r_{l_1,l_2} with source l_1 and target l_2 to state that there is a relation $r^{-1}_{l_1,l_2} = r_{l_2,l_1}$ with source l_2 and target l_1.
- The relation is not necessarily injective, i.e. an lexical item in the target set, which is the same set as the source set, can have more than one element in the source set. Not even within one lexical dimension, the relations are injective, because $1 : n$ relations are possible. An example of these are the relations mapping parts of speech onto lemmas, where the class `verb` for example is related to all verb lemmas.
- The relation is surjective, which is trivial because of the identity relation.

The relations are not the same but define a multidimensional space, each lexical relation defining a dimension, resulting, in a hypergraph.

Figure 3.12 shows a three dimensional lexicon space with the axis $orth$ for *orthography*, $pron$ for *pronunciation in SAMPA notation* and POS for *part of speech*; a sample of various values is included, the orthography and pronunciation originating in three different lexicons, i.e. the Verbmobil lexicon (the German word `anwesend` with its pronunciation, see Section 2.3.16), the EAGLES Termbank from MARTIF format (the technical term `linear discounting` with its pronunciation, see Section 2.3.14) and the Oxford Advanced Learners Dictionary (the English word `unit` with its pronunciation, see Section 2.3.2). The data format of the values is not given outside of the dimensions. It shows the position of two lexicon entries in the 3 dimensional lexicon space but also some positions in a two dimensional lexicon space.

By selecting a start axis the lexicon macrostructure is defined, by the selection of the dimensions and the order of them the microstructure is included in the lexicon space.

As the three dimensional lexicon space can only partially be represented in a two dimensional medium, the same holds true for a multidimensional lexicon space. In such a space, the types of the lexicon items in the LG approach

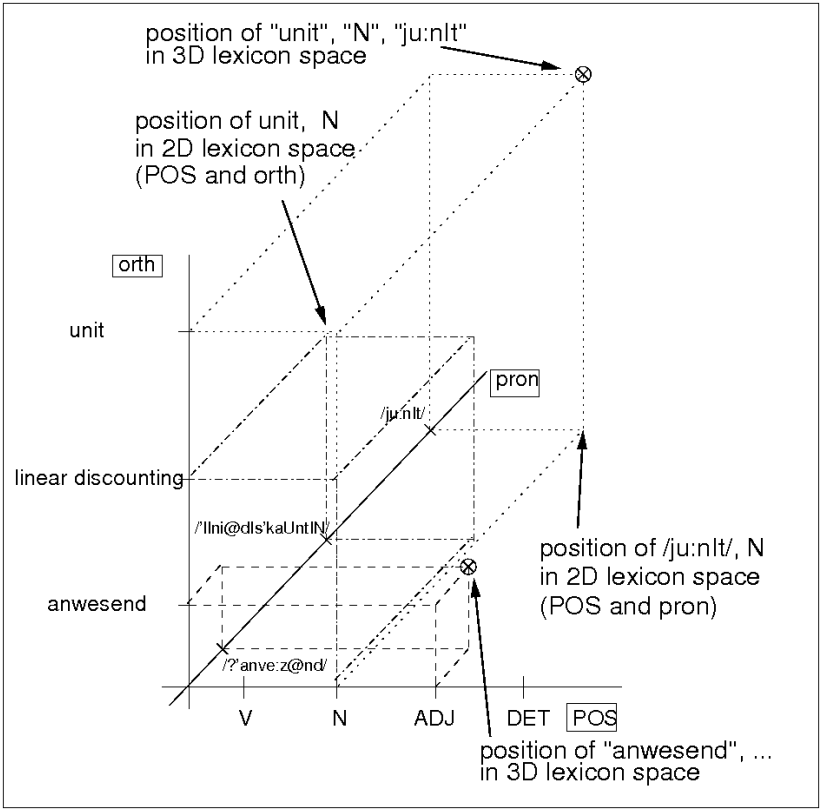

Fig. 3.12. Three dimensional lexicon space covering `part of speech (POS)`, `orthography (orth)` and `pronunciation (pron)`

define the axes, with the lexicon items the segments of the axis, the relation types define a two dimensional lexicon. The edges are typed to indicate the dimensions, for simplicity reasons the types of the lexical items is left out. Again, the graph is too complex to be printed here.

After the discussion of the microstructure and macrostructure in the Lexicon Graph paradigm leaves the problem where the lexicon mesostructure can be found in this lexicon hyperspace. The dimensions represent the microstructure and the macrostructure is the axis or dimension for starting the exploration of the hyperspace. The answer is simple: The mesostructure defines subspaces of the hyperspace. This means that the classification of hierarchies and relations between dimensions based on some similarity relations —

which are the mesostructure — enables a better handling of an n-dimensional space by creating subspaces with less dimensions, according to the number of dimensions used in the mesostructure description. For example, a mesostructure clustering *orthography* and *pronunciation* under *surface structure*, while other dimensions such as *semantics* and *morphology* are taken into consideration, the subspace *surface structure* with its dimension can be omitted. Of course an investigation of *surface structures*, for example, to explore the relation of a graphemic and phonemic level, relies on the subspace definition as well.

3.7 Summary of the Lexicon Graph approach

In this chapter a graph view of the lexicon was introduced as a formalism that allows the combination of different lexicons in a generic representation. It provides a solution to the problems left open in other lexicon representations and allow a portable representation and data format and takes advantage of a modern lexicographic infrastructure.

The lexicon microstructure is used to define different relations between lexical items, the macrostructure is used to define a way to access the lexicon graph. The mesostructure is the definition of subgraphs in the lexicon graph.

An implementation of this approach was described including procedures for merging different kinds of lexicons. Procedures for querying were used to illustrate the content of the lexicon, using a graph representation with labeled edges to represent the lexicon hyperspace in a two dimensional medium.

The evaluation of this model still has to be performed, especially in the areas of functionality, expressiveness and simplicity. A more fundamental problem still left out so far is the relation of the lexicon to other resources, especially corpora. This refers to the problem of the origin of lexical information, for which a formal model exists. The evaluation will be discussed in the next chapter, the connection of lexicon and corpus in the following part.

4

Testing and evaluating the Lexicon Graph Model

Based on the requirements for modern lexicons the Lexicon Graph Model was created to describe existing lexicons. This model needs needs to be evaluated for

- Functionality, which covers the areas of available tools and possible access
- Expressiveness, which relates to the problem of interchangeability into other formats without data loss
- Simplicity, which is connected to the quest for as simple a representation as possible to enable maintenance and use.

This section approaches the evaluation procedures and actual tests conducted with the implementation.

4.1 Testing methods

For the evaluation of traditional published lexicons (such as Crowther (1995) Messinger (1978), Matkins (1994) or Terrell et al. (1995)) users usually are informed about

- Number of headwords or lexemes, sometimes called *coverage*. This is a problematic classification as the part of the language covered by one lexicon is not only determined by the number of headwords or lexemes but also by the sentences used in definitions, explanations and examples, inclusion of types of lexical information, etc.

- Number of figures, which might be relevant, for example, for dictionaries of concrete objects and persons, but which does not indicate the usefulness for systems
- Number of pages with extra information, such as rules or language background, which is relevant especially in learner contexts.

These tests are statistical descriptions of the lexicon structure which use, for example, the number of headwords instead of number of words, the later would be a content description. In the context of a lexicon database these statistical functions can easily supplied by standard statistical procedures in the system. Statistical evaluations can be called *quantitative testing*. A *qualitative testing* is not involved, though.

For other systems other ways of evaluation have been described that include a qualitative testing besides some specific quantitative methods. In speech synthesis, a detailed description of system tests are available, for example, in Gibbon et al. (1997b) (Chapter 12, *Assessment of synthesis systems*). Testing methods for synthesis systems are based on

- Individual component testing, such as word recognition rate, grapheme-phoneme conversion correctness, stress assignment, morphological and syntactic analysis. These components are tested in a *glass box approach*, i.e. every component is evaluated for its functionality.
- System as a whole testing, such as a usability test where a user is trying to communicate with a system and the number of unsuccessful interaction is measured. This is done in a *black box approach*, i.e. the system is evaluated just on the input and output side without looking at the components.
- Performance testing, where the time required for processing is evaluated, which is relevant for real time applications. This is performed either for each component in a *glass box approach* or for the whole system in a *black box approach*.

Another issue is a *usability test*, i.e. the test where users have to work with a tool to be evaluated and where issues can be assessed such as time for accomplishing the task, ergonomy, etc., which in themselves can include qualitative and quantitative methods.

The question with these test methods is if they can be transferred onto lexicon models and systems and if so than how they can be applied. As the the test methods mentioned before are created for system tests it has to be discussed in which way the LG paradigm relates to a system. It does so in several ways:

- The LG approach enables the systematic connection of corpus and lexicon in terms of corpus based lexicon generation and corpus access. The connection was already discussed in Section 2.1.4 and an implementation of a direct relation of corpus and lexicon is given later in Section 7.2
- The representation of lexicons in a graph based formalism allows the use of the lexicon in applications, therefore these systems are directly connected to the model
- The data structures in themselves provide a pattern that implies an interpretation, for example, the *linking* of lexical properties to external knowledge. This can be interpreted as interfaces between different application domains.

In other words: the approaches of testing systems can not only be applied to the Lexicon Graph but really has to be applied to the model. This does not mean that another area of the model is sufficiently taken into consideration, i.e. the complexity of such a system on the implementation side. As the size of the lexicon can be large with large numbers of lexical items and relations, the number of nodes in the document tree in an XML implementation becomes huge, resulting in a rather large search space. The optimization of a search strategy on these data structures and on the storage is again beyond the scope of this work and hence not discussed here.

4.1.1 Synthesis based testing

Synthesis is one possibility for testing the expressive power of a model, i.e. by re-construction. For the evaluation of a model for a lexicon an existing lexicon is used and transformed into the format that has the expressiveness of the new model. The new model is then transformed back into the original format. Ideally the result should be identical, but there are good and valid reasons why this might not be the case:

- Ambiguity in expressing constructs: in one or more involved formalism there is a way of expressing the same thing in two different ways. As this kind of structural ambiguity cannot be resolved without changing the original structure, there might be a deviation from source and target. One area where this might cause problems is the case of polysemy and homonymy, which are rather hard to differentiate. The Lexicon Graph model does not allow for ambiguity in the entries, but the interpretation in a given context can be ambiguous.
- Underspecification in the source format: sometimes there are certain structures which are undefined or underspecified. In NLP systems this

is an issue treated in the semantic analysis, but in respect of formats this is not wanted. One example is the absence of metadata due to inconsistencies in the work of a data providing person.

A synthesis based approach on model testing is based on the assumption that a structure that is used as the primary data can be reproduced using the model as an input.

4.1.2 Other evaluation types

Other types of evaluation of a model are discussed here but not into full detail. For example, the statistics of the Lexicon Graph cannot be compared to other lexicons as the test set is comparatively small. A test of ergonomy and usability does not apply in this context as well as these are basically questions that are related to a user interface, especially to *graphical user interfaces* (GUI).

It is possible to argue that a performance test could be accomplished in the way that the number of steps needed for a specific procedure can be counted, for example, to transform a given structure into another or to query a structure for special information. Although this sort of tests might give a slight idea of the complexity involved, it is still not suitable for predicting performance for several reasons.

Performance depends not only on the number of operations but on the way these operations are implemented. A good example of the problem involved for this is the computer hardware benchmarks for different CPUs. Some processors are more optimized for floating point, others for string processing. This continues to the software implemented for processing hierarchical or flat structures, or document or data driven structures such as the distinction of *XSLT* and *XQuery* for processing document centered XML or data centered XML (see Section 5.1). Kay (2004b)(pp. 170-183) discusses optimization techniques for XML processing, which is structurally relevant for the representation of a Lexicon Graph as well, if represented in XML or another syntax, where he mentions that especially building an internal representation of the tree structure can be handled differently in both approaches, one requiring on the fly generation of the tree structure, while the other can use a precompiled version for performing typical tasks.

The size of the data to be processed does not correspond to the structure. The complexity of processing data of a certain structure might not scale linearly with the number of data items. Trippel et al. (2003) provide an estimator for the complexity of the lexicon generation using a time-calculus approach. The possible power-set of relations does not allow linear scaling. As the Lexicon Graph allows a similar connection of all lexical units (the node set), using,

for example, a *is not related to*-relation, similar effects can be predicted but in practice these are rather unlikely to appear.

4.2 Evaluation of the Lexicon Graph Model

The Lexicon Graph Model was evaluated by transforming a number of lexicons into the format, i.e. a part of the Verbmobil lexicon, a termbase represented in MARTIF, some entries from the Oxford Advanced learners dictionary and a part of a parser based lexicon and merging these into one consistent lexicon. The later was based on the BOMP lexicon (Portele et al. (1995)) was simpler than the Verbmobil lexicon, and therefore not discussed above in detail.

Another evaluation strategy is based on the resynthesis of a lexicon entry in a different format starting from the Lexicon Graph. In Section 3.4.3 an entry of the Oxford Advanced Learners Dictionary was represented in the Lexicon Graph format. As the structure of that lexicon entry originally consists of embedded structures, besides flat structures as the Verbmobil lexicon, this lexicon entry is a sufficient test case for testing the appropriateness of the lexicon format. Based on the Lexicon Graph implementation shown in Section 3.4.3 the goal is to represent the same information as in the original using a query language to the graph. The query language of choice for XML data is in this case XQuery (see Boag et al. (2003)). This allows a flexible way of navigating the XML tree, interpreting the element names, content and attributes.

The following XQuery produces the lexicon entry for the OAD. The *examples* are embedded in this query in the definition context, just as in the print version.

```
let $document := doc("oad_sample_implement.xml")
for $orthography in $document/LG/lexitems/lexitem
where $orthography/@type="orth"
return
<entry>
 <orth>{$orthography/text()}</orth>
  {
  for $relation in $document/LG/relations/relation
  where $relation/source/@idref=$orthography/@lexid
  return
   for $relationtype in $relation
   where $relationtype/@type="has_pos"
```

```
   return
    for $posrelated in $document/LG/lexitems/lexitem
    where $relationtype/target/@idref =
          $posrelated/@lexid
    return <pos>{$posrelated/text()}</pos>
  }
  {
  for $relation in $document/LG/relations/relation
  where $relation/source/@idref=$orthography/@lexid
  return
   for $relationtype in $relation
   where $relationtype/@type="has_pron"
   return
    for $pronrelated in $document/LG/lexitems/lexitem
    where $relationtype/target/@idref =
          $pronrelated/@lexid
    return <pron>{$pronrelated/text()}</pron>
  }
  {
  for $relation in $document/LG/relations/relation
  where $relation/source/@idref=$orthography/@lexid
  return
   for $relationtype in $relation
   where $relationtype/@type="has_def"
   return
    for $defrelated in $document/LG/lexitems/lexitem
    where $relationtype/target/@idref =
          $defrelated/@lexid
    return
     <def>{$defrelated/text()}
     {
     for $definition in $document/LG/relations/relation
     where $definition/source/@idref=$defrelated/@lexid
           and $definition/@type="has_example"
     return
      for $exampleitem in $document/LG/lexitems/lexitem
      where $exampleitem/@lexid =
            $definition/target/@idref
      return <example>{$exampleitem/text()}</example>
     }
     </def>
}
</entry>
```

The output of this lexicon query to the lexicon graph is shown below, some line-breaks are edited for legibility. It is obvious that the output, besides the embedding of examples and order of definitions, is the same as the one shown in Section 3.4.3.

```
<entry>
  <orth>unit</orth>
  <pos>n</pos>
  <pron>/ju:nIt/ </pron>
  <def>a fixed amount or number used as a standard of
       measurement
    <example>an unit of currency</example>
    <example>The metre is a unit of length. </example>
    <example>a bill for fifty units of
         electricity.</example>
  </def>
  <def>a single thin, person or group that is complete
    in itself,  although it can be part ofsth larger
   <example>a family unit</example>
   <example>course book with twenty units.</example>
  </def>
</entry>
```

A transformation by a formatting stylesheet (for example implemented in XSLT, see XSLT (1999)) results in the original print rendering. This is accomplished by interpreting `orth` as something to be printed in bold and `pron` to be rendered in italics and the definitions being enumerated, etc. The resynthesis of the original structure is hence successful.

The input for this test was based on the lexicon graph that was created from the original lexicon entry. The question is, if the information is maintained even if larger lexicon structures are included. To test this, the same query was applied to a lexicon graph consisting of a merged lexicon graph composed of the sample MARTIF lexicon graph, the sample of the Oxford Advanced Learners Dictionary and the sample of the Verbmobil lexicon, all mentioned in the previous section. The result of running the query was identical to the one shown above.

The result of operating queries on a merged lexicon graph shows that the lexicon graph can be used as a lexicon interchange format, as it can be transformed back into the source formats. This is a crucial in the context of portability, because it allows the transformation into a data format that is needed, for example, by specialized tools.

This work was not concerned with the creation of a lexicon systems export and import features but with the concepts that are used to allow these. However, the data was transformed into the XML representation of the Verbmobil lexicon for a small portion of the data in initial tests. Other formats were not tested.

Flat microstructures can be generated using the LG paradigm, storing in a relational database system or in other vector based formats is possible. Using a functional programming language such as XSLT for the transformation these flat structures can be enriched by hierarchies.

A conceptual evaluation was produced based on the requirements to modern lexicography as discussed in Section 2.1.6. The LG model conceptually answers especially the modeling of a lexicon according to arbitrary linguistic units and structures and allows the inclusion of any type of multimedia as objects in the lexicon structure. The other requirements are part of the individual implementation, not of the concept. For example, there is a concept for the extension, and the lexicographic resources can be included in the lexicon graph to provide for a transparent relation of source and description.

4.3 Summary

This section discussed the functionality, expressiveness and simplicity of the Lexicon Graph model and assessed the model according to assessment methods introduced. The result was that the Lexicon Graph Model is indeed a formalism that is suitable for all requirements and can easily be implemented to match existing lexicon structures. The relation to corpora and the associated data models still needs to be discussed, which will be done in the next part.

Part II

Practical considerations in corpus based lexicon generation

Practical considerations in corpus based lexicon generation

Based on the graph structure of the lexicon described with the Lexicon Graph Model, this part adds to that for more practical considerations. As modern lexicon developments are virtually all corpus based the question arises: How can a lexicon be extracted from existing corpora and annotations? This question is especially striking in the context of multimodal material, i.e. annotations that are related to a lexicon, possibly in a score instead of a linear text.

For the purpose of lexicon extraction from corpora the following questions are essential and will be answered:

1. What is a corpus? This covers the question of creating a corpus, representing the corpus, the structure of corpora and how is this structure related to the lexicon.
2. How is the content of a corpus described? That means, what information is necessary to allow getting ideas on the quality of a corpus and the content of a corpus. This covers the area of metadata descriptions for corpora, for which a number of recommendations exist, though these have some problems. The structure of the metadata is also described.
3. What microstructure data categories can be found in corpora that can be used for lexicon extraction? This is another question related to the metadata of corpora.
4. What kind of lexicons can be easily produced from a corpus? The most obvious ones will be described, which are textual concordances. These concordances are extended to multi-tier scores for multimodal annotations.
5. What higher order lexicons can be produced from a corpus? And how does this relate to the Lexicon Graph model? This points to the question, how to use multi-tier concordancing for generalizations, starting from simple word lists, and looking at temporal calculus based extraction of generalizations.

Finally some possible extensions, implications and ideas for further research will be addressed.

5

Relation of annotation structure and lexicon structure

Modern lexicons are based on corpora. As there is a procedural connection between corpora and lexicons a closer look is also taken at the formal relation of lexicon and annotation, starting with an analysis and description of existing annotation formats used in corpora. This involves the questions on corpus creation, structure of annotation and the comparison of lexicon and annotation structures. It shows that the structures of annotations and lexicons can be seen as strongly related.

For the written language domain Kilgarriff (2001) (Section 7) discusses a similarity in annotating a corpus and authoring a lexicon entry. In his context the annotation of a corpus is deciding which meaning from a sense enumeration lexicon is to be assigned to a word in a corpus and authoring a lexicon entry is the classification of distinct senses. These processes are similar in the work flow but they are also similar in structure and content. For the multimodal domain corpus based lexicon generation means that lexicons are based on detailed annotations of speech signals. Whatever domain is under investigation: corpus based lexicography presupposes the existence of an annotated corpus.

Gibbon (2005) creates a continuum between corpora and lexicons which is reproduced in Figure 5.1. Although the illustration implies the different labeled layers as part of a lexicon typology being discrete, it can be argued that it really is a fuzzy transition from one level to the other. This is also true for his implied distinction of lexicon and corpus, which can be seen as two sides of a similar object. The simplest structure is a primary corpus, such as a recording or manuscript, together with a transcription or aligned annotation this corpus is called a secondary corpus. With classificatory markup such a corpus becomes a tertiary corpus. A lexicon based on at least a secondary corpus, such

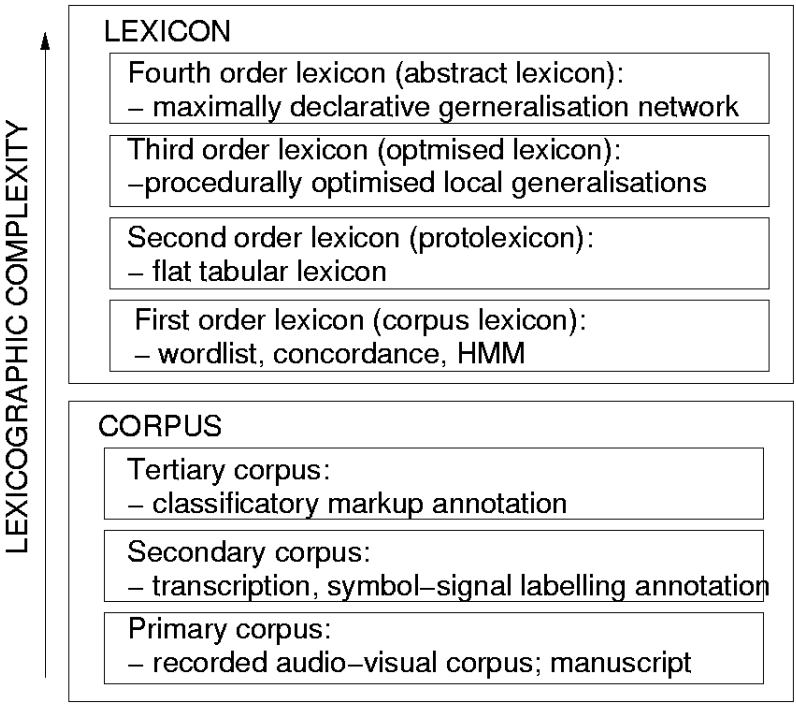

Fig. 5.1. Continuum of corpus and lexicon according to Gibbon (2005)

as a wordlist or concordance is a called a *first order* lexicon. The recording in this form already implies a classification, for example by ordering identical items together, so that it is also related to tertiary corpora. A *second order* lexicon is a lexicon in a table form, as it is frequently used by applications. In the case of such a lexicon being procedurally optimized and containing some generalizations the lexicon is a *third order* lexicon. Most print dictionaries are of this kind, using local abbreviations as generalizations. They are not maximally generalized in a declarative network such as the Lexicon Graph, hence they are not *fourth order* lexicons. The transition between these individual lexicons and the corpora is part of Chapter 8, before coming to this chapter on generating lexicons some prerequisites need to be discussed.

In the previous chapters the lexicon was described and defined as a lexicon graph, i.e. the lexicon itself is graph structured. If lexicon and corpus are so closely related in the work flow and the most abstract level of the lexicon is based on a graph structure than the question is in which way the structure

is similar to the corpus. As the corpus is used for the creation of lexicons, another question is how to create a corpus. These corpus related questions are discussed in this chapter.

5.1 Annotation structures

The importance of having a corpus has been mentioned before. The question remained, what structure the corpora have. Corpora and annotations are in this context used almost synonymously thought they vary. A corpus is a data source which is annotated, i.e. contains additional information from a research domain. Corpora for spoken language hence contain at least a signal and a related annotation such as a time aligned transcript. Time alignment does not imply a close time alignment, but the idea that the transcript is produced based on a specific part of a signal, for example, the whole signal or a tiny interval. As the structure of the source data is very general, the annotation structures are used to relate the corpus to the lexicon. Two different classes of annotation structures are distinguished initially, which corresponds to a well known classification of XML formats (see for example Wittenbrink and Köhler (2003), p. 23; Kay (2004b), p. 155):

Document centered annotation formats: the digitalized corpus serves as a transport medium that is created for describing the structure of a document, which can be rendered for human use by enabled programs. The document structures can as well be used for the transformation into other, possibly also document centered annotation formats. Most existing text-corpora are based on these structures.

Data centered annotation formats: the content of the document is more important than the structure, as the content consists of information for applications. Typically the structures are rather flat and seldom deeply embedded using recursion.

The differentiation of data and document centered annotation formats will be used in the discussion of textual and signal based corpora.

5.1.1 Annotations from signal based corpora

Signal based annotations share a common ground as all of them include references to some kind of signal, which provides a linear order of annotation elements. The reference to the signal itself, is either included as metadata or

included only implicitly by the user identifying the corresponding signal by a file name or storage location.

Descriptions of the annotations and signal is only available inasmuch as the tool requires them. The functionality of the tools is not in the focus in the context of corpus creation for multimodal lexicons, only the annotation formats are being discussed here. The resulting data formats can be classified similar to XML data as previously done into

Document centered annotation formats which resemble a textual structure with paragraphs and possibly with headings.

Data centered annotation formats in which the annotation is clearly structured into the units according to the level of annotation.

5.1.1.1 Document centered annotation formats

In document centered annotation formats, for example as used by the *Text Encoding Initiative* (TEI) or by the tool *Transcriber* (Barras et al. (2003)), the reference to the linear order given by the signal is included using so called *milestones* (see Barnard et al. (1995)), which are pointers to a specific point on the linear time scale, outside of the TEI community called *time-stamps*. In XML, the milestones in an annotation are typically defined by empty elements that can occur at any position of a document. Intervals are defined implicitly as the region between two milestones or pointers to the time line.

Document centered annotations typically do not use more than one annotation of a signal. This annotation however can be structured hierarchically, allowing for embedded structures.

The data model of Transcriber (Barras et al. (2003)) allows the alignment of transcriptions to a signal. Created for the annotation of radio broadcasts it allows for different speakers and topics, but everything originally based on orthographic transcriptions. Consequently multiple linguistic levels to be coded on more than one tier are not included.

The syntax of the data format is based on XML, as a document centered format timestamps are included with pointers, which are represented as empty elements with an attribute containing the time in milliseconds.

The TEI (see Sperberg-McQueen and Burnard (2001b), Section 11) provide another structure for the annotation of speech, though it does provide a reference to the time only by pointers to an external time line (see Sperberg-McQueen and Burnard (2001b), Section 14). Different annotation levels are not intended, only word level annotation are presented, though parallel annotations are alignable using a linking mechanism from the time line, based on milestones in the annotation.

5.1.1.2 Data centered annotation formats

Data centered annotation formats define explicitly or implicitly intervals on the time line, i.e. they either represent start and end of the interval or one or the other is inferred from the preceding or following segment respectively. Some annotations also define a point in time to have a certain property, which can also be interpreted as an interval between two such points where the interval starts or finishes with a certain property. Embedded text-like structures such as paragraphs, headings, and sections are represented in data centered annotations in different ways, e.g. by creating different annotations for each level of annotation. Subject classification is part of the metadata which can be included in some formats.

Data centered annotation formats can easily include more than one annotation level by adding new tiers referring to the same time line. These formats are used by many tools, including software and data formats from the area of phonetics such as *Praat* (Boersma and Weenink (2004)), *Wavesurfer*(Sjölander and Beskow (2004, 2000)), ESPS waves+ (Entropic Research Laboratory (1998)), and the *TASX-annotator* (see Milde and Gut (2002)), *ELAN* (see ELAN (2004) in the bibliography) *ANVIL* (Kipp (2003)), and *EXMARaLDA* (Schmidt (2004)), which itself uses the data format of the *Annotation Graph Toolkit* (Bird et al. (2004)).

ESPS and Wavesurfer

The classic program for phonetic analysis is ESPS waves + or Xwaves. As a tool created for the most detailed analysis of speech signals, allowing for detailed annotations, the data structure allows the inclusion of arbitrarily many levels of annotations in new tiers, resulting in a multi-tier score view. In the data structure, all tiers are maintained separately connected only by the common time line.

The single segments are represented in a simple attribute-value pair using ASCII encoding, containing only the end time, the start time of the interval is inferred from either the preceding segment or if it is the first interval the start of the signal file.

Wavesurfer (Sjölander and Beskow (2004)) has a very similar file structure except that it uses explicit time intervals, providing a start and end time as a character delimited table of triples. It was developed to replace Xwaves at some time because Xwaves is no longer maintained.

Praat

Praat is a software for phonetic analysis, which can also be used for annotation with its own data format. This data format represents a flat hierarchy where all segments are part of a tier and the segments consist of a triple of start time, end time and label. Additionally an enumeration of the segments is available. A special kind of tier is available for non interval segments, where only a point on the time line is given.

The data format of annotations in Praat are based on an ASCII encoding, which can be interpreted by the software itself and worked on with different procedures. Praat does not support arbitrary labels as the data format interprets certain character sequences as the encoding of IPA symbols which are rendered by the Praat software using *TrueType Fonts*, but not based on Unicode tables. Praat is therefore unusable for general annotations.

TASX

The *Time Aligned Signaldata eXchange format* (TASX, originally developed by Milde, Gut and Trippel, see also Appendix 10 for details on obtaining the grammar) is different from the other data formats in the sense that it was developed independently of an application as an interchange format for different applications. A specific goal for this format was the combination of the benefits of *xwaves* and *Praat*, allowing accessing and reuse of the resources in a corpus linguistic context, using existing XML based methods. One of the first applications was a concordance for audio data (Trippel and Gibbon (2002)), where the resources were converted from existing *Praat*-corpora. As the TASX format was developed in a multidisciplinary context with corpora for endangered languages (in the project EGA in the DOBES consortium, see Gibbon et al. (2001)), for second language acquisition (in the project *Learning Prosody*, see Gut (2003)) and for multiple modalities (in the ModeLex project, see Gibbon et al. (2004)), a mechanism for the inclusion of all sorts of metadata was included as well as a strict signal based coding. Later an annotation tool working on the TASX data structure was developed (see Milde and Gut (2002) and TASX-Annotator (2004)).

The TASX format is a data driven approach to annotation formats consisting of a flat tree structure in XML syntax. Different tiers, in TASX-terminology *layer*, contain annotation segments (*events*) which are a representation of a quadruple of start and end-time, identifier — these three represented as XML attributes— and the label, which is the element content. Different recording sessions can be combined into a large collection of sessions, which form the corpus. The metadata structure for the *TASX*-format

consist of a representation of attribute-value structures that can be applied to every level of the annotation tree, i.e. on event, layer, session and corpus level.

The *TASX* format is the most flexible format, as it allows a mapping from all data driven formats, without information loss; the mapping back is not necessarily lossless as the metadata that can be represented in TASX does not have an equivalently strong representation in other formats.

5.1.2 Annotations from text corpora

Besides signal based annotations there are corpora based on written language, for example books, and articles, relying on the orthographic forms of the language and referring to the written modality only.

For corpora for the written modality a classification into document and data centered formats is possible as well as described in Section 5.1:

Document centered annotation formats for the written modality: Examples for document centered annotation formats are for example LaTeX, the *HyperText Markup Language* (HTML), or more complex document types as described by document grammars such as *DocBook* (see Walsh (2003)) or by the TEI (see Sperberg-McQueen and Burnard (2001b)).

Data centered annotation formats for the written modality: A data centered approach to text is for example VoiceXML (Boyer et al. (2000)), which is not document but dialogue centered, providing a structure for modeling human-machine communication. Another application is the processing of multiple annotations of the same textual data (see Witt (2002b), esp. Chapters 3 and 5 and Sasaki et al. (2003)) where by the enumeration of characters the linear order is explicitly given, creating a kind of a virtual time line. This can be seen in direct analogy to the signal based annotations.

As texts in proprietary formats are inadequate for permanent storage of language resources (see Bird and Simons (2002)), these are not taken into account.

LaTeX

A common digital text format for publications is available with LaTeX (Lamport (1994)), which provides a markup for texts to be compiled into a graphical rendering. Originally the data format for the graphical rendering was intended to be independent in the so called *device independent format* (dvi), but

usually compiled into a printable format such as the Portable Document Format (PDF) or PostScript (PS), both developed by the software manufacturer Adobe. As a lot of print and layout relevant information is considered structural information, it can be used for the translation into a structural markup. However, structural markup is not intended and merely a side-effect from the consistent use of layout macros.

For large automatically used textual corpora, the use of print markup as structural markup does not seem to be suitable without prior transformation into a different format.

HTML

The intention underlying the *Hypertext Markup Language* (HTML, see Berners-Lee and Fischetti (1999), p. 86) originally was to allow the efficient distribution of hypertexts by defining a document syntax rather than layout. With the increasing popularity of the World Wide Web, HTML is widely used. The structural markup however, was to some extend used in a different than the intended way, coding layout information. This results in a mixture of *presentation structure* and *document structure*, for example, resulting in a non hierarchical use of headings. The reason for this are specific renderings by different browsers.

For longer and more complex resources than web pages, HTML does not seem to be suitable as HTML does not provide a sufficiently powerful structure. The standard textual blocks for example are paragraphs, headings are available for 6 different levels. Structures such as chapters and sections can not be modeled sufficiently well. For linguistic corpora containing other levels of annotation, besides the paragraph level, no sufficient structures are provided.

TEI and DocBook

The TEI and DocBook conventions have been provided for the use with larger documents such as technical manuals, books, etc. Both standards in fact provide a whole family of possibilities which share common features.

The TEI provides structures for literary texts such as prose, poems and drama, but also for dictionaries, terminological databases and feature structures besides the previously mentioned *speech annotation* schema.

DocBook as an alternative was basically created for the encoding of technical documentation. DocBook and TEI share the characteristic of providing structural markup for different text genres, providing descriptions of the most

complex documents as a whole. Nevertheless, DocBook and TEI do not provide a consistent linguistic annotation, though with the feature structure description (see Sperberg-McQueen and Burnard (2001b), Section 16) the TEI provides a way for this particular kind of hierarchical annotation.

A linguistically motivated annotation scheme was provided by the *Expert Advisory Group of Language Engineering Standards* (EAGLES). The EAGLES guidelines for syntactic annotation (Leech and Wilson (1996)) provide more details on the syntactic level, using labeled brackets inside of the text in an ASCII format. It does not provide any further way of including other linguistic levels but word level.

XML

To use XML inside of a section on annotation structures might seem to be misplaced as XML provides a syntax for markup rather than a markup. Nevertheless, XML provides a general framework for specific annotation formats and can be adapted to the use of the wanted annotation level. In fact, besides the EAGLES annotation[1] and LaTeX all formats described so far are defined in the XML syntax or have at least an XML version.

The idea is to combine the strengths of the different annotations and formalisms, especially as they describe linguistic information, and create a common structure, represented as a context free grammar. This is not possible for all possible structures as Witt (2002a) (Chapter 3) discusses. He describes that due to overlapping segments of different linguistic levels of annotation the XML formalism is inadequate for this kind of information. Witt points out that the annotation of different linguistic levels based on the same source document in different instances, i.e. multiple copies of the same primary data but with different annotations of the copies, is a suitable procedure to solve this problem. Although a similar method is mentioned by Sperberg-McQueen and Burnard (2001b) (Section 31, especially Subsection 31.5 titled *Multiple Encodings of the Same Information*), they do not discuss the way of actually relating these different, possibly simple or complex markups. Instead Sperberg-McQueen and Burnard mention that relating different markups results in certain problems:

Size of the corpus: a lot of information, namely all available in the primary data is redundant, therefore requiring a lot of space

[1] It is assumed that EAGLES would have been using XML if it had been around at that time. The first version of the XML standard was published in 1996, but as common to new technologies and representations, it took some time to gain publicity.

Data consistency: as the information of the primary data is duplicated for every annotation level, the risk is high that an update is only performed on a limited number of files, esp. not on all files, resulting in an inconsistent data repository.

The relation of the different annotation layers is not introduced by hyperlinks between them but by introducing an implicit or category time line (see Section 7.2) given by the linear order of characters in the primary data. A simple possibility for this is the enumeration of characters of the primary data mentioned before.

Using the so introduced time line, dependency relations between the annotations can be expressed by time logic constraints. The result is an annotation which provides the same primary data multiple times, with additional alignment information, distinguishing it from a pure score annotation as used in signal annotation.

Encoding textual resource in TASX

The virtual time line for textual annotations on multiple layers offers another way of producing the same result, avoiding the problems that occur due to duplication of information. The virtual time line is used just like the time line referring to a signal. The individual annotation levels follow in a score, one tier containing the segmented primary data, for example a text orthography with one letter per segment, and the other tiers containing the annotations of the respective textual level.

The score approach to textual data relates to the use of the *Time Aligned Signal-data eXchange* format introduced in Section 5.1.1.2. Figure 5.2 illustrates the use of the use of TASX for textual data. One tier is reserved as the primary signal tier, replacing the signal data, the other tiers contain one type of annotation each. If the segmentation of the primary tier changes all other tiers have to be modified as well. The requirement for such a change can, for example, be due to initial typing errors.

The need for adjustment of all other tiers to the primary tier is a major drawback of this approach, if the quality of the initial transcription cannot be guaranteed. The reason for this is that the annotation is dependent on this one tier, i.e. the boundaries of a non-primary tiers depend on the primary tier, the interval boundaries are references to other tiers. The dependent definition of tiers could be accomplished by replacing the usual time unit value of the left interval boundary, i.e. the boundary for the start of the segment, by an identifier of another segment on another tier, interpreting the start of the interval as

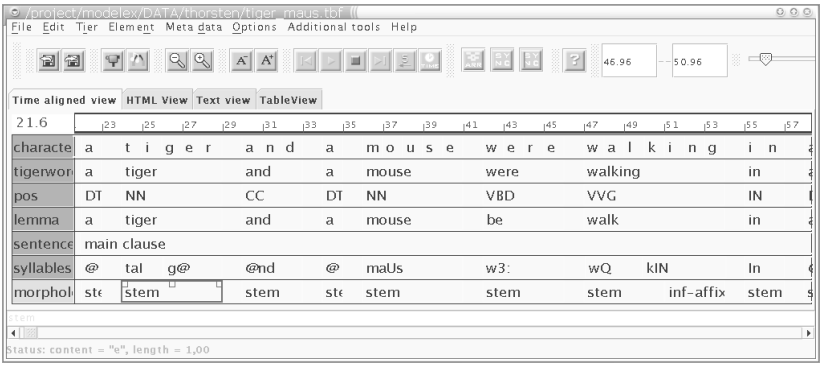

Fig. 5.2. Textual annotation based on the *character enumeration* of the linguistic levels *orthography*, *part of speech*,*lemma*, *clause*, *syllables*, *morpheme classification* for a textual sentence in the TASX-Annotator

start with in time calculus terms (see Section 8.2). The same is possible for the end respectively.

For the TASX-format three different classes of relations between annotation are distinguished.

TASX level 1: all segments are annotated relative to a time line, timestamps are given according to the units of this time line. This level is analogue to standard encoding of signal data.

TASX level 2: timestamps of segments are either given according to a time line in the resp. unit, or relative to another segment on another tier, allowing hierarchical structures.

TASX level 3: by using XML namespaces (defined in Bray et al. (1999)) the annotation format and the content can be separated from each other, allowing for tree-structured content embedded within a time based annotation. Namespaces are used by referring to more than one document grammar for some markup elements by a namespace prefix to the element name.

The definition of tiers which depend on another tier corresponds to TASX level 2. However, the TASX level 2 approach, though conceptually possible and syntactically allowed in the TASX-format, has not been implemented yet for a number of reasons.

First, for a broader segmentation of dependent tiers it is possible to find appropriate events having the required start or end marks, but if the granularity is finer or the boundaries of an annotation segment are at different

positions from the primary tier, there is no way to code this with reference to the primary tier. One might argue that in the case of non existent boundaries on a primary layer the problem could be solved with a reference to the virtual time line, which does not only allow integers as segment boundaries but floats, but this would imply a precision which is not present in category time (see Section 7.2).

Second, the reference method for TASX level 2 allows for moving boundaries to remain adjusted, but segment boundaries that are deleted result in undefined intervals on other tiers. Consequently a requirement of TASX level 2 primary tiers is that a segment that is used as a reference, must not be deleted. This means that a TASX level 2 processor has to test upon deletion for existing references to this segment, or the segments once introduced are illegal to delete at all. In fact the later possibility would be simpler to implement without any additional drawbacks: if a change is required due to an initial mistake in the primary data, the content of the segment can be deleted. Empty segments do not cause any differences when transformed back to a text, if they contain nothing, not even a space character. However, extensive use of this procedure may result in a number of legacy segments, which might be used by automatic, time calculus based processors, as these do not take advantage of content but of the presence of intervals. To solve the problem of indefiniteness when deleting segments, the primary tier has to remain fixed in terms of existence of segments with their content and boundaries, which is only possible if the quality of this tier can be guaranteed. A real solution to this problem still needs to be found and implemented.

Third, it might seem to be problematic to insert new segments into the primary tier. The reason for the problem of insertion is that the adjustment of a segment to a segment of another tier does not imply what happens if at the very boundary a different segment is inserted, i.e. if the relative boundary should remain adjusted to the position on the original relative time line or if the relative boundary should remain adjusted to the original segment. In the latter case the relative segment does not need to be changed, in the first case, either the identification of the primary data segments need to be changed, which is contrary to the idea of uniqueness and permanence of identifiers for a segment, or the boundary value of the relative segment. If the boundary value of the related segment is chosen, the value of the relative segment is in fact a mediated link to the time line, i.e. the boundary is not left as the identifier but is interpreted as having the same — numeric — value as the reference segment; the processor might replace the value by the time line value internally. For rendering in an application the replacement is a standard procedure. The problem of inserting new elements on the primary tier underlines

the need for a permanent segmentation, which is again only possible if the quality can be guaranteed.

The annotation of primary identical data in a score instead of using TASX is not free of all problems but using a number of copies of the same data and annotating these copies has a major drawback in the consistency and size of the corpus. Both representations depend highly on the persistency and quality of the primary data. TASX level 2, introducing relative tiers, does not necessarily provide additional possibilities for text based annotations as the virtual time line introduced, for example, by the enumeration of characters, is equivalent to the identification of relative start and end values of segments on the primary tier. In this case TASX level 1 is sufficient.

5.1.3 General Annotation: Annotation graph

Annotations for both signal and text can be represented in a score, which shows that they are structurally similar. Bird and Liberman (2001) provide a general formalism by describing annotations as graphs, given by a *time line*, nodes and labels. They do not specify if the time line is a category or absolute time line, though the nodes correspond to time stamps. The *labels* are the content of the annotations. Annotation graphs have a number of formal properties. For example, annotation graphs do not necessarily need to be connected, though every node needs to have at least one connection. And due to the nature of time the annotation graph is directed and acyclic.

It is obvious that the TASX format is a binding for such an annotation graph as illustrated by Figure 5.3. The timestamps are explicitly defined and refer to the time line, while the element content is equivalent to the labels of the *Annotation Graph Model* (AGM).

The representation of tree structures in annotation graphs as they are frequently found in morpho-syntactic annotations (see for example Calzolari and McNaught (1996)), does not contradict the use of the AGM. In fact, there is always a way of transforming an annotation tree into a score. The reason for this is that the annotation tree always has a finite number of elements, hence a finite number of annotation levels. The worst case would be to define one tier per element, but usually the elements on the some hierarchies can be represented on one tier. Consequently the transformation of a tree into the score format is not necessarily unique, nevertheless the existence of this mapping allows the use of a general annotation graph as a model.

To allow for hierarchies of hierarchical trees in annotations to be kept in the original format, a relative tier definition is needed as described for TASX level 2. This is consistent with the AGM as well, as this is equivalent

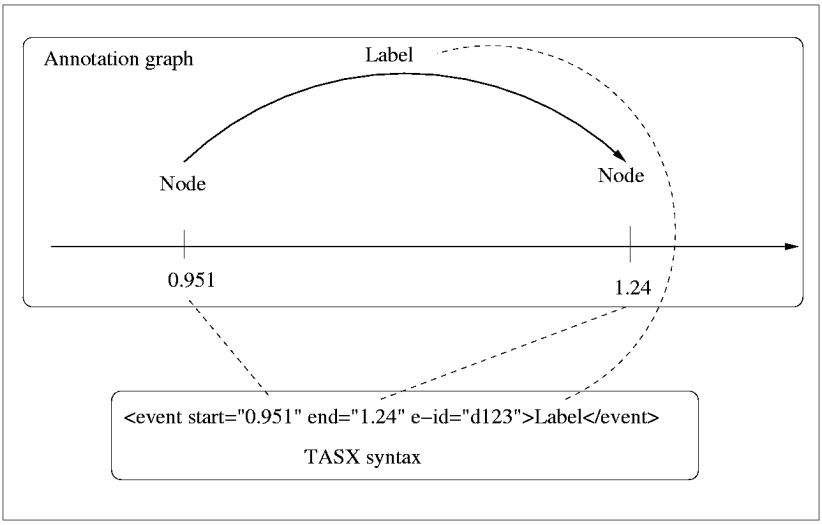

Fig. 5.3. Relation of TASX syntax and Annotation Graphs Model (AGM)

of defining more than one path from one node to another using intermediate nodes. The paths, i.e. the sequence of edges from one node to another, are formally independent from each other. Hence graph structures can be used which allows for a formal structure to compare annotation structures to a lexicon structure.

One problem remains with the AGM, which is the assumed independence of different arcs and paths from one node to the other in the case that they are not defined in relation to each other as in TASX level 2. Even if this relation is not explicitly defined as subgraphs over a restricted node-set it is possible to define hierarchies of different annotation tiers. One example would be a phrase annotation vs. a syllable annotation, that clearly has a relation but the paths are not the same, though they may share some common nodes. The definition of hierarchies of annotation tiers can be done by referring to metadata descriptions of the individual tier, which links an annotation to a position in a hierarchy of annotations for example using an ontology such as GOLD (Farrar and Langendoen (2003)).

5.1.4 Lexical information included in corpora

After analyzing the different structures of corpora the question is left open, which kind of lexical information is contained in them. It seems to be ap-

propriate to differentiate between data and document oriented corpora again. However, the analysis of corpora refers only to structural information that can be inferred from the corpora, not on existing inferable information, which is subject to the concrete corpus.

Lexical information from document oriented corpora is often distinguished into *qualitative* — information about existence and regularities — and *quantitative*, which is a statistical approach. Both share a core part, though the result looks different, in the earlier case a list of words or strings, while in the later some sort of number.

Quantitative information contained in corpora which seem of relevance for lexicons are, for example, *n-grams* or *k-nearest neighbor*, which is resulting in a list of possible contexts, in the case of n or $k = 1$, resp., this is a list of existing units in the corpus, for example a wordlist. If syntactic structures are part of the corpus these can also be relevant for a lexicon, for example, for part of speech classification. All this qualitative information can also be used for statistical analysis, for example, the frequency and distribution of contexts, syntactic structures or simply segments.

In data oriented corpora available in the TASX format the same operations are possible as in document centered annotations, for both it is true that only information can be retrieved on linguistic levels that are part of the corpus. In both no information can be used that is not part of the original annotations, e.g. from a syntactic annotation no phonetic information can be derived. Hence, the retrieval of lexical information is suitable especially for richly annotated corpora as, for example, described by Witt (2002a). Trippel et al. (2003) describe the relations over linguistic levels making extensive use of the interval relations based on time calculus. For example, the relation between different modalities with reference to a corpus can be defined from a phonemic corpus with annotations of syllable structures and corresponding morphological structures as well as prosodic structures in contexts.

5.2 Annotation and lexicon structures in comparison

Recent developments show that lexicons have a strong connection to corpora and vice versa. Lexicons are generated using corpora and corpora are enriched using lexical information. The interrelation is no coincidence but based on similarities in their underlying semantics and syntax. It is argued that the differences between both of them is a result of

- Coverage of corpus and inclusion of exceptions in lexicons on the one hand,

- Different layout especially for human processing
- A fixed precedence relation in annotations based on the directedness of time.

These differences are not necessary and refer only to one specific aspect of the other, presupposing a restriction of a structural variable that is left open in the other context.

5.2.1 Directed graphs as common structure of annotation and lexicon

The formal model of Annotation Graphs has already been introduced in Section 5.1.3 discussing the graph structure of annotations. Annotation graphs are directed and acyclic graphs, due to the directness of the fundamental time line. A lexicon is usually interpreted in terms of trees, for example structured according to ontologies. Another option is to define table structures in the paradigm of relational databases. However, in a more universal approach this is not as easy. The assignment of a property to a lexical item can be interpreted as the annotation of a lexical item in an interval from its start to its end. Figure 5.4 illustrates this interpretation, showing one lexicon entry in a way that each data category is one tier of a score annotation of the headword.

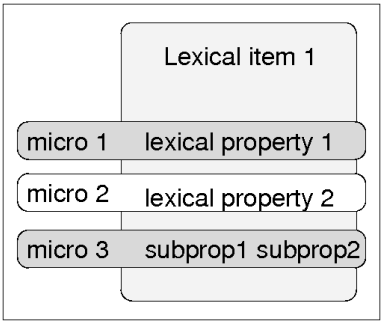

Fig. 5.4. A lexicon entry as an annotation of a lexical item

Interpreting a lexicon entry as a graph allows a wider flexibility, for example, to include units that are smaller than the actual lexical item. An example here is a phonemic transcript where each phoneme can be decomposed according to constituent features. A decomposition of units is also necessary for non letter-based scripts, for example, for Japanese syllable based scripts

a morphological analysis requires different segmentations of an existing lexeme and needs to be included.

Interpreting the lexicon as an annotation solves another problem, namely the interpretation of the common classifiers used in lexicon entries. For annotations it is accepted that each unit appearing in a source is annotated whenever it occurs, hence an annotation layer can include numerous duplicate items. In lexicons however, the reappearance of classifiers such as part of speech identifiers has not been dealt with as they were seen as necessary references to a classification scheme. Nevertheless, there are statistics for the occurrences of lexical categories, e.g. the number of German feminine nouns in a lexicon, etc., which correspond to a distribution of segments in an annotation, for example a wordlist with the accompanying statistic. The possibility of statistics on reappearing classifiers of course refers to the lexicon macrostructure. Figure 5.5 shows a simple lexicon with 4 items with 3 categories for lexical structures. The data categories are again represented as tiers and each lexical item is represented by one segment in the score. Some values appear more than once and some gaps are present. In this case the lexicon microstructure and macrostructure is as well an annotation graph, hence a directed acyclic graph.

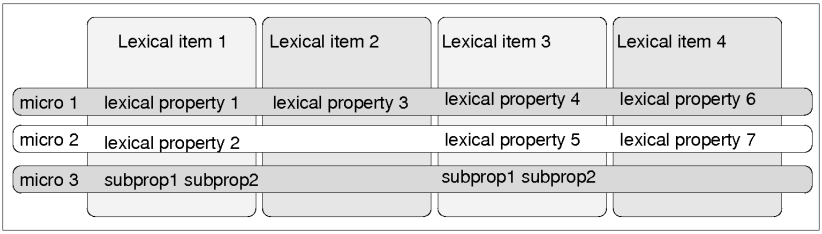

Fig. 5.5. A lexicon with 4 items as a score annotation; the lexicon microstructure can be seen vertically

The previous discussion of similarities of macrostructures and microstructures with annotation graphs in this Section shows a possible diverging structure of annotation and lexicon: the lexicon mesostructure. The mesostructure as previously defined is the connection between the individual lexicon entries, e.g. cross-references, rules and external information as well as a classification of the lexical items. This is in fact the tree structure that has been used before, for example the mapping of a lexical item to a position in a taxonomy. Recent lexical approaches such as WordNet (Fellbaum (1999)) do not restrict

themselves to tree structures here, but allow nets for the representation of semantics. Restricting semantic structures to 'simple' taxonomies with simple tree structures already reveals that these are not linearly structured as annotations are.

The AGM does not cover the discussion of the interrelation of annotation layers. Layers can be represented as related to one another as well, as shown in Section 6.1. The organization and classification of layers using metadata relates to the classification and organization of lexicon entries.

5.2.2 Mapping of Annotations onto Lexicon

The transformation of annotations, being graphs (see Bird and Liberman (2001)), onto lexicons, which are trees, can be accomplished in two ways:

- A rule based approach: the relation of different arcs of the Annotation Graphs are defined or there is only a single arc.
- A logic based approach: the relation of different arcs is unclear and therefore cannot be directly applied. In fact, the relations need to be found first, which is an analytical approach which uses different annotation.

Both approaches are valuable in lexicon synthesis. The former allows the use of rules for enrichment of lexical information and evaluation of annotation against the rules. Enrichment and evaluation can be used both for evaluating the rules and for discovering annotation inconsistencies and problems. The problem discovery allows the acquisition of rules and possibly automatic subclassification, being inferred. The logic based approach is discussed in more detail in Section 8.2 and is left out here. The rule based approach is related to decision trees in artificial intelligence, which is beyond the focus of this work.

5.2.3 Differences of lexicon and annotation

As lexicons and annotations are based on directed graphs, they could be treated as the same structures. Nevertheless, there are differences which need to be taken into account, which are in the ordering, content and treatment of exceptions.

The most prominent difference between lexicon and annotation — besides their presentation structure which will not be discussed here — is the order of appearance of lexical items. In an annotation lexical items are listed in a linear order, based on the syntax of a language and therefore usage. The sequence of items can be described in terms of time, either *absolute time* in

some temporal unit (e.g. milliseconds) or in terms of *category time* (relative precedence). However, if the linear position or time stamp of a lexical item is interpreted as a property of this item, the annotation could be interpreted as a chronological lexicon for a certain linguistic source, such as a text or a signal. *Annotation* is another example of a lexicon macrostructure then. In this case, any annotation could be restructured and used as a lexicon itself.

Another relevant difference does not so much concern the structure but the content of both: While an annotation contains empirical information of what is available, a lexicon is supposed to contain generalized lexical information. A good example is the use of semantic information, which is rather complex to encode. Consequently semantic information is placed in a lexicon and a semantic annotation is done by classifying segments as being related to an entry in a semantic lexicon. The relation of a semantic annotation to a lexicon could be spelled out, but because of the complexity and redundancy — everything that appears more than once had to be annotated with this complex system — this is not done in an annotation, but if it is done, programs enrich the annotation using a lexicon. The enrichment of annotations by complex lexicon entries is true for some linguistic levels, especially if the linguistic levels have a regular dependency, so that one can generate the other using some rules. This is used for example in grapheme-phoneme transducers. Consequently, an annotation does not need to provide an annotator created phonemic transcription, as this can be generated automatically, if necessary and if the rules are known. On the other hand, extrapolating lexical information from a corpus based on logic is only possible if the annotation is large enough covering a variety of cases and covering enough different linguistic levels. This lexical information therefore contains the constrains that are used in transducers, such as grapheme-phoneme transducers, hence the corpora can be used to generate the rules for other contexts then the ones in the corpus, defining hypothetical rules. The real problem exists in creating a corpus which is large enough to cover all required relations. Statistical approaches with probabilistic rules also try to bridge the gap between empirical approaches and rule based based systems, but without the rules being generated from automatically, probabilistic rules could only state how often a rule is applicable.

For a trained linguist with something informally referred to as a ‘good linguistic intuition’ it might be easier to find some constraints than for some computer application. These constraints can be used for lexical generalizations. However, as the number of relations between linguistic levels taken into account is limited to the focus of the linguist, the precision to be expected is higher using a large corpus and a fully specified calculus over all

linguistic levels. In summary it can be stated that even in terms of content a lexicon and an annotation share features, if the corpus is large enough and if the annotation covers sufficiently many annotation levels.

Exceptions are the hard case for lexicon and annotation. In an annotation it can be argued that there is no such thing as an exception as everything is derived from empirical evidence. Therefore singular events are not exceptions but segments with a low probability in an uneven distribution of lexical items. In the tradition of Chomskyan linguistics, the lexicon is on the other hand "a list of 'exceptions,' whatever does not follow from general principles"(Chomsky (1995), p. 235). In other words: a linguistic relation that does not appear often enough to create a rule — an exception — is to be inserted in a lexicon and the general rules belong to a grammar. This does not imply, that every exception should be placed in a lexicon, as it is nevertheless intended to represent generalizations for words. For Chomsky, the origin of these exceptions, however, is left to the reader to specify. This can either be part of a linguists *intuition*, which is the most questionable way of proofing the existence of an exception, or by corpus evidence. In the latter case the consequence is the insertion of non standard, singular relations between annotations on different linguistic levels into a lexicon. As the general relations are already included — the distinction between lexicon and grammar blurred as in HPSG (Sag and Wasow (1999)) — the distinction of annotation and lexicon is diminished again.

5.2.4 Lexicon and annotation in summary

As modern lexicons are created on the basis of corpora, the need to compare corpus and lexicon from a structural point of view was apparent. Graph structures provided reasonable grounds for the comparison of annotations and lexicons. The graph structure is present in standard annotation formats for signal and textual annotations; tree structured references exist for the classification of the different annotation levels. Lexicons on the other hand can be interpreted as graphs as well using the lexical categories classified by an external hierarchy as annotations.

Major differences between lexicon and annotation can only be found in the presentation structure, which has consequences for the access structure, such as order and presentation of items, and coverage. This does not constitute a principle schism but one that is used in practical work.

5.3 Use of corpora

Corpora are a collection of source material with some interpretative information, especially linguistic information, as Leech (1993) states. Modern descriptive linguistics makes extensive use of empirical proofs for rules, constraints and other findings that are taken from source material and their interpretation. However, the use of corpora and the restriction to descriptive linguistics already points to a fundamental problem raised by Chomsky (1957), i.e. linguistic *competence* can hardly be described by the *performance* of language. The description of the constraints of a language according to Chomsky and his followers requires a certain amount of *introspection*. To describe it informally, a linguist who describes the rules of a given language can do so due to his or her intuitions and learned language skills.

It is not intended to discuss Chomsky's criticism of corpus linguistics here, which might be appropriate for many areas of linguistics, such as syntax and semantics. For other areas this is not true, and even after Chomsky in many areas corpus research has been carried out extensively. MacEnery and Wilson (2001) (p. 13), for example, state that empirical findings have always been the major source for new developments in phonetics, giving evidence and tasks for researchers, rather than few ideas which were based on introspection alone. They say (p. 14) that it is the researchers interpretation of observations that results in erroneous theories instead of the corpus evidence.

To evaluate theories that are based on corpus evidence two methods can be used, familiar from speech and signal processing, namely *precision* and *recall*. The former here describes the number of correct application of rules, i.e. how many evidences are processed with the correct rules. For the latter the number of found instances is used to which the rules are applied, i.e. how many instances of a phenomenon are in a source and how many of them are processed. The question of recall relates to the question, which evidence is available in a given corpus at all.

The requirements and design of corpora has direct implications to the lexicon that is based on these corpora. Hence this section discusses implications of size and classes for the use of corpora.

5.3.1 Representativeness and size of corpora

The use of a corpus for the investigation of linguistic phenomena could be said to depend on different factors:

- The genre: words or constructs restricted to a particular subfield of the language occur in a defined subfield only. Corpora that do not relate to

this subfield do not contain these words or constructs. The concept of *genre* in itself is problematic, as the division of resources into subclasses already implies the existence of genres.
- The social and regional variety of a language, which is the focus of sociolinguistic research (see for example Labov (1972)).
- The frequency of a word or construct.

The problem is sometimes described in terms of *sampling* and *representativeness*, which states that not every structure can be recorded in every environment, hence for statistical analysis the representativeness and sampling of a corpus needs to be defined. Mehler (2000) (Chapter 4.2) discusses in detail that the representativeness of a corpus is meant to be a condition but is an illusion and cannot be achieved on formal grounds. However, the size and genre of a corpus is probably the closest one can get to representativeness even if genre is based on intuition and size not a relevant factor if the corpus is too homogeneous.

Zipf (1935) (esp. chapter 2) investigates the distribution of items in corpora and finds out that the frequency of word occurrences in corpora correlates with the distribution. To be more precise, Zipf found out that the most frequently used word in a corpus usually appears double as often as the second most frequently used word, and so on.

From Zipf's law it is possible to deduce the necessary size of a corpus that has a certain number of occurrences of a phenomenon. If r_t is the rank of a phenomenon in a list of phenomena ordered by frequency, f_t is the frequency of this phenomenon, than the product is constant, depending on the size s of a corpus and a language constant l_c, which is about 10 for English words according to Zipf (Zipf (1935), p. 46f). Hence, the size of a corpus to contain n phenomena at least once is

$$s = l_c * r_t * f_t$$

with $f_t = 1$ and $r_t = n$ for the last rank, and $l_c = 10$ for English words this results in

$$s = 10 * 1 * n = 10n$$

which means that for having n phenomena of English words investigated, at least $10n$ words have to be in the corpus under investigation.

The consequence for corpus research is that a corpus to cover a large portion of phenomena needs to be sufficiently large. A sufficiently large corpus is called a *saturated corpus*.

The focus on the size of a corpus indicates in which area corpora are of major use, namely for statistical language research and for lexical driven tasks. The use for statistical and empirical investigation of language seems obvious. As one task of lexicon acquisition is based on the identification of lexical items which can be found in corpora there is also a link to the lexicon. The use of corpora in lexicon development exists, at least since the first developments of the *Oxford English Dictionary* (and its successive editions such as Murray et al. (1970)) where words in lexicons are included based on corpus evidence. However, the use of corpora in lexicons is not necessarily true for all areas, such as use cases that are included in a lexicon, especially for function words which are not always used or taken form corpora. At least usual lexicons mention no source for these (see for example Crowther (1995), Terrell et al. (1995), Matkins (1994), Summers (1992), or encyclopedia such as the German Brockhaus (1996 – 1999)). The connection of lexicons with a concordance allows the reference to empiric findings and contexts.

Another application used in modern lexicography is the automatic collection of lemmas for the inclusion in a lexicon. Based on a part of speech analysis for a given language, a list of lemmas is extracted that can be used for inclusion in a lexicon. Gibbon and Lüngen (2000) describe this procedure based on a spoken language corpus.

5.3.2 Classes of corpora

The application domain for a corpus has consequences for the requirements of a corpus. MacEnery and Wilson (2001) (p. 106 ff) list a number of different user groups, such as corpora for context investigations, occurrences, usage, morphology, semantics, stylistics, socio-linguistics, teaching, history, language varieties, psycholinguistics, cultural studies, language and speech engineering. These fields of corpus usage can be used to subdivide the corpus field into two areas, non-spoken and spoken language corpora.

Non-spoken language corpora

Non-spoken language corpora consist especially of texts that have not been produced for oral presentation, such as books, newspaper articles, etc. A special case are texts that have been produced for oral presentation such as screenplays. Although texts for oral presentation are part of the non-spoken language domain, there is a rendition in the audio-visual modality. Non spoken language corpora are especially used for semantics, stylistics, teaching, and history.

Non-spoken language corpora are used for distribution analysis of words, stylistic research, morpho-syntactic studies, providing empirical evidence (see MacEnery and Wilson (2001)). These corpora can originate from different domains, for example the *Institut für Deutsche Sprache* (IDS, Mannheim, Germany) provides numerous corpora for written German texts, ranging from newspaper texts (e.g. *Berliner Morgenpost* and *Süddeutsche Zeitung*), via political texts (for example the complete works of Marx and Engels) to literary texts (such as some works of Goethe). Sometimes textual corpora are used with additional features such as parallel texts. Sharoff (ming), for example, uses different translations and editions of *Alice's Adventures in Wonderland* as a corpus. This is possible as a text becomes a corpus for linguistic research by applying linguistic methods and questions.

Spoken language corpora

Spoken language corpora contain annotations of spoken signals, based on audio or video recordings. There is some alignment of the annotation with the signal, though this alignment might be given implicitly. Areas which use spoken language corpora are socio-linguistics, language varieties studies, psycholinguistics, speech engineering and phonetics.

A distinction is made between non-spoken language corpora and spoken language corpora to express that some spoken language corpora are written, for example transcribed, instead of non-spoken language corpora that are not created for being spoken.

As everything that is possible with textual corpora, is possible with spoken language corpora, for example, by using the transcript, these are more general, but providing further problems. One problem stated by Leech (1993) is the impossibility of the distinction of representation and interpretation for spoken language corpora, as the textual representation of speech implies the interpretation by an annotator. It can be argued, that the same is true for the reception of textual representation, and even Leech (1993) states that this distinction is artificial, useful only to differentiate between the acknowledged authenticity of the *raw corpus* and the interpretation. For spoken language corpora, the same status is granted to the signal, hence the distinction of a raw corpus and its interpretation is not very useful.

Gibbon et al. (1997b) (Section 3.2) differentiate spoken language corpora and non-spoken language corpora by 8 characteristics, which are from a technical perspective but address also content and ethical differences. A technical problem is the volatility of speech data, which is addressed as a main characteristic of speech, where the signal disappears as soon as it is released.

From the volatility of the signal the need for persistent storage of audio signals arises, covering problems ranging from recording quality (environmental conditions, quality of equipment, etc.) to storage (including data formats, compression, bandwidth, storage space, among others). A major difference, related to the volatility, is the processing of the language, that is oriented toward the actual performance time; the time aspect has consequences for error handling — where in non-spoken language a writer can make sure not to show his or her corrections — and recognition of words and structures, which is given in a written form by categories identified by letters, spaces and typographical symbols; in spoken language utterances, error handling needs additional processing, i.e. unit separation and identification.

As non-spoken language is meant to be read and — though sometimes limited — distributed, the legal status of a written document is relatively clear, as it is meant to be read by an addressee which may include a group of persons. For spoken language utterances the audience is not as clearly defined; the speaker includes only the hearers in his or her judgment of a situation, but does not take into account the audience of a stored utterance. The unclear audience results in the problem of intellectual property rights, including ethical issues, because the consequences of distributed audio signals are even harder to assess than for persistent, written data.

Another difference between non-spoken corpora and spoken language corpora is based on additional information available in spoken language corpora. For example, the presence of prosodic patterns, such as stress and intonation varieties for marking up irony and other phenomena are not present in writing; the lack of additional information form other modalities results in some kind of ambiguity which can be hard so solve.

The differences between spoken and non spoken language corpora are even more obvious for multimodal corpora, where other modalities than audio are included as well, creating similar problems for the other modalities. Hence, the discussion of spoken language corpora applies to multimodal corpora as well.

In the spoken language domain corpora are extensively used not only by linguists but also from neighboring disciplines. When talking about *spoken language corpora* a corpus is meant to provide speech recordings and accompanying aligned annotations, both in computer readable form and well documented, in order to be used for different purposes (see Gibbon et al. (1997b), p. 79). Transcriptions of speech are used in other context as well, such as sociology, psychology or educational science, where dialogues or actions are transcribed and annotated, but the motivation is not focused on language. In spite of existing similarities, non language oriented corpora are excluded from the

present discussion. The corpora taken into consideration are containing some sort of signal data transduced into a required format (e.g. audio file formats), analysis such as Fourier transforms, metadata (see Section 6.2), and aligned annotations on various linguistic levels, such as orthographic level, phonemic level, phonological level, and prosodic level.

For speech technology there are a number of corpora available with only one level of annotation depending on their purpose, while for more detailed linguistic analysis multilevel annotations of speech signals become more and more relevant. Sometimes these are differentiated into *speech corpora*, i.e. single purpose singular tier annotations, and *spoken language corpora* with more than one annotation level.

In the field of *speech technology*, i.e. the interdisciplinary field of engineering, linguistics and computational linguistics where the key interest is the processing of speech, the same uses for corpora exist as for text. In addition to these there are the following areas:

Speech recognition: for speech recognition training corpora are needed, that contain a variety of intended accents, the words that are supposed to be recognizable, all with accompanying, time aligned annotations to allow a machine learning algorithms to align the internal signal information with properties of the signal.

Speech synthesis: in trying to achieve highest quality, modern speech synthesis systems use speech recordings of actual speakers to split the signal and recombine these signal chunks into generated speech. One method to do this is by using a diphone base, for a more detailed description see Dutoit (1997).

Dialogue systems: for man-machine interaction complex models have been used, for example, in so called *Wizard of Oz* experiments, where a trained person pretends to be a system, communicating with a person who does not know of this pretense, via a computer interface. The reactions of the user are studied to allow for a prediction of user behavior.

Speaker recognition/verification: many applications work with user profiles, including security settings, sometimes verified with different graphical user interfaces. A trained system can use signal specific characteristics to identify a speaker or verify their identity. The corpus that is required for a speaker recognition or verification system application consists of a number of different speakers, who are supposed to be recognized, and a control group.

Language recognition: for many classification tasks, for example in automated switchboard applications, language recognition is essential to be

performed before a speech recognition can be successful; for the task of language recognition corpora are needed that contain a variety of different languages, annotated for the language.

5.4 Requirements for corpora

The question of requirements to corpora is the focus of this section. The reason for the general discussion of corpus requirements in the lexicon context manifold, for example the structural closeness of lexicon and corpus results in a possibility of using the requirements for corpora to apply them to lexicons. Another reason is that in a corpus base lexicon generation process the lexicon can only contain what is in the corpus.

A set of requirements for annotations is, for example, defined by Leech (1993) who mentions seven requirements for annotations. These requirements are basically intended for morpho-syntactic annotation but can be applied to other annotations as well.

1. The annotation has to be *removable* without destruction of the original source. This can easily be accomplished if the original is not modified at all, which is rather common for signal based corpora where the signal and the annotations are separated.
2. The annotation should be *extractable* from the original for separate processing and storage, including interlinear format. For signal based corpora this is the same as the previous point. The extension to interlinear annotation which Leech (1993) does not discuss in detail, points to the annotation of primary identical sources or standoff annotation as described by McKelvie and Thompson (1997) and Witt (2002b) (Chapters 3 and 5).
3. Predefined *annotations schemes* such as content models, restricted vocabularies and grammars are used for annotators and corpus users.
4. *Technical information* such as when and by whom an annotation was created have to be recorded. This is stating the need for metadata (see Section 6).
5. The end user needs a *reliability* indication, i.e. how reliable the corpus and the annotation is. As the annotation always includes interpretation, the corpus can only be as good as the interpretation and can only be offered for researchers for their convenience and assumed usefulness, not as a statement of facts about a language.
6. The annotation has to be *'theory neutral'*, though he does not discuss if this is even possible. Theory independence includes the inclusion of agreed annotation schemes, if there are any.

7. The absence of a general, standard annotation scheme results in possible deviating standards, claiming the same authority, i.e. no standard is given absolute authority.

Another procedure for annotation was created by the idea of *Standoff annotation*. Standoff annotation initially was developed for the case of the annotator having read access to a data source but no write access to the data, consequently the annotator cannot add annotations into textual data that are not already in there. Nevertheless, the primary data can be used and referred to and standard annotation symbols (see Section 5.6.2) can be used for annotation.

The annotation elements of a standoff annotation can use hypertext anchors and links to refer to the primary data and by doing so adding annotation levels. A standoff annotation can be used in parallel by different users, resulting in multiple annotations of the same primary data, which is besides layout the same structure as the tier annotation of spoken language data. The different layers can follow different annotation conventions, which includes the use of different theoretic foundations. However, standoff annotations, fully extended and combined, result in documents that share the primary data but have different annotations. This results in a procedure mentioned before which is the *Annotation of identical primary data*. In an XML context this is done by inserting start and end tags according to a given document grammar such as a DTD or Schema into multiple copies of the original source. Primary identical annotations can be founded upon different theories as well.

The requirements on a corpus are also defined in terms of reusability, and documentation with metadata by Gibbon et al. (1997c). The standard procedure for signal annotations, using different tiers for different annotations levels, and even separating annotation schemes from each other by using different tiers fulfills all of these requirements, if the annotation scheme is documented and the annotation tier refers to the appropriate, externally defined scheme.

5.5 Collecting corpus data

The creation of multimodal resources consists at least of the recording and the processing of the recording. Both will be discussed in this section.

5.5.1 Recording data

For the individual recording of data three major areas have to be taken into account. For video recordings for multimodal corpora the requirements are general, therefore the requirements are described in the context of video corpora.

The first aspect of recording is the *technical* aspect. For the recording of speech and multimodal corpora the technical equipment has to be appropriate. Appropriate equipment involves the availability of recording equipment such as a camera on a tripod, microphones and proper lighting, as well as sufficient storage media such as video tapes and energy for running the equipment.

The second requirements concern the setting of the corpus recording, i.e. what and who should be recorded. Trippel and Salffner (2002) for example describe in detail the scenario for the recording of a multimodal corpus,in which story teller of different cultures are used as informants for recordings of practiced speech with accompanying gestures. The description includes the description of the location of the narrator as well as a possible audience, and a set of recording prerequisites such as the recording of metadata.

The third and most neglected factor of multimodal corpus recording concerns the *ethics* of recordings. Ethic issues of multimodal corpora are difficult as a result of the volatility of speech which prevented the extensive use of this kind of data in the past. A second reason is the unawareness of informants of possible uses of the recordings. For technically less educated informants possible uses are an issue that needs to be discussed by the recording personal with the language informants.

5.5.2 Processing recordings

Part of the technical side of the recording is the post-processing of the recorded material. The options for post-processing are limited by the quality of the recordings, and usually a different medium is needed for permanent storage. To allow a maximum of reusability the quality of the initial recording has to be maintained. Processing steps therefore contain

Digitalization of the signal: The transfer from the original recording medium to a digital processing platform,usually a computer or a digital mixer unit with appropriate recording devices.

Cut of the recordings: usually additional sequences are present on the original recordings used for the technical setup, which are not to be used for the primary signal. In certain contexts personal data on the tapes have to

be anonymized, for this a coverage of personalized data with noise, e.g. a clear tone replacement for the audio file, or block overlays for video data, are preferred to cutting out, as this enables researchers to perform this themselves or to observe continuation of uncovered information.

Encoding of the signal: the encoding for archiving purposes has different requirements as for the actual work in the annotation process. For archiving purposes the encoding should preserve the signal, possibly without any lossy compression and without introducing further noise. For actual work the compression might be different, a lossy compression might be good enough for a given application. A visual annotation of gestures, for example as performed using the CoGesT annotation system (see Trippel et al. (2004b)) does not require uncompressed signal data but a video where the distinct body positions can be identified. For the latter it can be necessary to modify the signal, for example, by changing the contrast, color and brightness settings.

Storage on permanent storage devices: As the recording of multimodal corpora and their annotation is costly and reusability is a goal, the archiving of a corpus is required, see also Section 2.1.4 for a more detailed discussion.

5.6 Corpus annotation

Most corpus annotation has to be performed by human annotators, at least initially, before sufficient data is available to train a system. For quality assurance and consistency the annotation system needs to be defined with the requirement of human annotation and automation in mind, otherwise the annotation is not suitable for automatic post-processing. The relevance of corpus annotation in lexicon development is obvious if the lexicon is induced from the corpus. The annotated units can be inserted and used in the lexicon. For multimodal corpora, there has been approaches for annotation systems only recently.

Kennaway (2003) describes the requirements for a sign-language annotation system in terms of *language-independence*, *posture and movement recording* and *omitting irrelevant information*. A more general requirement for arbitrary, human usable and processable annotation systems can be given in the following terms:

Consistency, allowing automated post-processing, i.e. by a formal definition of a syntax, allowing automatic syntax evaluation and annotator assistant programs.

Language independence, providing a language independent description of linguistically relevant information of a signal. The language independence feature is meant to enable the use of the system independently of the language used in the signal. For certain features this may not be possible but in these cases the restrictions have to be defined, i.e. which type of language can be annotated using a specific annotation system. The result is a system that starts from a formal definition of linguistic instances, leaving the concrete functional interpretation to a research and theory based concept.

Unambiguous, annotator oriented description, targeted at a clear understanding of the system. The orientation towards the annotator is necessary to assure the consistent use of an annotation system, by one user within one annotation or by different users (*inter annotator consistency*). Two problems are related to annotation consistency:

1. The granularity of the annotation: if the annotation is too detailed allowing different symbols to be used, the complexity can result in an increased error rate. The solution is to create annotation categories as general as possible.
2. The underspecification of annotation: if the annotation is rather broad allowing a lot of generalizations, the variety of matching instances can be too many, so the annotation system needs to be as detailed as possible

Granularity vs. underspecification results in a clash, and a suitable compromise needs to be found.

Standardization, for portability reasons a widely used annotation system is usually more suitable and therefore preferred to a system that is slightly more appropriate to a specific task.

The requirements have to be taken into account when an annotation system is selected and the annotation process is started.

5.6.1 Selecting existing annotation schemes

For different modalities a range of annotation schemes exist. The encoding of annotation systems, ranges from idiosyncratic, proprietary fonts to standard font encoding tables as, for example, used by Unicode (Aliprand et al. (2003)). Based on the mentioned requirements annotation schemes can be selected.

Annotation of the audio modality

For the audio modality the selection of an annotation schema is rather simple, though concurring standards exist. For the annotation of words, usually the language specific orthographic conventions are used, based on some sort of a standard encoding. For some languages there is more than one standard writing system, for example in Japanese with the *Kanji*, *Hiragana* and *Katakana* systems, while in others there is no writing system at all — for example in the west African language Ega (Ivory Coast), though there are proposed writing systems (see Salffner (2004)). For some language there may not be a *standard* orthography as in Ibibio (Nigeria).

In the spoken languages without a standard orthography often an ad hoc orthography is developed based on phonemic transcriptions and the resemblance of Latin letters is used as the orthography (see for example Bole-Richard (ated)). The phonemically oriented orthography is motivated by western languages, however.

The transcription on other linguistic levels is also rather standardized, for phonemic transcriptions the *International Phonetic Alphabet* (IPA) (see Nolan et al. (1999)) is used, sometimes using different encoding systems such as the SAMPA (see for example Gibbon et al. (2000b)) IPA encoding using ASCII.

For other annotations the standardization of annotation systems can also be available. With the *Tone and Break Index* (ToBI, see Silverman et al. (1992)) there is a *de facto* standard procedure and core set of symbols for the annotation of intonation, with specified systems for a variety of languages, for example for German and English.

Annotation of non-audio modalities

Annotation systems for non-audio modalities, such as the visual modality, is not as easy. Non audio modalities have not been researched from the linguistic perspective for a long time, due to the technical requirements for annotation. The storage capacity of video data is comparatively large. The processing of the large amounts of data coming with non-audio modalities, for example for time aligned annotation, requires powerful processing computer hardware. Due to the technical requirements the visual modality is rather new to the field of linguistics, though some annotation systems are available.

The annotation systems for visual modalities differentiate between different parts of a body, for example there are annotation systems for facial expressions used in psychological studies of faces. Rosenberg (1997), for example,

compares the *Facial Action Coding System* (FACS) (see Ekman and Friesen (1978)), which is classified as an observational coding system, to other systems, including the measurement of electrical potentials from facial muscles. FACS provides a face description system which is based on a number of prototypical expressions of a face, which can be combined. Face description systems assign values for different parts of the face and movements, for example raises of the eyebrow, or stretched lips. The system is not formalized but only defined by prototypical expressions of the facial area.

For hand gestures there are different annotation systems that can be grouped into the following areas by the origin of the annotation system:

Annotation systems derived from robotics for example as described by Kranstedt et al. (2002): The annotation system serves to model a humanoid behavior of a robot. Consequently the annotation is based on precision and defines exact positions and local vectors. For human annotators the required exact measurements are hard to achieve, usually the transcription is generated from data gloves or comparable systems.

Annotation systems from speech technology to model human-machine interaction (see for example Steininger (2001)), where especially pointing gestures by human users are modeled for the resolution of deictic expressions.

Annotation systems for the transcription of sign languages, see for example HamNoSys (Prillwitz et al. (1989)). Sign languages are restricted to a gesture space, i.e. the conventional areas within which signs are produced.

Annotation systems for psycholinguistic research as done by McNeill (1992) and others. Based on the idea that gesture and speech are inseparable evidence of speech there is a separate description. Gestures are separated and described according to functional prototypes.

Semantically independent gesture description systems from linguistics, for example the FORM system (Martell (2002)) or CoGesT (see Gut et al. (2002), Trippel et al. (2004b)). Gesture description systems are meant to describe gestures independently allowing a more general connection of speech and gestures.

The decision which system to take is far from easy as all systems are currently under development and not widely accepted. However, a semantically independent gesture description system can, if acceptable defaults are found, be transformed into an avatar input format similar to the formats used in robotics. Kennaway (2003) describes the conversion procedure for gesture

description to avatar formats for HamNoSys, which is, though applied to a limited domain, based on similar premises.

Driven by the needs of the extension of spoken language systems towards multimodal processing, the annotation schemes for deictic gestures in human-machine communication are widely used, nevertheless, they only cover a small part of communicative situations.

For other gestures, for example, the previously mentioned eyebrow movements, there are only ad hoc transcription systems used in limited project contexts.

Annotation schemes for written language data

For textual data morpho-syntactic annotation has been in the focus of corpus based linguistics for quite some time. The long development process for morpho-syntactic annotation results in the question if the selection of annotation schemes is similar to the previously discussed areas. For the audio modality there are simple, established core annotation schemes, for non-audio modalities there are hardly any established ones.

For morpho-syntactic annotation there is a huge variety of annotation systems, which are highly theory and language dependent as the description system of a language is always theory dependent even if the theory is put into words (see for example Erlenkamp (2000), p. 9). One example, defined by Schiller et al. (1995), identifies more than 50 different morpho-syntactic tags for German, split into 11 parts of speech. The Bielefeld-Verbmobil Lexicon (see Gibbon and Lüngen (2000)) distinguishes 13 different syntactic categories which are used instead of parts of speech, using 70 inflectional classes for nouns alone. Similar phenomena of having a variety of classification systems can be found for other languages, for example Crystal (1967) discusses the variable use of word classes for English.

A step towards the description of best practice was accomplished with Leech and Wilson (1996) within the European EAGLES project, where 13 different parts of speech for language independent corpus description were defined. Based on the EAGLES recommendation, the MATE annotation guidelines discuss the use of 15 different parts of speech (cf. Mengel et al. (2000)).

The variety in number is becoming even more confusing if typologically different languages are taken into account or even languages that use a completely different inventory. For example, Erlenkamp (2000) (Section 2.2 and 2.3 and Chapters 5 and 6) discusses the universal nature of word classes in

the context of German Sign Language, showing the absence of noun-verb distinction in the traditional sense.

Although the classification of morpho-syntactic categories and other features of written language differs with reference to language and theory one feature is shared by all of them: a fixed and small set of categories, used according to a catalog of fixed categories, based on simple, character based descriptions, usually using simple ASCII encoding.

The fixed annotation scheme is the basis for all annotations, resulting in automatic pre-evaluation of the annotation for syntactic correctness and consistency for automatic processing.

5.6.2 Annotation of corpora

The annotation of corpora, especially for huge quantities of data requires both, trained personnel and tools assisting the personnel. Some parts of the annotation process can be automated, for example the word level annotation based on a standard orthography can be enriched by a POS-tagger to include the annotation for already known words or for information based on inflectional morphology. As the annotation enrichment includes a lexicon, the annotation with added information by this process cannot be left uncommented. The rules or constraints — lexicons for annotation enrichment are part of these — are incomplete, because the recall is limited (see Section 5.3), consequently the use of enriched corpora is limited as well. If the resulting annotation is not evaluated, checked and corrected by experts the best possible result of this corpus would be the 'proof' of the existing rules and constraints.

The problem of annotation enrichment leads to the fundamental requirement for the annotation of corpora which is the manual annotation of the data. It can be argued that the manual annotation is error prone and possibly inconsistent, but there is no way to avoid manual annotation, as the human factor is the knowledge containing procedure. Using linguistic knowledge of annotators does not imply, that this knowledge is obvious and available, as Trippel et al. (2003) point out that the relation of different linguistic levels can be derived automatically more completely as manually because the latter requires appropriate hypotheses, while the earlier can show patterns which can hardly be detected by human investigation or can only be found by chance.

The process pattern recognition requires a machine readability and the only appropriate way of annotation is based on the use of software allowing this, assisting the annotator in time alignment of signal and transcript and other labels. The use of 'pen and paper' with manual notes on timing would require a post-digitalization, resulting in double work and even more errors.

However, the use of a special tool is less relevant for the processing of the annotations than the data format and availability, as the data format can be transformed and processed with different tools. One prerequisite of the annotation of different linguistic levels which allows a view of more than one linguistic level at a time is by providing a way for annotating different annotation levels, as used in phonetics in a score.

Using these annotations points to another related topic, which is storage and archiving of corpora. Storing and archiving is necessary for reusability as discussed before. Ensuring reusability by a transparent data structure is a requirement, including a document grammar, a content model for the labels and annotations, and a proper description of the resource, including the description of the used symbols and encoding. A standard data format allowing these different levels is given with the XML syntax description, allowing for document grammars, already available with Document Type Definitions, and content models, which are more fully supported by newer schema languages such as XSchema (Thompson et al. 2001 (2004a)).

Archiving of XML data is another problem, as the storage space is not crucial any more, especially for textual data due to extended hardware capacity, but the problem exists for storage persistence. Persistency is a technical issue, discussed, for example, in Bird and Simons (2002), see also Section 2.1.4. Related to persistency is the requirement of interchanging resources

Different institutions that combine efforts and resources or need to merge resources produced by using different tools, at different times or by different infrastructure is becoming more and more important due to the cost of resources. Interchange includes the portability issue (see Bird and Simons (2002)) as already discussed in the lexicon context. The combination of resources leads to some major problems which have to be resolved when combining resources. The cases of resources based on *different* source material and on the *same* source material have to be distinguished.

The easiest case is the first one, where resources are different based on different source material which is a problem of format conversion into a common format in order to use the same tools and procedures. The use of a common format, for example, applies to data on different languages, for which corpora can be processed using the same tools.

The case of resources based on the same source material can be subdivided into similar annotations and identical annotations. If the data contains similarities, because it is based on the same source material, then the problem is the combination of the different annotations, consisting of a union of the resources to be combined. If the intersection of the resources is empty that means that the annotation of different linguistic levels is done in different

ways or at least the annotation is based on different standards. The combination in this case can be accomplished by merging two annotations, creating one annotation with the union of the tiers. However, one problem of merging annotations is the consistency of the metadata description. The merging of the metadata is discussed in Trippel et al. (2004a) and in Section 6, where the MetaLex approach allows inferring information on the annotation levels.

Far more complicated is the case of identical annotations, which seems to a absurd to consider as a problem because one can easily be left out. The real problem therefore is not the handling of identical annotations but the *identification of identity*. The simplest possible identity is a character based identity of different annotations. If no difference can be defined, for example because both sources were taken from a third hand party, then the resources are identical. Nevertheless, storing a corpus usually includes some sort of processing, for example the transformation into a common format.

An XML based annotation format such as the TASX format can be formatted for human interpretation by inserting space characters, line breaks, etc. These formatting issues are not supposed to change the semantics of the resources, but the character based comparison shows differences, hence the earlier case of similar resources could be assumed. However, this is not desirable as the relation of the segments to each other indicate the identity.

An even harder case of almost identity can also be illustrated using the TASX-format. The segments, called *events* in TASX-terminology, have a start and end timestamp and an identifier, all three realized as attributes. Two differences here can lead to problems, first the *precision* of the timestamps. Depending on the tools for processing the annotation format allows arbitrary precision, but the annotation tools and annotators do not. In fact, there are physical limitations to the precision, for video data based formats for example one individual picture which is usually, depending on the format, a part of a second. For audio files, though the granularity is finer, there is also a similar phenomenon. In addition to the granularity there is a perceptive precision, a granularity where the signal processing algorithms, either automatically or by the human annotator, cannot further differentiate. Some tools allow or include some sort of rounding for these reasons. Different tools handle rounding differently, hence the data transformation can result in differences that are far from processability and perception but which are available in the source data.

As different tools use different methods of granularity, including rounding and chopping floating point positions, a conversion forth and back can also result in deviating data. The same is true with idiosyncratic character encoding and reserved words and characters and escape sequences that might not have a bidirectional translation.

A solution to this is the definition of a *distance metric*, measuring the distance of timestamps, i.e. defining a value $\varepsilon > 0$ which allows a tool to regard two timestamps as identical if the one is within an ε interval of the other. As different annotation granularities, for example *discourse* vs. *phoneme*, require different granularity of timestamps, this ε has to be defined in dependency of the segmentation. One possibility would be to define ε as a certain fraction of the average length of a segment on one annotation level.

Another problem, not related to the timestamps is related to the *identifiers*. By definition they are supposed to be unique, hence they are often generated. If they are not referred to within a corpus, transformations can re-generate — possibly different — identifiers. In this case the identifiers could be left out for the comparison of two annotations. There is another caveat related to identifiers, especially if left out for comparison, which is the *predefinition* of an identifier. If two resource contain the same identifiers that are the same for different segments then the consistency of a combined source has to be maintained, if necessary by re-generating unique identifiers. The process of identifier re-generation has to be accomplished before the unification as references to these identifiers might need to be adjusted.

5.7 Summary

There is a strong relation between lexicons and annotations, procedurally as one can serve to enhance the other and formally as both are graph structured. It was argued that annotations can be interpreted as a special form of lexicons and that lexicons can be interpreted as annotations, for example a lemma based lexicon could be interpreted as an annotation of a list of lemmas. As lexicons are based on corpora, the creation of corpora especially in the multimodal context was discussed, arguing that the corpus has to be large and varied enough to be representative for a part of the language under investigation. It was shown that the formalisms used for spoken language characterization are general enough to serve as a general annotation model. Requirements for annotations were defined and it was shown which consequences these requirements have for the recording of source material, the selection of an annotation data format and annotation standards in the annotation process. The handling of the annotations for archiving, retrieving and merging was the final point of discussion. The discussion showed that the Lexicon Graph Model and the Annotation Graph Model can be seen in connection to each other and do not exclude each other, in this sense they are sound. What was not shown however was how annotations can be used as lexical resources. To allow the use

of lexicons however, the descriptive level of these language resources has to be looked at. The reason for this is that it is necessary to know what kind of information can be found in an annotation before it can be used as a resources for the extraction of lexical information.

6

Metadata for lexicons in corpus based lexicons and lexicon search

To describe the content of language resources metadata are added to these resources. This metadata should be able to offer information in the process of the extraction of lexical information. Hence, it is necessary to look at the available information. This is done by looking at different metadata, first by its use, then by existing standards and conventions for the definition and encoding of metadata. Furthermore, it needs to be considered if the metadata standards have a common structure and how this can be related to lexical structures.

The description of metadata is more and more coming into the focus of different research interests in the language resource community, though more recently it has primarily been used for archives and libraries. As the archives and libraries start from material that is intended to be interchangeable their ideas are reused by the language resource community.

More important for the lexicon development is the question which data categories for a lexicon are part of an annotation, i.e. which parts of a lexicon microstructure can be found in the corpus. The knowledge which kinds of information are available in a corpus is part of the metadata description of the corpus.

In the present discussion of metadata there is a bias towards corpora. The turn towards lexicons becomes relevant as the distinction of lexicon and corpus is diminishing, resulting in very similar but more general requirements for metadata. Lexicon metadata are not only used for lexicons but also processed in the same way as lexicons, therefore a metadata lexicon *MetaLex* is described. Another aspect in the discussion of metadata for corpora is that this metadata can be used in lexicon generation. Taking a multi tier score: every

tier contains information from a specific data class. This class of data can be automatically extracted based on the metadata applied to the tier level.

6.1 Purpose of Metadata

Metadata are information used for locating and classifying resources. Descriptions of resources are supposed to be general enough to differentiate and cluster resources according to a wanted application. Traditionally, metadata standards and categories have been developed by librarians and archivists, usually based on textual sources, though textual resources are only a subset of all resources. The descriptions for textual resources are more advanced in the process of general acceptance than others, hence they usually form the starting point for a discussion of metadata. The application of these standards to general resource descriptions with a bias towards multimodal resources in linguistics will be the focus of this section.

Metadata descriptions for spoken language and multimodal corpora are different from textual resources due to fundamental differences in the structure of multimodal corpora. Nevertheless, the detailed description cannot be of more importance for the classification of resources enabling reusability and portability, which are crucial due to the costs of the creation of corpora that are built on multimodal data.

Spoken language corpora are extremely expensive, much more than collections of (written) texts. According to various investigations time for annotation ranges from the factor 10 to the factor 100 for the annotation of multimodal data (see for example Gibbon et al. (1997a)), which results from the time needed for segmentation and transcription depending on the linguistic level. The signal is part of the corpus, as only the signal enables the use of certain corpora for example for applications in psycholinguistics, speech engineering and phonetics but also to a certain degree for socio-linguistics and language varieties studies. To avoid additional costs for obtaining corpora and for file transfer in addition to locating relevant information, very detailed descriptions of the content are required, enabling selection and restriction to existing corpora by grouping them according to their relation to each other. Resources can be related in different respects including similarity and coherence relations.

Similarity relations of resources are needed for classification, grouping and sorting of resources where similarities of resources are of interest such as similarities of Type/genre, Topic, (Grammatical) structure, audience language, author, age.

The listed similarity relations are used for archives, data repositories and libraries for cataloging and access by concordances, by fixing features for automatic filtering of resources and parts of resources or human users for manual search for resources.

Coherence relations are the content relations of different resources such as resources referring to each other, referring to a common external resource, or making use of the same premises such as knowledge or context. These coherence relations can be defined explicitly, e.g. realized as (hyper-)links, and relations of coherence can be typed. Explicit linking can be one-directional, by pointing from one resource to another, or bidirectional by pointing from a source to a target resource and vice versa. The type of relation can be gradually defined for example by typing one resource as being *related to* a second resource or by using meronymic and taxonomic relations or crossreferences. In XML the World Wide Web consortium's (W3C) standard *XLINK* (see DeRose (et. al.)) provides for these requirements. Though links and references as coherence relations are metadata as well, they are usually not included in metadata discussions.

For (linguistic) corpora the similarity relations are especially important. Insertion and maintenance of metadata in the context of corpora seem crucial for effective access and usability, hence this chapter neglects the coherence relations and discusses various aspects of metadata for similarity relations. Coherence relations within the resources are the focus of various linguistic disciplines itself, for example studying co-reference. This field is not a special part of the field of resource description.

6.2 Metadata for different uses

Metadata are subject to an application, i.e. the pure existence of metadata does not imply its usefulness. Just as different levels of annotation require different annotation units, different linguistic theories demand a description in different ways. This results in a variety of units and annotation elements. The varieties of annotation elements can be described in terms of metadata categories, some of them are described below.

Different uses of metadata are illustrated by Figure 6.1. The corpus is based on a data model taken from the TASX format (see Section 5.1.1.2 and Appendix 10). Each level of the annotation, i.e. segment, tier, subcorpus or corpus, can receive metadata descriptions which are in two classes.

Metadata can be used to describe a resource in a resource archive, resulting in a *Resource Description Library*. For the description of a resource in

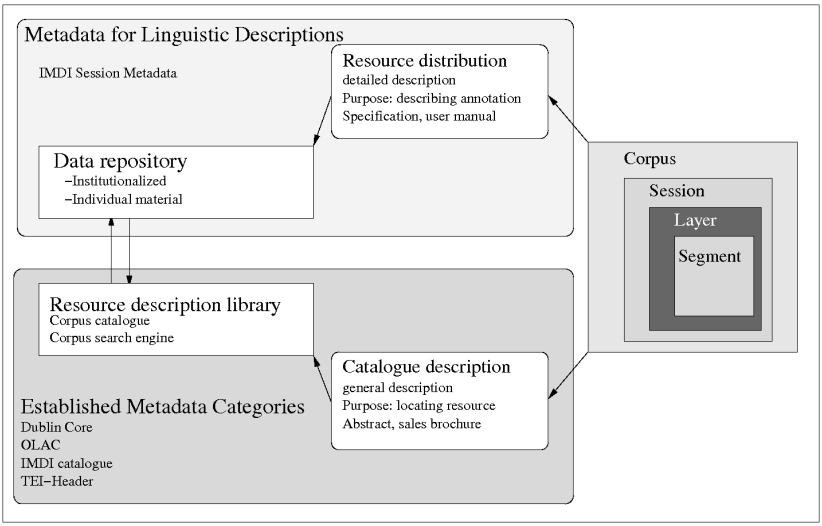

Fig. 6.1. Relation of catalog and annotation level metadata

such a library, which can be rather large data repositories, specifications are needed such as data formats, physical or logical location of a resource and system requirements for accessing and retrieving a resource. In the textual domain the metadata for a resource description library can be compared to an abstract for an article or a sales brochure, with highly conventionalized information, and the description being rather independent of the resource itself.

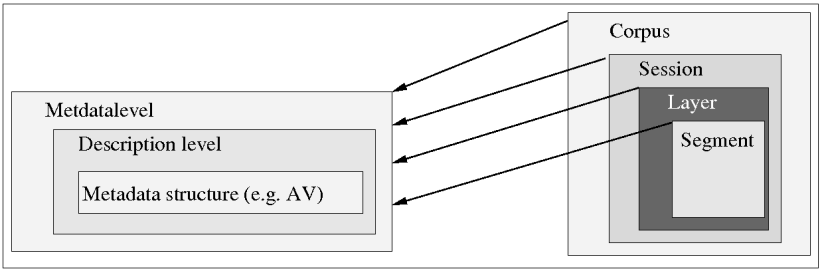

Fig. 6.2. Metadata representation for annotation levels

For the description of the individual annotation levels a different type of metadata description is needed. Figure 6.2 illustrates possible description levels, again exemplified using the data model of the TASX format. It shows that the metadata description can be attached to every level of the annotation. The details on the annotation level are used for applications and detailed research queries, allowing adjusting tools, for example by selecting an appropriate font for displaying a sequence of characters in a computer program or pointing to a linguistic theory according to which an annotation was produced. This *Metadata for linguistic description* can be interpreted as a *user manual* to a resource, describing everything in detail which can be found in the resource. Among these descriptions, metadata are on all annotation levels.

Metadata on the session level covers information needed on the structure — the data format — and the content of the individual primary data. Session metadata also includes information on the source of the annotation, i.e. the text or signal origin, such as authorship or recording attribution, place, and dates. Metadata on the layer level is used to specify an individual annotation layer or tier, using categories such as annotator, annotation formalisms including data format and encoding, technology used in annotation, etc. Metadata on the actual annotation event can include information deviating from the data on a larger level or technical information for retrieval software.

The introduction of metadata on the event level shows another characteristic used in the description of annotations based on this granularity feature: Information items can be inherited from other annotation levels. The inheritance of information is one reason why the need for the more detailed linguistic description level has not been overly prominent, yet, because it was assumed that the information was all present on the catalog level, therefore the description level could inherit it from the upper level. A more detailed discussion of inheriting metadata information can be found in Section 6.5.

Different metadata levels are interrelated, by sharing data categories and information. The linguistic description needs to be far more detailed. Metadata can be described as well in terms of uses, i.e. potential uses of the data because the data categories are dependent on the purpose they are used for. Corpus metadata and catalog metadata may share the same presentation structure but may use different vocabularies and data models. Metadata in Lexicons will have a different structure again, as their purpose differs, see Section 6.3.3 for more details.

Metadata for human use needs to be instantaneous interpretable for a person (given that this person has a certain communicative background such as language competence). The presentation structure (see Gibbon and Trippel (2000)) such as the layout is important to the user as well as the content,

though the presentation does not need to be absolutely fixed as robustness seems to be easier for humans than for computers, so that minor variation of words and description does not result in incomprehensible information for a person. Metadata categories of possible interest are defined in various practices, for example with Dublin Core (see Section 6.3.2.1), OLAC Metadata (see Section 6.3.2.2) and IMDI (see Section 6.3.2.4).

In the context of the semantic web (see Berners-Lee et al. (2001)) an increasing number of applications need to access metadata which is structured, well defined and provides for the applications input needs. As robustness is an issue for systems the structure needs to be fixed, following a document grammar and for interpretation purposes the vocabularies need to be semantically defined. Metadata definitions for system use can implement RDF and RDFS which were developed by the W3C (see Lassila and Swick (1999); Binckley and Guha (2003)) for metadata encoding. Some of the data categories are the same as the previously mentioned ones for human use, but additional data categories are relevant for systems, such as internationalization information (mime types, see Freed and Borenstein (1996a) and Freed and Borenstein (1996b), encoding such as Unicode ISO 10646 (2000), etc.).

6.3 Metadata sets

6.3.1 Historic metadata catalog

An early metadata catalog is described by Cutter (1904). For the creation of a library catalog he defined metadata categories for books such as `author`, `title`, `subject`, `classed subject`, `crossreference`, `formetry` — the latter being his term for *genre*. Cutter's definition of keywords is utterly intuitive, for him these are words "probably the first looked under by the class of people who use a library." (see Introduction to Cutter (1904)).

Cutter requires a consistent selection of keywords though he does not define consistency in terms of structural or semantic consistency but intuitive again and defines the macrostructure intuitively as the ordering which enables locating the headwords, which for him is basically the alphabetical ordering of headwords. Nevertheless, he introduces keywords and uses a hierarchical structure for authors and co-authors with content models for names, including variation of names in different languages. Some of the metadata categories described by Cutter are the same as in modern metadata sets, though he was early in defining such a convention.

6.3.2 Core metadata sets

The relevance for a standardization of metadata can partly be measured by the number of different metadata conventions that are around. The following sections will describe different metadata practices and discuss their common features and structures. The conventions are basically all intended for metadata catalogs, not for data repositories.

6.3.2.1 Dublin Core

The most frequently used metadata standard, which is also used as reference, is developed by the Dublin Core Metadata Initiative defining different core categories (see Dublin Core (2003)). The standard *unqualified Dublin Core Metadata* consists of 15 core elements which are optional and can be repeated arbitrarily often and include categories such as *title*, *description*, *source*, *subject*, *publisher*, *creator*, *language*, etc.

Further elements and refinements have been specified (see Dublin Core (2003)) to extend the number of elements, include the encoding scheme and use a defined vocabulary for assigning a type to resources.

The DCMI, based on librarian and archivers experience, is limited in the usefulness for linguistic data however. One reason is, that multi language documents cannot be sufficiently described using the metadata. Though it is possible to include reference to more than one language, it is not defined how to describe resources with a mixture of languages. For traditional material in libraries this might not be the usual case but for linguistic studies and resources for example in comparative settings this is not true. Another reason is that many data categories are under-specified. For example the category *date* does not define which date of a resource might be meant, such as the date of publication, and the date of creation, the date of recording. Nevertheless, there tends to be a broad agreement on the DC elements as being the core of metadata.

6.3.2.2 OLAC

The Open Language Archive Community (OLAC) (Simons and Bird (2002)) tries to overcome the shortcomings of DC metadata categories for linguistic resources, to enable the formation of a worldwide accessible data repository for linguistic data. To maintain a constant set of information within the catalog the content of the data elements are widely fixed by a controlled vocabulary. A comparison of DC and OLAC elements reveals that OLAC is a real extension

of DC. The extensions include two features that are crucial for data in the linguistic environment.

The first concerns the format, i.e. the data format depends on technical factors as well as on linguistic theory. Because electronically available data—and linguistic resources have to be electronically available for efficient processing (cf. Gibbon et al. (1997c)) — information on the used platform (esp. CPU, operating system and used software) is required. Technical information implies reference to technical limitations and possibilities. *Markup* consists in this area of a technical description of a data format and needs to be recorded as well, defining the structure of data. The second is about linguistic categories, which means that information on functionality and linguistic types as well as the subject language refer to linguistically relevant information. Different genres of texts as well, different subject languages are most significant in material dealing with these differences. Additionally *markup* can be seen as an implicit reference to a semantic interpretation of a grammar, where the form, e.g. an XML-tag `<v>`, is interpreted as a specific function, e.g. `beginning of a verb`. However, as traditional markup definitions imply the semantics only in the form of comments and specifications, a formal semantics of tags is not implied for most markup structures for resources[1].

The OLAC metadata set is an improvement for linguistic data but still there are shortcomings. One of them is the issue of underspecified data categories. A second is that the relation of different languages in multilingual resources is not easily specified. And finally, resources based on signal data are not sufficiently described mentioning the technical requirements for these resources.

6.3.2.3 TEI

The structural markup of different kinds of text with common structures was the objective of the Text TEI. For the differentiation of text sorts and for cataloging a general metadata header was developed. The TEI metadata header includes information on the description of the file, containing a bibliographical description of a resource file, such as authorship, property rights, source and size. The description of the encoding, including character sets, measurement units format, etc. is also contained in the header. This also applies to a description of the content, such as keywords, creation date, language Revision information, such as changes of the resource with date and extend.

[1] A change of this is approached with the definition of RDFS (Binckley and Guha (2003)) as a semantic definition language for RDF documents (Lassila and Swick (1999))

As Trippel and Baumann (2003) already showed, most DC categories have a direct correspondent in the TEI file description metadata, but with further subclassification and a finer granularity. The TEI metadata set is especially relevant for textual data in TEI encoding, hence no information on data formats is necessary as this is already part of the document grammar. However, similar shortcomings as for DC and OLAC apply to TEI metadata as well, as is the case for the underspecification of metadata categories, the handling of multilingual resources and the treatment of signal based resource.

6.3.2.4 IMDI corpus metadata

The ISLE MetaData Initiative (IMDI) used a slightly different approach for their metadata development by starting from the domain of multimodal data. Their intention was to catalog and retrieve such data. Consequently the metadata are distinguished into

Catalog metadata (see ISLE Metadata Initiative Catalogue Metadata (2001) in the bibliography) which contain information to be recorded in data repository catalogs, such as `resource name`, `title`, `description`, `subject language`, `content type`, `data format` (text, audio, video), `annotation unit`, `date`, and `property rights`.

Session metadata (see ISLE Metadata Initiative Session Metadata (2001) in the bibliography) which contain information on the primary data, i.e. one particular data recording session. The session metadata includes categories such as a session identifier, title, date, location, description, collector, project relation, content, genre, resources (media files and annotations).

The authors of the IMDI metadata set bore in mind the categories of DC and OLAC wherefore they included a mapping of metadata categories to their newly specified ones. Due to the distinction of *session* and *catalog* and many new categories, the granularity is much finer to match the requirements of the corpus resource management both technically and linguistically. This is apparent in the technical information on the data format, which includes a quality and file location field, and the linguistic categories on language, participants, etc.

Some main shortcomings of other metadata practices are not solved, however.Though for some data categories a controlled vocabulary is included the content model of other categories is still under-specified. Additionally, the handling of multilingual data is possible by differentiating between different

participants with their roles and languages within the resource but a change of languages within a resource is not sufficiently handled.

6.3.2.5 Conclusion of metadata for corpora

Existing metadata conventions are suitable for different purposes. DCwere developed for the cataloging of resources in libraries the DC metadata are suitable for bibliographical sources such as texts, articles and books; however for corpora and multimodal data a lot of metadata categories are missing.

OLACwas developed for the cataloging of linguistic resources in data repositories, intending to extend DC by additional linguistic data categories and some technical information of electronic material for linguistic resources of one language, one annotation, one annotator and one medium. In contrast to this theTEI metadata was intended for encoding texts the TEI metadata allows the encoding of metadata categories relevant for textual sources.Finally, with IMDI from the language engineering perspective approaches resources from the catalog and session level differently, allowing an inclusion of multimodal data in data repositories and describing annotations to a certain degree.

The main shortcomings of all metadata sets remain. One apparent problem is the under-specification of data categories by not defining content models for all data categories. This could be answered by closed vocabularies and by defining formally the semantics of the data categories. Finally the handling multilingual sources can be made possible by differentiating different languages into different annotations.

The differentiation of languages into different annotations is related to another major shortcoming of corpus metadata not mentioned before but applying to all introduced metadata sets: the problem of handling annotations provided on different layers, such as standoff annotation (McKelvie and Thompson (1997)), primary data identical annotation (Witt (2002a)) or even score annotation as common in phonetics (e.g. annotations on different linguistic levels with `Praat`, `TASX-Annotator`, `xwaves` or `wavesurfer`). These tools can create annotations in which the levels can include different kinds of annotations units, caused by their annotation standards and linguistic theories. Also the date annotators and time of annotation can vary, as well as the tools, which result in different restrictions to the annotation.

To solve these problems the metadata needs to be available on different levels. These levels are related to the structure of score annotations, i.e. metadata on catalog level being the information used in large data repositories for locating a specific resource providing basic information for the retrieval of

further metadata, such as the file format and location of the resource and infrastructure requirements for retrieval (such as software to access the data). In addition to the catalog metadata metadata on the session level needs to be provided. On the session level information is needed on the structure, i.e. the data format, and the content of the individual primary data. But using the same argument metadata on the layer level includes information on the specific annotation such as the annotator, annotation formalism such as data format and encoding and technology used in annotation, etc. Finally each segment, or annotation event needs to provide the metadata which might include deviations from the layer metadata or technical information for retrieval software or other tools.

6.3.3 Lexicon Metadata

Corpora as large data structures have a longer tradition of formal metadata descriptions as lexicons, which contain metadata but where conventions for metadata are not as well established. For machine readable lexicons this is even less so, for example the database for the Babylon tool (see Babylon (undated) in the bibliography) describes a lexicon source format for a multi purpose glossary, which does include no metadata information besides some general information — a glossary-title, author, language to be described and target language in the header of the lexicon source file. The reason for this is that these lexicons are produced for a single, fixed application that has built-in support for the data format, and implicitly contain a description of coverage and data categories. A look at print dictionaries and standards derived from these provide a first approach to lexicon metadata before turning to more general approach to lexicon metadata categories.

For traditional print dictionaries such as the OED (Murray et al. (1970)) or the Oxford Advanced Learners Dictionary (Crowther (1995)) the first metadata descriptions can easily be seen as an application of Dublin Core to the book, including archival information on property rights including publisher and editorial information, year of publishing, type of publishing, etc. Nevertheless, a dictionary usually contains an extra section where a potential user finds information on the use of the book, in other words a semi-formal description of the structure of the book.

The TEI guidelines (Sperberg-McQueen and Burnard (2001c)) describe a markup for print lexicons and as these include structural descriptions it could be expected that there is at least a structural way of encoding structural information. However, there are no additional metadata categories provided be-

sides the standard TEI-header, which was already described for corpus metadata (Section 6.3.2.3).

Prose descriptions used in the front of print dictionaries can be marked up using other, general text features, i.e. no structural markup for lexicon description but text description is used. One feature especially relevant for lexicons, is the presence of a sort key, which is intended to be given as an attribute to lexical entries, to enable a sort that deviates from a character based ordering (see Sperberg-McQueen and Burnard (2001c). Section 12.1). The reason for the inclusion of a sort key exemplified by some letter-number combinations are traditionally sorted according to the numerical value, but that can be sorted by spelling out the number as well. In the latter case the spelled out number in the sort *key* is the spelled out variant, implying that no formal description of a macrostructure is included, though the problem is at least marginally addressed. Only the alphabetical order is taken into consideration. The lexicon microstructure is not explicitly described outside of the document grammar, and the mesostructure is described by pointers, but no formal metadata description is given.

The absence of a formal metadata description of lexicons is also true for the TEIs ideas on terminological databases (see Sperberg-McQueen and Burnard (2001a)), where a concept based structure is assumed. For terminological entries, metadata are included on the level of the individual entry, such as information on editing and status of a term. The metadata on individual terms is handled structurally in the same way as other lexical information, though the microstructure contains a simple hierarchy as well.

The ISLE Metadata Initiative (IMDI) are besides metadata descriptions for corpora also working on a proposal for metadata elements for lexicon descriptions (Wittenburg et al. (2001) and a more recent draft by the IMDI Group, IMDI Team (2003)). The distinction for the description of lexicons is drawn similarly to the ones for corpora, i.e. the lexicon as a whole referred to as *lexicon resource* (*lexicon object* in the older proposal) and individual *lexicon entry*.

The lexicon resource is described similarly to Dublin Core categories, including technical information such as resource location, encoding, date, size, etc. Property rights are only addressed in terms of access restrictions in the current draft.

The description of the individual lexical entry is in fact the description of the lexicon microstructure, including currently 10 different data categories, including *headword*, *orthography*, *morpho-syntax*, *phonology*, *semantics*, etc. Administrative and editorial categories as used in terminological dictionaries

are currently not included, as well as further data categories extending the corpus metadata conventions.

Just as with corpora completely specified lexicons require a variety of data categories for a thorough description. Two basic uses can be distinguished, the lexicon resource description and a detailed description of the lexicon itself. For locating a lexicon for a specific purpose, a general description of the individual resource is required.

The general description is similar to that described for the catalog descriptions for corpora and other resources as described by the core metadata sets in Section 6.3.2. Nevertheless, a lexical resource needs further information, for example the number of entries is relevant information for the description of a lexicon. In addition to lexicon statistic information, at least some information on the lexicon structure has to be included, which could be described as a lexicon *type* identification, i.e. information on the contained lexicon data categories. Without the data category information a relevance of the lexical resource can not be assumed.

The description of data categories included in a lexicon does not need to provide a document grammar for the lexicon from the start, because for the reuse in other applications a transformation could be required any ways, but an indicator of the available categories enables the restriction to appropriate information only. A detailed description is necessary for the lexicon source to be included with the resource itself. The detailed lexicon source description is itself twofold, i.e. a formal syntactical and formal description of the resource, and a more detailed description of the lexical entries.

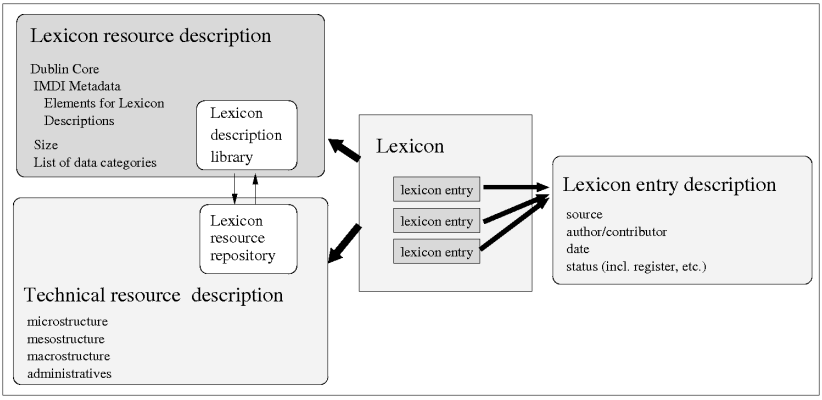

Fig. 6.3. Description levels of a lexicon

Figure 6.3 illustrates the description of the lexicon on various levels. The *technical resource description* and the *lexical resource description* shown are closely connected to each other, though their purpose is different. While the technical description contains the concrete syntactic information on the resource, including a formal specification of the data categories contained in the microstructure, this is not true for the more general resource description, which can again be compared to a sales brochure in a catalog, which is supposed to follow the syntactic constrains of the catalog. Nevertheless, it contains a general description of the resource as the technical resource description does.

Analogue to corpus metadata, lexical metadata can be differentiated into different groups, such as *content*, *property rights* and *instance*. Editorial information comprises data on the creation and editing of lexical items, covering contributing personnel with institution and qualification, as well as dates and a change log.

One class of descriptors is problematic in the context of lexicon descriptions, which is the *content* category of Dublin Core. For a full description of the content a reference to a semantic field or at least to a register could be wished for, among others. However, the distinction between metadata and lexical data then becomes obscure. To avoid the artificial differentiation between metadata and content, a strong restriction on lexical metadata categories is required, where the content categories are as much reduced as possible, editorial information can be applied automatically by applications, based on the user name and a system date, besides application inherited technical descriptions. The instance categories are required for identification and basic description of a lexical item. As one lexical item can have more than one type, all these categories besides the instance information are optional and can appear arbitrarily often.

A problem of the inclusion of metadata for every lexical item is the massive redundancy if the same metadata are recorded for different lexical items, such as the same person inserting and/or maintaining a number of lexical items, in which case maintenance and data integrity, for example changes and updates of metadata can result in inconsistency. If a tool does not insert administrative information automatically, the acceptability by human users may be low as the data has to be inserted manually. A solution to the redundancy problem is provided in Section 6.5.

6.4 Encoding and representing metadata

The description and coding of metadata can be accomplished in different ways. In Section 6.5 it will be argued that it follows the same principles as lexicons; however, this section will deal with possible structures of metadata representations covering simple attribute-value structures (AV) to complex hierarchical tree structures and coding these in XML.

6.4.1 Metadata structures

Existing metadata standards such as Dublin Core and OLAC provide an AV structures in which data categories are named and given a specific value. A flat AV structure can easily be stored in relational databases.

The relation between the attribute and value is not made explicit in an AV structure, but if the relation is made explicit, the result is not an AV pair but a triple. As in AV structures these triples contain an attribute (a subject) with a value (an object) but also a relation (a predicate). Consequently, every AV-structure can be represented in a triple, by explicitly stating the predicate. One example of this is the *Knowledge Interchange Format* (KIF) (see KIF (1998) in the bibliography) or the *Resource Description Framework* (RDF, Lassila and Swick (1999)), a standard issued by the W3C.

A hierarchical organization of metadata categories results in another structure, a tree structure. A hierarchy of metadata relations is used for example for inheriting information. Examples for standards organizing metadata in such a way are IMDI and TEI, as described before. A more detailed discussion of these structures can be found in Trippel (2004).

6.4.2 Storing metadata in relation to the resource

Another problem, not addressed before is the question where to store the metadata. Two possible solutions exist which share the expressive power and are hence formally equivalence. The first option is to store metadata in the context of the described item. Each described item receives the full set of metadata categories providing available information on this item. The second option is to store the metadata externally, each descriptor pointing to the described item. This means that all metadata are collected externally, either at a different location, i.e. a different file, or separated in an extra section, only connected to the items by pointers.

The external metadata storage serves for a better functionality in locating items that share the properties of the metadata, i.e. a grouping by metadata

features is already implied in external metadata representation. On the other hand the problem arises that some features can be relevant for only a small number of items, hence the granularity of these groupings can be extremely fine, resulting in small sets of items.

The integrated metadata storage is practiced in the TASX-annotation format, resulting in redundancy as in principal all available metadata that is relevant for each item has to be included with all items. For corpora it can be argued that hardly any metadata has to be coded with the individual segment as a lot of information is already implied by the tier level. However, for other data structures, such as lexicons, lexicon items are not grouped into tiers without reference to a microstructure, which should be part of the metadata description.

The problem of deciding for external or internal metadata representation can be avoided using a metadata inference concept, which also shows that the organization of metadata is strongly related to lexical concepts.

6.5 Metadata lexicon

Avoiding redundancy and providing consistency and completeness of metadata descriptions is a problem for detailed description of resources on multiple layers. To avoid redundancy while remaining consistent, a mechanism is introduced that is well known from lexicon development, i.e. inference techniques and default inheritance.

Up to now there have been two different connections of metadata and the lexicon, i.e. metadata for lexicons and metadata for resources used for lexicon creation. In this section a different, structural approach will be used that is not so much concerned with metadata categories at all but with the organization of metadata.

The lexicon microstructure, available in data a lexicon, constitutes a flat structure, a lexicon vector, with different categories, each having a content model. The same flat structure can be observed for metadata, as the structure can be interpreted as simple data categories without any further hierarchy. Consequently the listing of metadata definitions of Dublin Core, OLAC, TEI and IMDI already are sample microstructures for a metadata lexicon. Figure 6.4 shows the microstructure using data categories already mentioned in previous sections. Each metadata category receives a value.

The analysis of the different metadata sets shows that a mapping between different microstructures is possible. It is also possible that the application

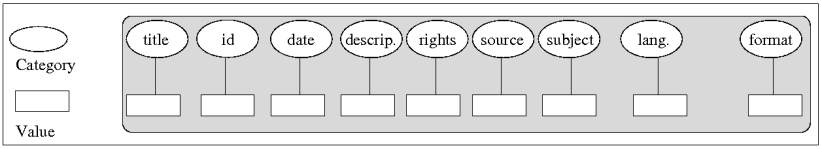

Fig. 6.4. Sample microstructure composed of metadata categories

of more than one metadata set results in redundant information, but allowing inference and a mapping from one metadata category to another resolves duplication. The idea is simply to inherit information from once defined properties. A simple example is the inheritance of personal information by having some sort of a representation of all persons involved with a resource and the appropriate data categories refer to this representation.

Trippel et al. (2004a) describe the consistent storage of metadata using inference techniques to derive metadata from other, existing metadata defined on other levels of the corpus. By this technique higher level metadata — such as authorship attribution of a resource — can serve as the default for even the smallest parts. The default overriding technique applied there allows the definition of deviating values, e.g. authorship attribution to an individual, instead of a group of authors.

For the organization of metadata, different hierarchical structures of metadata have been used. The metadata hierarchies do not necessarily mean that they exclude each other. The comparison of different metadata standards in Section 6.3.2 for example shows two organization principles, first a flat structure, as a list of metadata categories and second a classification into *content*, *editorial information* and the *instance*. Other structures can be thought of, e.g. according to subdisciplines, using the category or even subjective categories as 'importance'. Mapping data categories on a hierarchy results in an ontology such as the GOLD ontology described by Farrar and Langendoen (2003).

Just as lexicons metadata catalogs and descriptions of resources can have different macrostructures, i.e. they can appear in a variety of orders, such as ordered by author, institution, size, media, or some other data category from the microstructure. As the metadata organization is based on the same principles as other lexicons the metadata structure can be described in the same way as a lexicon description illustrated by Figure 6.5. The different metadata descriptions constitute a metadata lexicon entry with the microstructure vector, which is symbolized by the different labeled boxes. The individual data categories of the metadata descriptions can be classified and subclassified us-

ing the mesostructure, which is a hierarchy represented by the labeled field. The lexicon macrostructures are the different sorting options for the metadata lexicon entries.

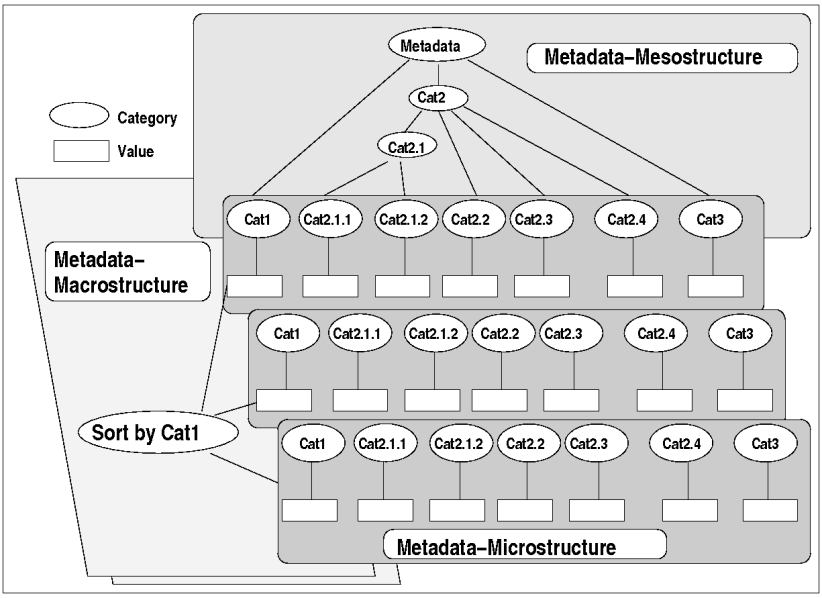

Fig. 6.5. Integration of metadata lexicon structures

6.6 Case study of a metadata lexicon

After employing on the metadata lexicon in such a theoretical way, a look at a case study can add to this section. For a case study, a corpus of the German fairy tale *Das Eselein* by Johann and Jacob Grimm was recorded with a semi-professional story teller on audio and video and annotated on 10 levels, including prosody, gesture, word. The whole session has 9 metadata categories, one of them serving as the container for 57 IMDI data categories. Each annotation tier/layer currently has 7 to 15 additional metadata categories, including identification categories for session identification, layer type and name. Additionally, categories used by an application such as font information are available, as well as data warehousing information, e.g. annotator description

and annotation process description. Some of the individual elements have one technical metadata item.

For taking full advantage of the *MetaLex* approach a number of preprocessing steps are needed in order to use the Metadata that is currently embedded in the corpus. These are:

1. Extraction of the metadata into a metadata repository. As the metadata needs to be transformed into a different data format for extensive use of non-XML technology based techniques, the metadata needs to be separated from the corpus itself. In practice this is done with a simple XQuery (Boag et al. (2003)) expression.
2. Transformation of metadata into attribute-value structures in DATR syntax, to be able to use DATR inference engines. This is accomplished by using the identifier of the superordinate structure as headword. The reason for using the identifier is to enable the reallocation of metadata with the annotation tier, segment, and subcorpus. The structure needs to be referred to by the headword, because the metadata are supposed to describe the characteristics of the superordinate structure. As the whole structure is not intended to be there, the identifier serves this purpose. This step involves transformation of a whole document and is performed with XSLT XSLT (1999).
3. Querying the metadata using a DATR inference engine. This allows global inheritance and default overriding, thus serving the purpose.

For the conversion into DATR format there are certain problems to be solved that are related to the original DATR syntax and the available format for the metadata structures. Especially treating reserved characters, which are different in DATR and in the XML structure, non-ASCII characters, etc. can be difficult if a tool does not support for example Unicode. A special problem is the representation of substructured trees in DATR's attribute-value formalism. The transformation of hierarchical tree structures such as IMDI metadata into DATR format involves *shredding* (Draper (2004), p. 335ff), in which potential terminal symbols of a tree structure are used to define the arity of a database table as DATR does not use tree structures but directed graphs for interconnections between different hierarchies. This could lead to ambiguities because the substructures of the metadata do not necessarily require identifiers for unambiguous identification if they are determined in the tree context.

DATR, however, is a lexical representation language that represents information not only in an attribute-value formalism but is a form of encoding directed acyclic graphs, to be more specific even feature structures. The representation of feature structures in the Lexicon Graph Model, however was

already shown in Section 2.3.19. Hence two tasks have been accomplished at the same time. Firstly, the structure of metadata representations was transformed into a formalism which is representable in the Lexicon Graph model. Secondly, using tools such as DATR inference engines and XQuery it was possible to actually harvest this Lexicon Graph, both using inference and querying the tree directly. A part of a sample metadata lexicon and a query is available (see Appendix 10 for details).

6.7 Summary

This section discussed the connection of metadata and lexicons, starting with the conventions used for corpus description and transferring the methods found there to lexicons. A technique of inheriting metadata from other existing metadata structures was discussed. Finally the organization of metadata itself was analyzed and compared to the organization of other lexicons. The result was that metadata itself can be interpreted as a lexical structure. Furthermore it was shown that the metadata can be used for different annotation layers even if is underspecified for each segment which could be used for lexical data extraction. How this metadata can be used for the extraction of lexical data by the definition of the microstructure of a lexicon is a pending issue discussed in the following two chapters.

7

Application: The Lexicon Graph in a concordance

This chapter shows how the consistent model for annotations and metadata description can be used for a lexical application, namely a multimodal concordance. This concordance is described in its functionality and requirements and can use the metadata description available in the annotation. The implementation of the concordance is also described.

Describing annotation and lexicon structures and extensively discussing the metadata, provides the grounds for lexical applications: for concordances. Section 5 discussed the formal relationship between lexicon and annotation, including a continuum of corpus and lexicon shown in Figure 5.1. Here, concordances are termed first order lexicons.

7.1 Concordances and concordance structures

Concordances are the connection of corpora and lexicons. Within a corpus, an item — such as a word, a sequence of words or an arbitrary string — is identified and presented in a context. For example, a concordance of Shakespeare's Sonnets in a printed format can be based on a wordlist of all words in alphabetical order which appear in these Sonnets. This is followed by a list of contexts in which these are used, either with a complete sentence, line, or a specified number of words before and after the target word. This is usually highlighted in the context or replaced by a symbol to mark its position. An example of such a concordance is Spevack (1973). A concordance with context information and highlighted keyword is usually called a *KeyWord In Context* or KWIC concordance.

KWIC concordances became popular with increasing abilities for language data processing in the 1960 and 1970s, producing a large list of contexts. Lay and Petrarca (1970) for example, describes a system called a *Double KWIC* concordance, which is a technique in which the interval of the keyword is kept constant, i.e. a constant number of words before and behind a word is taken. In the double KWIC concordance, the position of the keyword is changed in every list item, e.g. if the interval covers i words before the position n of the keyword and j words after the position of the keyword then this sequence is listed in all its varieties, i.e. the keyword is placed on all positions in the interval from i to j and the other words of the sequence are wrapped around according to their position in the sequence. The goal was to identify usage patterns.

Another form of a concordance is an index in which keywords are referenced as to their use in a text to locate information that is related to the concept of the keyword. Indexes are also lexicon like, having a keyword and page, verse or line numbers as a lexical value, see for example the index of this work.

A similar process was introduced by Trippel and Gibbon (2001) for signal based concordancing, in which an interval of a signal is located by a *key-string* in an annotation that is referring to the signal. A *key-string* in this context is something like a keyword, that is not restricted to single word units. As the procedures for arbitrary strings are the same as for words, the term *word* is usually taken for all strings that can be concordanced within this section.

As a first order lexicon, concordances have structures, as have other lexicons. The microstructure of a concordance is either the keywords followed by a list of contexts, or a keyword with some pointer to a context in a text.

The selection of possible words in a concordance is part of the macrostructure. In many versions of concordances this list of possible search words is restricted, certain words are usually not concordanced, especially very frequent words, among which are many function words. A stop-wordlist defines all words in a corpus that are not concordanced. For example, the index of this work was created based on a wordlist of which the most frequent words and the words from a closed word class were excluded. Additionally, a certain frequency of the use of a word was assumed to be relevant as a technical term to be included. Editing the list by the author was the final step in producing the list of words to be indexed. The same processes are used by some concordance providers.

Another part of the macrostructure is found in the sorting of the entries in a concordance, which is possible for example based on the headword or on the first word of the context. In modern concordance systems such as in

other lexicon systems, the macrostructure is not as important any more as electronic search facilities do not require sorting because only the matching lexicon entries are presented after a query.

The mesostructure of a concordance is rather flat as there is hardly any relation between the data categories of the microstructure besides the sequence. However, the rotation of contexts as part of the Double KWIC concordance mentioned before implies an explicit use of the sequence as a structure.

The implementation of these structures varies in the specific use in a concordance. Concordance applications or tools are sometimes called *concordancers*, samples of these are Wordsmith (Scott (996)), and SARA (see SARA (2002) in the bibliography), or for multimodal data, the *Portable Audio Concordance System* (PAX, see Trippel and Gibbon (2002)).

All concordancers have some common features:

- Concordancers process data in a well defined data format, such as pure ASCII text, where some special data is encoded according to a defined document grammar, for example for XML. If necessary, these formats have to be produced first using a normalization process;
- The keywords have to be made available to the system using an acquisition function, for example by creating a wordlist or a list of words without the stop words or less frequent words;
- The concordancing function, in which the acquired keywords are located in their context.

In early concordances normalization, keyword acquisition and concordancing were done manually, the normalization consisted only in legibility of the text sources by the person working on the text. The manual concordancing has some major drawbacks:

- Manual indexing is time-consuming, as a person has to do it, and therefore expensive
- For manual processing it is rather likely that certain occurrences are missed out due to human shortcomings
- For large corpora only subsets of linguistic units, such as certain words, were indexed, collocations and units that seemed less important or too frequent for indexing were omitted.

With the so called ‘electronic revolution’, concordancing has become a task of computer programs, enabling the indexing of all available linguistic units including collocations. Automated concordancing is much faster, so that corpora of several million units can be processed within minutes or seconds.

This depends on the data format, computing power and concordancing criteria. Concordancing systems perform normalization and keyword presentation either in real time or are compiled to be used statically from memory. Real time concordancing means that a processor is fed with a resource upon which the processor normalizes it and acquires the keywords. This keyword list is then presented to a potential user without prior storage of the results. Statically precompiled concordances can usually access the data faster, as all of the previous steps are performed once, and for all texts and words. The result is then stored for later use, which requires larger storage capacity. For every resource update, a recompilation is necessary.

7.2 Concordancing in a lexicon

The connection between first rank lexicons and other lexicons was mentioned before. For concordances, this relation can be made explicit, using them from within another lexicon.

The motivation was, and still is, to find examples and evidence for particular uses of structures such as (lexical) words in a text. An application of concordances is found in examples used by researchers to illustrate their theories. Examples are meant to give evidence for one particular structure. Consider the following famous Chomskyan example:

> Colorless green ideas sleep furiously. (Chomsky (1957), p. 15)

Though Chomsky used this example to illustrate that *grammaticality* and *meaningfulness* are independent features of sentences, this example still qualifies for a different phenomenon: Linguists construct examples to illustrate structures that are under consideration. Nowadays, Chomsky's example is used rather frequently[1], so a corpus of linguistic texts would show a combination of *colorless* and *green* as perfectly possible. At the time of Chomsky's writing this was a pathological example — and it is still used exactly for this purpose— far from natural language use and this example is used as an anecdote. Nevertheless, the use of constructed examples constitute the problem that ubiquitous examples sometimes are neglected while awkward or rare examples are discussed extensively.

With the use of concordances, the coverage of frequently and rarely used expressions changes. Evidence can be taken from original data supplied for

[1] A search for a combination of all these words using the Google internet search engine on May, 26th 2003 returned more than 2300 hits.

other purposes, i.e. data that was not created for demonstrating a structure but that was supplied independently of the structures such as books and articles. Nevertheless, it is still possible to use rather unlikely or even *singular* examples that are recorded erroneously. To prevent isolated examples, concordances provide the use of ubiquitous instances by showing all instances of a particular linguistic unit at once, possibly with statistical information on the number of findings but at least implied in the number of references.

A necessary condition for the use of corpora is what Sinclair (1991) calls to 'accept the evidence' and 'reflect the evidence'(p. 4). This states that researchers need to be willing to use the results of a corpus based analysis even if they do not reflect their theoretic background or their interest in special cases. In addition to that, reflecting the corpus implies that the majority of findings in a corpus need to be reflected in the results that are discussed.

Analysis built upon statistical grounds with units taken form corpora does not mean that the researchers comprehension and intuition can be neglected. In fact the contrary is true: As the analysis can be accomplished automatically, the amount of data that is available grows immensely. Not only can singular or isolated examples be investigated for explanation but also large quantities of structures and even classes of structures where examples deviate from each other structurally only in minor aspects. For the detailed analysis, every linguistic unit can be taken into the center of analysis.

The researchers intuition and comprehension is therefore necessary to monitor the automated results and select a subset for further explanation and research. Even deviations from former intuitions and rules require a most thorough analysis and therefore provide grounds for further research. The representation of meaning in dictionaries is one of the issues, both being part of the expert intuition and corpus evidence:

- Different meanings are represented as discrete, though there are sometimes only gradual distinctions and semantic groupings of meanings
- The representation of meaning does not necessarily reflect the language users distinction and categorization.

The 'real world' findings can be related to the lexicon. Dictionaries containing lists of words, complex lexicons, they can all use the examples from the corpora with the assistance of a concordancer. Based on an annotation, for example according to FrameNet (Johnson et al. (2002)), ambiguities are not solved and a lexicon could show two different frames, distinguishing the meanings. The frames are developed on the basis of annotators intuition upon which the corpus linguistic analysis starts; consequently both intuition and machine analysis are needed.

For lexical analysis another problem is significant, which is the problem of citation forms. Sinclair (1991)(p.8) mentions that there are lexical units where the distribution of meanings is distinguishable by the concrete grammatical form, for example by an inflection. Therefore each distinct form of a word could be regarded as lexicalized and hence be recorded as a unique lexical item. Otherwise it seems desirable to group items and allow for inference of new forms because by grouping, it is possible to cover less frequent items. In the case of grouping a lemmatization seems appropriate where a lexical item is classified by an abstract lemma which takes the place of each different form as a lexicalized item for querying the corpus.

Lemmatization is relevant as well for the selection of a lexical key, keyword or headword in the lexicon to access a lexicon entry. This is discussed in the Section 2.2.3 .

In multimodal applications the context and the actual rendering of a sign are extremely important. The description of gestures, using for example the CoGesT system (see Section 5.6.1), allows a variety of different movements in context. One problem is that the transcription of a movement is not unique, i.e. sometimes more than one possible transcription can be found that is suitable for the description of a gesture. Another one is that a gesture cannot be separated from its context, i.e. gestural constituents influence each other. And lastly, in a multimodal setting the different modalities and environmental conditions are reflected. This results in the definitive need to access the original setting as well as possible when analyzing or describing different modes. Even from within a lexicon, using a concordancing function access to the actual contexts has to be provided.

It seems to be appropriate to have a look at the purposes of concordances. Starting with concordances for textual corpora the purpose can be described in the fields of:

- Education,
- Information lookup/retrieval
- Dictionary and grammar building

In educational contexts, authentic uses of lexical items can be analyzed and studied with the possibility of verifying a particular use or deciding for an option if more than one exists. For information retrieval, the concordance is used for finding information on a particular subject, based on a word — search engines on the World Wide Web are used for this purpose for example, as well as archives and indices of books. For dictionary and grammar building, the lexicographer searches for authentic structures which they want to include in a lexicon or grammar, quoting a selection as examples for rules and structures.

More complex requirements appear in non-textual modalities. In the spoken language field, for example, annotations are produced on more than one tier (see Section 5.6.2). These tiers are related to each other in a corpus by a common time line. Annotation on source identical data or standoff annotation is an equivalent model, for example when using two different document grammars that cannot be unified. This has been discussed in Section 5.4. For concordancing, the distribution of relevant contextual information on different tiers or even more on distributed annotation files results in additional needs. However, these annotations are available in linear order, either by the time line referring to *absolute time* or by sequential enumeration of characters according to a time line referring to *category time* (see Carson-Berndsen (1998), p. 68). Both are formally consistent with *Annotation Graphs*. Fig. 7.1 illustrates the use of *category* and *absolute time* for the same utterance by mapping a signal based tier annotation to a score annotation of the same utterance based on the textual level, both mapped to a time line. For the earlier a category time line is used, for the latter an absolute time line.

When different annotation levels are separated into different layers, the problem of overlapping sequences can be neglected. The relation between the different layers can be drawn using a time calculus, which is described in Section 8.2.

Another use of concordancing is in the evaluation of corpora. Error prone manual annotation is still used for the creation of corpora and probably will be in use at least for basic segmentation and initial annotation before an annotation can be enriched using a lexicon (see for example Witt et al. (2000)). Concordances can be used for evaluation and correction of these manual annotations.

- An error on the syntactic level can be checked by selecting all occurrences of one error if this error happens to be found at least once; every instance of a possible error is another possible error and a complete list of all occurrences will be a way (manually) to check all instances. If the rules are sufficiently obvious, for example if an annotation unit cannot be correct in a context that can be specified then this can be used for automatic error replacement.
- If certain contexts are sufficiently well defined, the annotation can be searched for deviating patterns, that can either be checked manually, or if they are similarly systematic, automatically corrected.

Locating the errors using a concordance for corpora provides an efficient tool for manual correction of corpora by a researcher. But concordancing re-

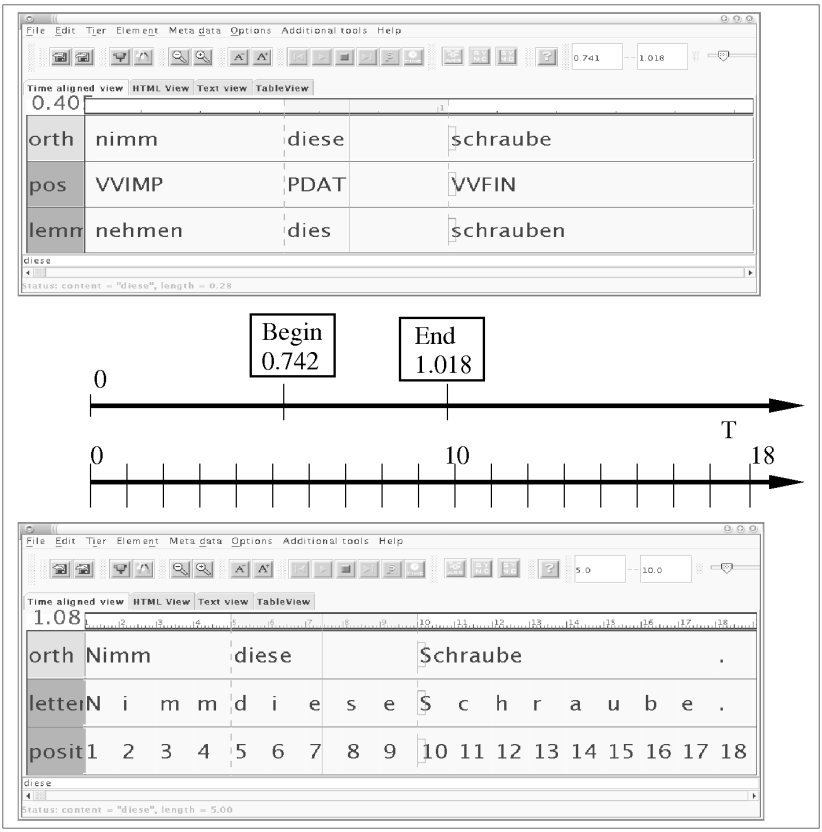

Fig. 7.1. Relation of *absolute* and *category time* structure. From top: annotation of the utterance "Nimm diese Schraube" (Take this screw) according to the acoustic signal with orthography, lemmatization, POS; absolute time line, category time line, annotation in category time (enumeration of characters), textual source

sults can also be used for locating patterns by a tool, enabling automated correction.

7.3 Problems of modern concordancing

With the extension of concordancing to arbitrary corpora new problems arise that are not answered by previous tools. These are the problems:

- Multimodality
- Connection to higher order lexicons
- Treatment of multi tier scores
- Consequences of the use of the lexicon graph model
- Result of the use of metadata
- User interfaces.

Including other modalities than text has consequences to traditional concordances. Although text and signal based corpora can be seen as being related to time (see Section 7.2 above), this leaves out the problem of including the signal. It would be possible to produce a timestamp output with a concordance result and then 'forwarding' the signal to a given time-stamp but forwarding and rewinding is not the intention of a concordance that wants to allow for accessing the instances. A signal instance in this context is a part of the signal that has the same time stamps as the associated annotation unit on the selected interval. This signal interval is part of the original, complete signal and therefore needs to be selected with appropriate tools which form part of a concordancing system.

Another problem is related to the modalities is the relation to higher order lexicons. The question is especially striking as the lexicon graph model allows the inclusion of arbitrary modalities. The use of these other modalities for accessing a lexicon seems to be related to signal recognition and would be out of the scope of this work. However, what is should be included here is the connection of the concordance to another lexicon, for example using the concordance as a macrostructure to the lexicon, i.e. enabling a user to go from a string found in the concordance to a corresponding lexicon subspace in the lexicon graph.

Resulting from the traditional multi tier score representation of annotations for multimodal corpora is the question, how to concordance a multi tier score. It is necessary to not only provide for a *word*, or more general *annotation unit* selection, but also for the selection of a specific tier where this string should be found.

The lexicon graph model further allows the use of all lexical items to be used as headword. Traditionally, concordances access a corpus on the word level, i.e. the occurrences of words are indexed, at least according to lemmas. But for different linguistic units indexing by word or lemma is not sufficient, for example when querying for sentence structures in English to find out if the SVO structure is annotated correctly for corpora of English. Lemma based access is not sufficient for this purpose. Another question could be the occurrence of specific syllables, prosodic patterns, etc. The lexicon graph model

also allows the restriction to a certain microstructure which corresponds to annotation levels. The result is that a selection of the annotation layers needs to be taken into account.

As the annotation and especially the individual annotation tiers are classified using the metadata described in the last chapter, it is possible to use this metadata for classification purposes. This becomes even more necessary in large corpora which may include a number of different texts and signals. Each can be associated with several layers of annotation which can be in different languages, on different topics and in different scenarios. A concordance system for these corpora needs to provide for varieties within a corpus by enabling the selection of annotation units and layers and subcorpora according to associated metadata such as language, scenario, and text.

The many requirements for the concordance show that there are a lot of options involved, restriction to tiers with certain characteristics, annotation units, subcorpora. Additionally, the interval, i.e. the size of right and left context of the found item, needs to be specified. For a person to use the concordance system an appropriate user interface needs to be implemented that allows the user to specify these options. Ideally, initial choices influence later choices, for example by presenting only available layers for a subcorpus if a user selected a subcorpus by some criterion, e.g. by selecting a language, or by presenting only the different subcorpora that have a particular layer if a layer is selected first. The complexity of such a user interface can be rather high, hence criteria such as ergonomics and aesthetics are not discussed in this work.

7.4 Requirements for a multimodal concordance

The function of a multimodal concordance is described as a mapping of the annotated digital signal into an enhanced KWIC concordance $f : CORPUS \rightarrow < KWIC, SIGNAL >$, where KWIC is the enhanced $KWIC$ concordance and $SIGNAL$ is the signal output with possibly different renderings, such as a spectrogram or a sound. The enhancement of the KWIC approach is, that the output is not only linear on the same annotation tier, but the context is extended to the other, parallel tiers.

The functionality of the multimodal concordance can be described in the terms already used for the lexicon description:

Portability: the concordance system, just as other language resources, is supposed to be portable as described in Section 2.1.4. In the context of a

system this means especially that a system needs to be portable to different generations and versions of software and operating systems. This is sometimes also called *interoperability*.

Standardization: related to portability is the consistency of use of data format and programming languages, which should rely on standards. The standard conformance of the involved functionality is required not only for portability purposes over systems but also for archiving purposes.

Free access structure: just as described for the lexicon graph, the concordance has to allow different data categories for accessing the corpus, i.e. based on all annotation units that are provided in the corpus. This also enables a multi functional use of the concordance.

Open architecture: extensibility of the concordance tool requires an open architecture to allow the addition of additional functions and tools. An example could be the phonetic analysis based on a selected part of the corpus.

Easy extensibility for new corpora: Corpora are often not closed but constantly being worked on, such as editing annotation layers, new sessions, signals and data. This has to be possible and easy.

Easy connection to other lexical resources: for the connection of this first order lexicon to higher rank lexicons the connection to resources is essential.

Extension of the context to parallel tiers: Other than text based concordances, the multimodal concordance has to find a way of including the parallel tiers.

Location independent access to the data: as fieldwork corpora and research is carried out on a worldwide scale the access to the data should be independent of the users physical location.

Though the list of requirements is lengthy, the user interface was left out for the purpose mentioned at the end of the previous section: aesthetics and ergonomics are to be left out of this work, and the options of accessing the system are mentioned in the list explicitly.

The annotation is in the TASX XML-binding of the annotation graph model, possibly annotated on multiple tiers. The tiers and annotation sessions are annotated with additional metadata.

7.5 Designing a multimodal concordance

A multimodal concordance based on the requirements mentioned in the previous section is a complex system. This section discusses the design of such

a concordance system. Figure 7.2 shows the lexicon graph that needs to be extracted from an annotation graph based on the headword, which is the keyword in the concordance.

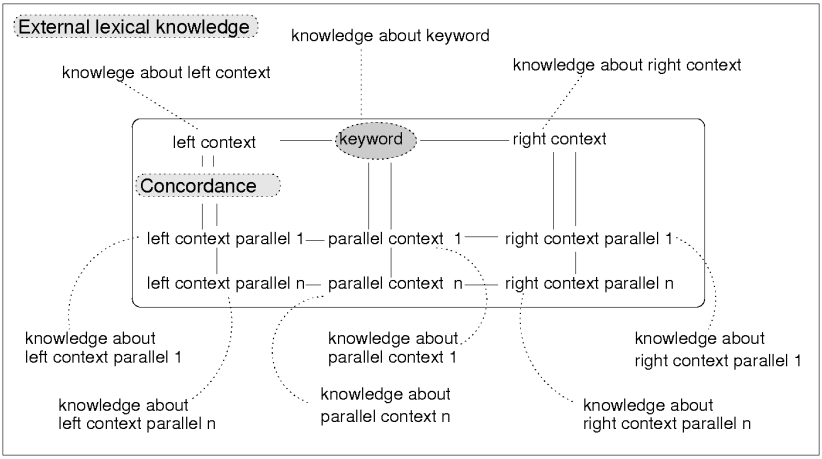

Fig. 7.2. Multimodal concordance as a Lexicon Graph

Two classes of contexts can be distinguished, namely the contexts which are of the same type as the keyword, i.e. they are from the same annotation tier called left and right contexts in Figure 7.2, and the ones that are not from the same tier, i.e. from parallel annotation tiers, the instances called *parable context*s in Figure 7.2. The parallel contexts also have contexts of the same type, which are at the same time also parallel contexts of the keyword's left and right contexts. All of theses lexical items can be related to some external lexical knowledge, for example to an external lexicon containing information on wordclasses, inflection tables, etc. The inner box of Figure 7.2 represents the concordance without the relation to external lexical knowledge. The parallel context, however, does not have to be letter based, also pictures or signals can be contexts of the keyword in this setting.

The Lexicon Graph structure of the multimodal concordance is obvious, but the question remains if this lexicon can also be analyzed in the terms of the structures used for the description of other lexicons as in Section 2.2. And indeed, it is possible to describe this concordance in the same terms.

Microstructure: Data categories for the concordance are left and right context, and for each parallel tier one parallel context. These parallel contexts have a subcategory of left and right contexts again.

Mesostructure: The link to the external lexical knowledge, either made explicit or implicit, is part of the mesostructure. In the case of more than one parallel contexts, e.g. due to a different segmentation on the different tiers, the order of parallel contextual items by their timestamp is also part of the mesostructure. By the classification system of the tiers, metadata descriptions are also available.

Macrostructure: The macrostructure is based on the headword, which is supposed to be freely selected from the annotation items. If the annotation items contain semantic information, the concordance can be an onomasiological lexicon. In the case of a query by some sort of orthographic form or lemma, a concordance based on a Lexicon Graph can also be a semasiological lexicon. The primary sorting criterion is by the class of the headword, though this can be a semantic class, numeric value of a concept in an ontology, etc., as long as it is in the original annotation. The secondary sorting strategy is the position in the annotation with reference to the time line.

This structure also fulfills the requirements to modern lexicography as discussed in Section 2.1.6. This is true especially as the data is modeled according to linguistic units, the data is available according to the researchers work flow, the relation between resource and description is transparent, by extending the corpus the lexicon is enlarged, and multimedia can be used. The question of user interface is a question of the implementation, as is the backchanneling, which is reduced to the correction of the source data.

As a tool for accessing language resources a multimodal concordance also creates a higher level of portability of resources as discussed in Section 2.1.5. Other aspects of portability of the concordance workbench are part of the implementation as defined in the requirements above.

The first decision in the design process of the concordance system is if the concordance is to be a static or dynamic system. This has immediate consequences for the requirements mentioned above. A static concordance is usually faster in the access and the volatility of the data can be avoided by finding a permanent storage device. This could be advantageous to portability. However, the permanency of a precompiled concordance is a problem for the extensibility: for each change in the corpus, the static concordance would have to be regenerated. Having the whole system on a permanent storage device would also not allow for location independent access as this would mean

that the data had to be distributed, before it was accessible. A solution would be to distribute the data via the internet. In this case, the system would be accessible from all over the world, i.e. location independent. The data could be provided centrally, hence maintenance, backup and access would be available on some server. Central server architectures could be possible for a static concordance system, but the extensibility issue still remains unsolved. This would be much easier using a dynamic concordance, where the concordance is processed during runtime. The available corpora could be used from the start, and extension would be possible by adding corpora. Server side setup would also allow low resource requirements on the user client. The client would not be required to do the main computing, this would be done on the server. As the dynamic concordance is the prerequisite for a static one, which is a stored version of all possible queries of the dynamic system, a network based dynamic concordance system is the method of choice for the requirements mentioned before in Chapter 5.

A client-server architecture also allows a concept which is portable to different operating systems by using for example standard web browsers as the user interface. This also allows connection to other lexical resources, by using hyperlinks as points of access to other lexical resources. It is possible to create a link from each lexical unit as a query to a lexicon graph. As a consequence, a web-based client-server architecture can be taken to fulfill the extensibility, location independence, and part of the portability requirements.

The other requirements are based on the procedures and software needed on the server side and are independent of these.

The procedures needed for the concordance are firstly the lexicon generation function $f_{lexgen} : Annotation \rightarrow Lexicon$, where the lexicon is the list of annotation units, such as the word list, lemma list, etc.

This of course requires an existing annotation. The lexicon annotation itself is a function $f_{anno} : Signal \rightarrow Annotation$. Creating corpora has been discussed in Section 5.5.

The concordancing functions are the inverse of the lexicon generation function $f^{-1}_{lexgen} := f_{conc} : Lexicon \rightarrow Annotation$. This mapping locates the target unit in the annotation and hence allows accessing the right and left context. It is, however, not necessarily a function any more, as a lexicon contains each annotation item only once, while the annotation may contain it more often. Hence this mapping is a $1 : n$ mapping.

Based on the result of the concordance, corresponding signals have to be extracted in a signal selection function: $f^{-1}_{anno} := f_{sigselect} : Annotation \rightarrow Signal$. As this is a signal processing module

These functions are still the same as a single signal based annotation tier, described by Trippel and Gibbon (2002). Adding additional annotation tiers and wanting to access them results in additional functions.

The first of these additional functions, which is an addition to the lexicon acquisition function, is the classification function $f : class_{unit} : annotation - unit \rightarrow unitlist_{class}$. This function allows the creation of a list of annotation units only with units belonging to a specific class of units, such as words, lemmas, etc. The classification of the annotation units is given by metadata for the annotation units, as discussed in Chapter 6. The classes of the metadata can usually be inferred from the classification of the layer of which the unit is a part, but in general the class of each annotation unit would have to be checked by this function.

The second additional function is based on the extension of the context, hence extending the concordancing function. The location of the annotation unit is not sufficient for granting access to the context in a multi-tier annotation, as the sequence only allows the access to the left and right contexts. Based on the time stamps of the found annotation unit, parallel annotation units have to be identified. By the use of their metadata, these can again be classified and ordered in their classes, for example rendered in different tiers again. This function is a function $f : annotationunit_{start,end} \rightarrow parallelunits$.

A more general additional function is based on a reduction of the concordance search space by selecting a specific subcorpus, for example of a certain language, area, or time period. This is a selection of specific recording sessions, also based on the classification by the metadata, as the classification of the annotation units.

The structure of the multimodal concordance is illustrated by Figure 7.3. The lexicon acquisition module here is split into two areas, the unit classification and the unit listing, both referring to the annotation units. The latter process is based on the classification. There is also a classification component in the concordancing module, which is part of the context identification. This context identification adds to the unit identification to allow the inclusion of the additional annotation tiers. As the segments on the other annotation tiers are not necessarily sharing the same time stamps as the selected unit, this context selection needs to be specified for a time interval.

The result of the concordancing is a part of the annotation, possibly having the same data format. In fact, it could be used as a subcorpus itself, being the input for a similar concordancing-lexicon process, hence a similarity between the output and the annotation is indicated.

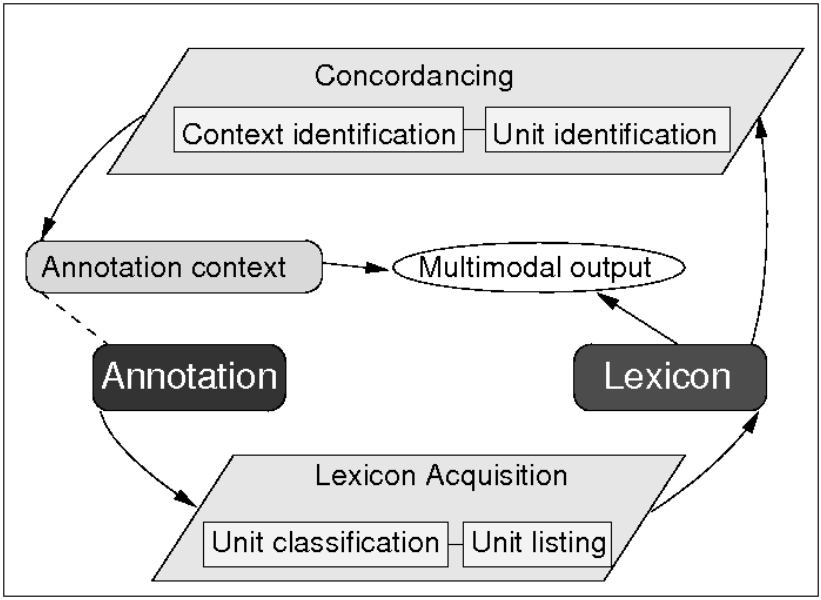

Fig. 7.3. Relation of an annotation and a lexicon in a multimodal concordance

Both, lexicon and annotation can have a multimodal output for the user, based on the original signal or some other form of multimodal rendering, e.g. by a speech synthesis system. The multimodal output requires additional signal processing and signal selection programs, which are not part of this model.

Figure 7.4 shows a possible flow diagram for a concordance system, which is based on a classification of subcorpus, annotation layer, and annotation unit. This classification is used in the corpus query, which is the annotation-unit-in-context lookup. Ideally the order in which the classification is implemented, should be arbitrary. This is the reason why the classification module are on the same level, but in practice the selection of the subcorpus, annotation layer and annotation unit are usually presented in this order.

7.6 Implementing a multimodal concordance

The corpus for the multimodal concordance implementation is based on the TASX format and contains several subcorpora:

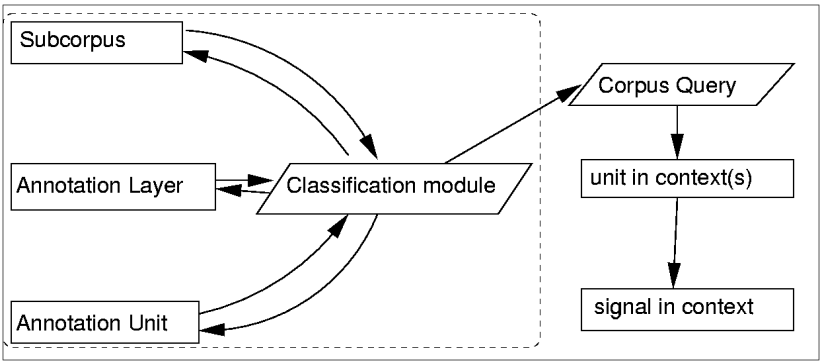

Fig. 7.4. Flow diagram of a multimodal concordance

- ModeLex-Corpus: Multimodal corpus with annotation on several tiers, including word-level, lemma, tones, phonetic, sentence, gestures
- LeaP-Corpus: Audio corpus with annotation on several tiers, including word-level and tones
- Ega-Corps: Audio/Video corpus with annotations on several tiers in the language Ega, which is an endangered language spoken in Ivory Coast.

The concordance is implemented using standardized techniques, most from the XML paradigm. The reason for this is that the interoperability of XML documents and XML based methods is very good, hence the system is very portable. The programming languages and tools are:

XQuery: in the context of querying XML documents the World Wide Web consortium works at the specification and standardization of the XQuery language (Boag et al. (2003)). Though the standardization process is not finished yet, the emerging patterns and principal functions are stable. The concordance relies only on the stable part of the draft standard. By using the standard XQuery language, the reusability of the code for other corpus based processes can also be increased.

XSLT: the XML Stylesheet Language – Transformations (XSLT) provides a functional programming language in the XML paradigm, which allows a powerful transformation of XML documents into other structures, such as HTML for web rendering. The transformations from an XML document which is the output of an XQuery are easily transformed into HTML for the web-GUI.

Perl: as a simple programming language for rapid prototyping Perl is used for the implementation of the CGI-interaction for the server based con-

cordance processing. With the help of Perl scripts the specific XQueries are generated and sent to the system for further processing.

Tamino: which is an XML database by Software AG (see Tamino (2003) in the bibliography). The query language for the database system allows XQueries to be submitted to the database. For the use with the programming language *Perl* a Perl API (cf. Hell (2003)) was developed and used that allows sending of XQueries from a Perl script to the database and process the resulting document, for example by sending it to the web server.

Java: for the processing of the audio signal and the cutting out of signal chunks, a tool in Java was used. For video, a similar tool was used based on the Java Media Framework. However, the implementation was based on the audio signal only.

Praat: for additional analytical phonetic processing functions of the phonetic software *Praat* were defined. With the help of Praat, the signal chunks extracted by the Java tool can be analyzed creating oscillograms, spectrograms, etc.

Apache: the web-based communication uses the Apache web server.

The user interface was designed on the basis of HTML web pages with HTML forms for the user selecting options. The user is guided through the following web pages:

1. *Subcorpus selection:* the user is asked to select the metadata category on the recording session level according to which s/he wants to restrict the search space. Only data categories that are in the corpus are presented to the user for selection. The list is created by a static XQuery, which is represented by the following pseudocode, the capitalized resembling the corresponding names of the TASX element names, also indicated are the embedded structures.

```
foreach SESSION/METADATA/DESC
return NAME
```

2. *Subcorpus selection:* based on the selected metadata category the user is asked to select a value for this data category. Only values that are in the corpus are in the selection list. The list is created by an XQuery that is generated in a Perl script to contain the value of the previous step. The pseudocode for the query is with the same conventions as above:

```
foreach SESSION/METADATA/DESC
where NAME = specified by user
return VAL
```

3. *Layer selection:* the user is asked to select the metadata category according to which s/he wants to select the target layer or tier. Only data categories that are in the corpus are presented to the user for selection. The list is created by an XQuery that is generated in a Perl script to contain the value of the previous step.

```
foreach SESSION
where exits (METADATA/DESC
             where NAME = specified by user and
                   VAL = specified by user
            )
do
     foreach LAYER/METADATA/DESC
     return NAME
```

4. *Layer selection:* the final search space selection for the lexicon acquisition is the value for the layer-metadata category. Again, only existing categories are presented by this query, which is generated by a Perl script. The result is transformed using XSLT into an HTML file.

```
foreach SESSION
where exits (METADATA/DESC
                 where NAME = specified by user and
                       VAL = specified by user
            )
do
      foreach LAYER/METADATA/DESC
      where NAME = specified by user
      return VAL
```

5. *Search string selection:* the result of the previous transformation is a lexicon which was created upon the specified annotation layers of a subcorpus. The corresponding query is again given in pseudocode:

```
foreach SESSION
where exits (METADATA/DESC
       where NAME = specified by user and
             VAL = specified by user
            )
do
      foreach LAYER
      where exits  (METADATA/DESC
            where NAME = specified by user and
```

```
                    VAL = specified by user
                     )
      do
        foreach EVENT
        return EVENT (content and time stamps)
```

The user can then select a lexical item for searching the corpus subcorpus and specify the time interval around the search word. This XQuery is maintaining the restriction to the subcorpus and is generated using a Perl script. The query is generated in a way such that the result is indeed a TASX file again, which is the part of the original annotation in the given interval. The pseudocode of this query is the following:

```
foreach SESSION
where exits (METADATA/DESC
                  where NAME = specified by user and
                        VAL = specified by user
            )
do
      foreach LAYER
      do
        foreach EVENT
where EVENT is in user defined time interval given
                    by time stamps
        return EVENT (content and time stamps)
```

6. *Analytical function selection:* based on the transformation of the annotation file into the HTML format, a form is presented to the user in which s/he can select further analytical functions in Praat for each context.

The transformation of a TASX-file into HTML proved to be a complex task. The reason for this is that for the HTML file the absolute time segmentation was not maintained, so the HTML table construct could be used. Hence, the transformation is a mapping from the absolute time to category time. To accomplish this, the number of different time stamps is calculated, each different time stamp providing a new table boundary. The width of each individual table cell is set by counting the number of timestamps inside of the interval given by the start and end timestamp of the interval. In the case of an empty interval, i.e. no segment is specified on a layer within a given interval, this segment is inserted as an empty segment in the HTML file. This irreversible transformation is only allowed as the HTML file is only a rendering format for the GUI and not created for analytical purposes.

7.7 Maintaining a multimodal concordance

As the concordance is modular, the individual modules can be maintained individually. A change in the application might require a change of the API but the XQuery expression and modules can remain constant.

Changes of the user interface do not affect the concordances functionality. Nevertheless, changes to the user interfaces are the most obvious adjustments, for example restricting metadata selection or implementing interfaces where only a limited number of restrictions to a corpus can be performed. In this case for other queries a new interface would be needed, which also requires adjustment of the XQuery generation.

Another, more frequent change is the extension of the corpus and the use of different metadata standards. However, other metadata standards in the corpus result in a query that is based on different metadata. As the query is generated based on existing categories, the program suite does not have to be changed. Extension of the corpus is made possible by storing additional corpora in TASX format in the database. As the concordance is dynamic, the changes will be effective immediately after the storing process is completed.

The concordance system was tested for its functionality, using two methods:

1. Manual concordancing a small example corpus. This method was used for initial functionality tests, especially to test if the time-interval selection was working properly as the comparable software does not provide for time-intervals but annotation unit-count intervals.
2. This concordance system was compared to the PAX-System (Trippel and Gibbon (2001) and Trippel and Gibbon (2002)). For this a larger corpus could be used and the number of occurrences could be compared with limitations to the contexts as the PAX-system provides only annotation-unit intervals. Additionally, the selection of metadata in the PAX-System was very restricted, only identifiers for subcorpora and annotation layers could be selected as well as words. The new system is much more flexible in the selection of metadata.

A formal evaluation of the concordance system is still a pending issue, as the criteria have to be defined. Some of them are:

- performance: How long does it take to query the corpus. For this purpose a comparison with the Perl based PAX system could be used if the corpus is identical
- scalability: What happens if the size of the corpora increases

- portability: How long does it take to port the system to other platforms and software environments

As this implementation was intended as a proof of concept study, the formal evaluation was left for future stages of the development.

7.8 Summary

Concordances as an obvious connection of corpora and lexicons were described, starting with the description of the concordance as a lexicon structure and as a function in a lexicon system.

A multimodal concordance system was designed and implemented, as well as evaluated, based on multimodal corpora containing audio and video data with multi tier score annotations. For this implementation the metadata was extensively used to define the macrostructure of the concordance, seen as a Lexicon Graph with all possible lexical headwords. The problem remains how to create a higher order lexicon from this, allowing for generalizations. This will be described in the following chapter.

8

Lexicon Graph and Annotation Graph in generating lexicons

With the multimodal concordance described in the previous chapter, a first order multimodal lexicon was described. This lexicon was generated, i.e. automatically created by a computer program, from multi-tier score annotations. The question raised from that is how to create other lexicons from multi-tier score annotations, that is lexicons of a higher order in the lexicon continuum shown in Figure 5.1 and discussed at the beginning of Chapter 5. In this chapter the question of corpus based lexicon generation is approached, which is also the question how to go up in the lexicon hierarchy mentioned at the beginning of Chapter 5. The method of choice is to look at some procedures to automatically generate corpus based lexicons first and then try to extent the procedures using a time calculus and inference rules.

Three types of lexicons are discussed here:

1. Simple corpus based lexicons, i.e. the extraction of lexical information from a corpus, as described by Daelemans and Durieux (2000).
2. Multi level corpus based lexicons, i.e. corpora which are annotated on different, non-context-free unifiable linguistic levels, as they exist for signal annotations. The generation of these lexicons is based on a time calculus.
3. Inference lexicons, i.e. generating lexicons using some lexical knowledge.

There are several reasons for creating lexicons based on corpora which are fulfilling some requirements discussed in Section 2.1.4.

Coverage: relying on introspection alone does not give a sufficient criterion to describe the coverage of a lexicon, i.e. which part of the language is covered by it.

Data collection according to work flow: as the work flow of a lexicographer is based on the order of the corpus, if working corpus based, this is trivial.

Extensibility: in a generated lexicon from a corpus, extensions can be produced by extending the corpus, i.e. including more data in the generation process.

Multimedia integration: by including multimodal corpora, which are based on video and audio data, and by applying a concordancing functionality as described in Section 7, the integration of multimedia events is also achieved.

The integration of lexical knowledge increases the coverage of a lexicon to include acceptable information not contained in the corpus. This enables the reduction of size required by a lexicon by omitting redundant information. The latter, however, could be left out of the discussion, as the size of a textual representation of a lexicon is small in comparison to, for example, multimedia events. Nevertheless, in certain environments, such as mobile applications, size is still a major requirement, though hardware capacities are growing there as well.

At the end of this Section, a brief discussion of lexicon data formats is added as this is related to the kind of information that can be gathered using the methods discussed.

8.1 Generating simple corpus based lexicons

A corpus based lexicon can be created easily with some simple methods. A simple corpus in this context is not one which is simple in its annotation scheme, or one which includes only linguistically well understood phenomena. The corpora taken into consideration here are the ones where all information and annotation is given in one sequence, i.e. on one annotation level, not using multiple tiers or multiple annotations for the same data.

An example of a simple annotation is a text in Latin script, in which words are delimited using a space character. A more complex example is an annotation according to a complex document grammar such as the TEI document grammar for texts in a document centered corpus format (see Section 5.1).

For a simple corpus based lexicon, different processes are possible. However they are all based on the annotations that are available, such as spaces or specific markup. More complex lexicons and learning strategies are discussed in Daelemans and Durieux (2000). From the structural point of view they do not provide further complications to the lexicon.

Starting with lexicographic processing of a text, a wordlist can be created as described in Section 2.3.1 which is a listing of all words within one text. Though this seems to be redundant, it covers a bit of lexical information which is usually not made explicit and which is either the source of a word or a lexical relation *is contained within*. These wordlists have been used by lexicographers as a list of word that needed to be included in a lexicon, and serve as the basis for their own manually produced lexicon entries.

Very similar to wordlists are *frequency lexicons*, which are wordlists with an additional specification, i.e. the number of occurrences in a given corpus. This information cannot be produced by a lexicographer on the grounds of introspection without a corpus.

For completion sake a sample script creating a sorted wordlist is available (see Appendix 10) with a corresponding representation of a frequency lexicon. It is an XQuery function which creates a wordlist with additional frequency — both absolute and relative — based on a TASX-file (see Section 5.1.1.2) which has a layer with the metadata AV pair `layer name / words .`

A word frequency lexicon is the basis for the inclusion of contexts and finding rules deduced form a corpus, such as a certain word occurs frequently, sometimes, seldom, or never in a certain context. These lexicons can be used for the creation of collocation dictionaries and of syntactic parsers, for example if it is known that after a certain word, such as a determiner, only a word of a certain syntactic class can follow.

8.2 Temporal calculus based lexicon synthesis

In the domain of signal processing, different linguistic annotation levels are represented on different tiers in a score. The segments on these

tiers do not have to share boundaries. As the segmentation is not unique, they cannot be integrated into one tier. To relate the different tiers with each other, a variety of methods can be used. In the previous Section the TASX format was used to create a wordlist which was a process on one tier only. Using the other annotation levels as well can be accomplished using time logics. The time logic based lexicon generation presupposes a sufficiently large corpus, i.e. saturated in a way that the linguistic units to be investigated and the relations appear sufficiently often (see Section 5.3.1). The lexicon induced from a corpus in this way does not extend the coverage of the corpus itself but enables the extraction of existing information that is hidden from the analyst or that is needed to be formally described.

Annotations are structured linearly, either by referring to an *absolute time line* — that is the time line of a signal — or a *category time line* (Carson-Berndsen (1998)), which is the linear sequence of annotated units, for example enumerated character strings as a baseline for textual annotations.

The relation of different annotation levels is not determined by the synchrony of annotated units, as the segmentation may be different on all annotation levels. Therefore, there needs to be one reference time line that is fine enough to distinguish all differing segmentations but allows for the serialization of annotation borders in a way so that it is possible to say which timestamp is larger, smaller or equal to another (for the problem of identity see also Section 5.6.2). The restriction of the granularity is also relevant for an absolute time line, as the granularity depends on the technical means for measuring.

8.2.1 Time calculus based lexicography on multilevel annotations

The idea of a time calculus based lexicography is derived from the understanding that some annotation tiers are related to each other but this relation can not be easily described without referring to the score (see Trippel et al. (2003)), i.e. using a visual description of a systematic relation such as the segments on the morphological annotation tier show similarities to the syllable annotation tier. The reason why only a similarity can be assumed is that the relation of different annotation levels is not determined by the synchrony of annotated events. For example Morrel-Samuels and Krauss (1992) mentions that there are time shifts between the annotated words and accompanying gestures.

In the case of syllable and morpheme segmentation the relation is regular. Starting with the morphemes based on a syllabification process, a syllable annotation can be inferred if the rules are known for a given language, although the segmentation is not the same on both levels. The different segmentation does not allow the integration of both structures in a context-free way (see Witt (2002b)).

The approach has to provide a means to describe these generalizable relations as well as systematic ones, for example using a certain gesture accompanied by a certain utterance. Coincidental relations, however need to be filtered out.

8.2.2 Time calculus

A time calculus is central for referencing between different annotation levels in which a signal is annotated on different linguistic levels. Every annotation

on each level has a start and an end and they can be related to each other in two ways:

1. Relation known from a knowledge base, either in form of linguistic knowledge or general knowledge. This includes knowledge such as *words* are part of *phrases* and relations such as grapheme-phoneme relations.
2. Unknown relations that need to be deduced from existing annotations. The relation within a linguistic annotation level can be generated statistically using traditional corpus linguistic techniques (see Biber et al. (1998)); for relations to other annotation layers this is not sufficient but needs further techniques besides a statistical analysis.

The use of knowledge bases for lexicon generation is well known (see Gibbon (2002a)) and has been practiced for some time in the study of artificial intelligence. The statistical distribution analysis has been used in corpus linguistics a lot. The relation beyond the limit of one level of linguistic annotation needs to be looked at.

In standard tier annotations the linguistic levels are presented underneath each other, so a researcher might relate the annotations and 'see' relations. For large data sets this is not possible due to a possible number of annotations. Users might tend to overlook relations or ignore some altogether. The automatic processing of these relations will be dealt with in this section.

For processing time relations between annotation units, the time relations need to be defined. For a person it might be possible, with a fuzzy notion of time relation to work on tier representations, but for automatic processing these relations need to be formally consistent. It is thus possible, starting from the idea that for every annotation unit, an *event* has a start and an end where the start is always before the end or at least the same as the end.

In a formal way the different time relations can be described in the following way:

Let E_n be an event with s_n start and e_n end with $s_n \leq e_n$. Then two events E_1 E_2 can have the following time relations:

1. $e_1 < s_2$; from the definition of $s_n \leq e_n$ follows that $e_1 < e_2$, therefore there are no more cases. E_1 is *after* E_2
2. $e_1 = s_2$; the same applies as above. E_1 *meets* E_2
3. Case: $s_1 < s_2$, which implies that $e_2 > s_1$. This relation has three subclasses:
   a) $e_1 < e_2$: It is said: E_1 *overlaps* E_2
   b) $e_1 = e_2$: E_1 is *finished by* E_2
   c) $e_1 > e_2$: E_1 *contains* E_2

4. Case: $s_1 = s_2$: from the definition of $s_n \leq e_n$ follows that $s_1 \leq e_2$, therefore there are no more cases than these three:
   a) $e_1 < e_2$: E_1 *starts* E_2
   b) $e_1 = e_2$: E_1 covers the *identical* interval as E_2
   c) $e_1 > e_2$: E_1 is *started by* E_2
5. Case $s_1 > s_2$:
   a) $s_1 < e_2$
      i. $e_1 < e_2$: E_1 is *during* E_2
      ii. $e_1 = e_2$: E_1 *finishes* E_2
      iii. $e_1 > e_2$: E_1 is *overlapped by* E_2
   b) $s_1 = e_2$: by the definition of $s_n \leq e_n$ follows that $e_1 \geq e_2$. E_1 is *met by* E_2
   c) $s_1 > e_2$: than $e_1 > e_2$ and E_1 is *before* E_2

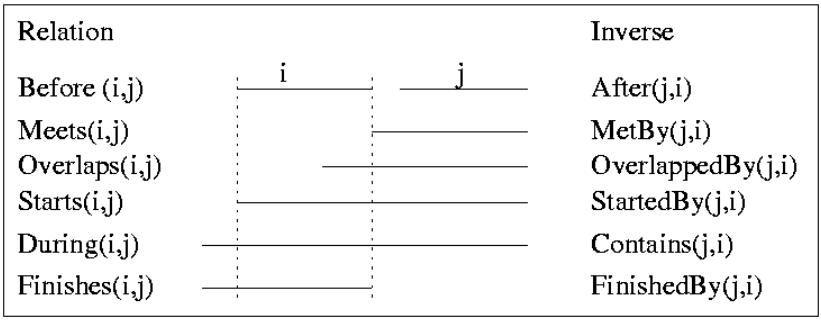

Fig. 8.1. Selected relations of annotation units, equality not included, from Allen and Ferguson (1994).

These are 13 time relations which are illustrated in Figure 8.1 and these are the only possible cases. However, it is not said that all time relations are productive for lexicon generation from multiple annotations. In fact the number of different annotations is rather large as every annotation unit is being related to every other annotation unit which results in the generalized Cartesian product. If the relations *before* and *after* are omitted for practical work, the number of relations to look at can be reduced significantly. The temporal calculus approach is illustrated in Allen and Ferguson (1994) or as a logic by van Benthem (1983). The approach is related to similar analyzes in constraint based phonology (see Bird (1995)). It makes extensive use of the graph structure of annotations.

Each annotation level can serve as the basis for lexical information such as wordlists extracted from an orthographic transcription, word frequency lexicons, etc. But if two or more obviously correlating annotation levels are present (such as orthographic word level transcription and canonical phonemic word level transcription), these relations have to be defined or discovered first. The temporal calculus based approach provides a method of finding assumptions in this area.

8.2.3 Classifying the relations of segments in a multi tier score

The acquisition of a lexicon is part of an integrated approach to corpus lexicography, as illustrated by Fig. 8.2. The lexicon is induced from a corpus by a lexicon acquisition function. The lexicon accesses the corpus by a concordancing function. Both relatively static units, lexicon and corpus can be used for multimodal output using appropriate technology.

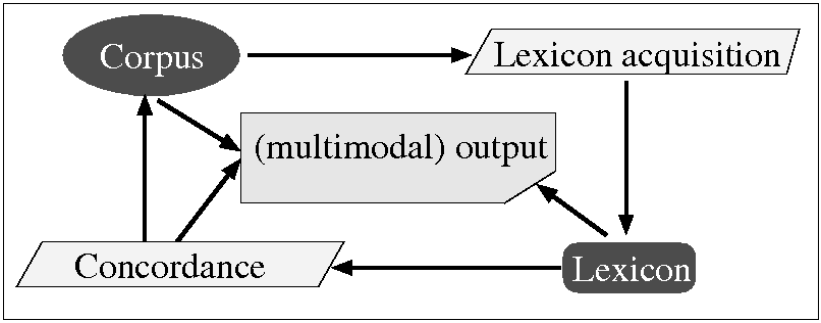

Fig. 8.2. Integrated corpus based lexicon system

Manually relating the annotated units is not possible for large data sets as it is error-prone, time consuming and tends to be inconsistent. Consequently a different way of realizing structured and consistent relations is needed.

Three possible classifications — which can be subclassified in a similar way — can be described for segments and layers. For segments — in TASX terminology *events* — ev_i from the set of all segments on one annotation tier EV, and tiers — in TASX terminology *layer* — la_j from the set of all tiers for one signal LA are:

1. *General* (implicational) relations that always exist, that is, for all e_i there is an e_j that is related. These are the most *general relations* and relatively easy to depict, for example that nouns are parts of noun phrases.
2. *Systematic* (tendential) relations that exist sufficiently often, in other words there are more than t elements e_i for which there is a e_j that is related. $t > 1$ is a threshold value of number of cases. If the threshold is sufficiently large, it could be an indicator for clustering a unit in different senses or at least to distinguish stylistic variation.
3. *Singular* (coincidental) relations that are unique in a given context: There is one or at least not more than t cases, where there is an e_i that has a relation to an e_j, where $t > 0$ a threshold value of the number of cases, so there are less than a threshold number of relations for two events. These cases give no particular information, as they might describe extremely rare examples as well as annotation inconsistency or indicate that the corpus is not sufficiently lexically saturated.

In lexicon synthesis, *general* and *systematic* relations are included. The systematic relations need further explanation and subcategorization and might need additional context information and linguistic knowledge. At least some statistical information is needed to enable a user to evaluate the relevance.

The position of a temporal relation within the hierarchical dependency structure — whether general or systematic — expresses the degree of universality of the relation, ranging from being a prototype for the whole lexical item to being a specific subclass of it.

8.2.4 Implementing a temporal calculus based lexicon

A proof of concept implementation of a temporal calculus based lexicon is based on two steps, the first is to create all relations between the segments. To extract the relations, each segment is taken and compared to every other segment that is not on the same tier. The time relations `before` and `after` are omitted to avoid the Cartesian product and to reduce the complexity. For this case Trippel et al. (2003) argue that as a rule of thumb the number of relations can be estimated by

$$\sum_{la \in LA} |EV_{la} * AVR * |LA|$$

with LA being the set of layers of annotations EV_{la} being the set of annotation events on layer la, $|LA|$ and $|EV_{la}|$ being the cardinality of these sets and AVR being the estimate average number of relation on every other layer.

Depending on the corpus, the AVR can be estimated. Corpora which have a large variation of segment length, i.e. layers with many long segments while on the others are many short ones, tend to have a higher average number of related segments.

The algorithm for the extraction of the annotations is illustrated by the following pseudocode, the element names referring to the structures in the TASX format.

```
for each LAYER_i
   for each EVENT_i on this LAYER_i
     for each LAYER_j , not i=j
       for each EVENT_j on LAYER_j
         compare EVENT_i/START and EVENT_j/START
         compare EVENT_i/END and EVENT_j/END
         assign relation between EVENT_i and EVENT_j
         if not EVENT_i and EVENT_j in same interval
            then omit relation
       next
     next
   next
next
```

An implementation for this algorithm using XSLT for the TASX format is publicly available, see Appendix 10, with some sample relations. However, this only gives the relations, but not the generalizations. For the generalizations both the base element and the one that is related to it are counted, i.e. how often these relation exists in terms of segments from the same layer and with the same temporal relation, and if this number is larger than the baseline, than this relation is either a general or systematic relation to be considered by a specialist. The following gives the pseudocode, in which the temporal relations are called STRUCT_REL.

```
for each STRUCT_REL
 if (count occurrences of BASE_ELEMENT and COMP_ELEMENT
     if  BASE_ELEMENT from LAYER_i and
         COMP_ELEMENT from LAYER_j and
         temporal relation identical)
      > BASELINE
     then
      return STRUCT_REL and number of occurrences
      next
     else
      next
```

A sample implementation of this algorithm in XSLT is available (see Appendix 10) with some sample relations. The interpretation of the results, however, still requires expert knowledge. Nevertheless, it allows the use of every data category as the headword and may point to systematic relations that are unlikely to be found based on introspection alone.

The time logic based lexicon extraction methods were applied to a subcorpus of the ModeLex project, the German fairy tale *Das Eselein* by Johann and Jacob Grimm in the TASX format. This annotation consists of 31 tiers with 14433 segments. Some tiers are not filled in yet and do not contain a lot of segments. By looking at the original data, an average of 4 related segments for each segment and each other layer was estimated, due to the different granularity of the annotation layers, starting from phrases, covering words and even syllables, but also gestures. The estimator mentioned above results in almost 1.8 million relations to be expected. Probably due to the annotation layers that are rather empty the real number of relations found is slightly smaller, namely 1481534. However, the large number and the used technology and scripts show that the processing time is a matter of hours rather then a matter of minutes. In fact, the processing took slightly more than 5 hours on an AMD Athlon 1150 MHz computer with 256 MB RAM. Though this is not a qualitative and quantitative evaluation with any significant results, it shows that the process is taking a lot of time and computing power.

The result of the time related lexicon, however, is already available in a format that is suitable to the Lexicon Graph model, each segment being a lexical item that is related to another by a time relation. The interpretation of some the generalizations and the investigation if these are related to existing rules and knowledge is left to future research.

8.3 Inheritance lexicons

Inheritance lexicons constitute a higher order lexicon of the third or fourth order according to the description at the beginning of Chapter 5. In the context of the generation of lexicons, inheritance lexicons are the consequence of filling the gaps left by a corpus, which does not contain all forms of a word. Inference lexicons are the kind of lexicon that use lexical knowledge to infer information about lexical items. These are especially interesting from the Lexicon Graph point of view, as they already use the reference structures described in the LG model. They allow a consistent maintenance by avoiding reduplication of entries and prepare a way of including a getting a maximum of lexical information from existent lexical resources. An example of an in-

ference lexicon is a lexicon entry that refers to a declination table for a noun. A non-standard form for a lexicon, for example a German dative case, can be inferred from the standard lexicon entry and the declination table.

The definition of references and inheritance is complex and requires expert knowledge, hence the motivation for this extra work needs to be explained. One reason for inheritance lexicons is the need for a redundancy free lexicon. Redundancy can be a problematic issue in the maintenance of a lexical database as previously described. A change in one lexical item requires, for consistency reasons, the update of all redundant bits of information. For this purpose, this redundant information has to be located first. Manual detection of redundancy is error prone, in which case inconsistencies are implied. For automatic processing, locating all redundant information and reducing the redundancy by using references and pointers is basically the same process, implying a form of compression. The issue of compression is neglected here. First, because the hardware capacities are growing steadily, second, because potential storage saving processes require more on the processing side for creating the full lexicon with redundancies, and third, the size of the pointers in a large lexicon is not necessarily smaller than the item itself. Hence the consistency issue is the argument here in the area of redundancy preventions.

Another reason for lexical inheritance is in extending the coverage of a lexicon for example in targeting the *Out Of Vocabulary* (OOV) problem, which is well known from speech technology in areas where statistical methods are used based on corpora. For example, a word that is perfectly acceptable but which, by coincidence, is not part of a training corpus, cannot be processed by a system. This is well known in other areas. Kilgarriff (2001) (Section 6) for example, evaluates existing lexicon theories with the help of *non standard* words of a corpus, which is for him, word meanings that are not covered by a reference lexicon.

The purpose of the corpus based, purely statistical solution is to increase the size of the corpus. However, if a different form of the word is in the corpus and the searched word can be derived from this form using simple rules, this word is a known word, hence the coverage of the lexicon can be bigger than the coverage of the original corpus. The necessary rules can be derived from a corpus using linguistic knowledge and clustering from the time logic inference of lexical structures as described before. The inheritance lexicon is not a corpus based method *per se*, but it can be used to enhance the lexicon derived from the corpus by expert knowledge.

Corpora may contain exceptions, that is forms and structures that are not described by existing rules. Exceptions to the rule cannot be handled by simple inheritance. This is true both for semantic and for grammatical modeling.

A solution to this is included in the DATR model (Evans and Gazdar (1996)), which allows the default overriding: exceptions are treated in a way that the default is ignored.

In the field of DATR many of lexicons have been implemented, a collection of some of them can be found in the DATR compendium (see Gibbon (1996) in the bibliography). A similar process has been used in the creation of the Verbmobil full form lexicon as described by Gibbon and Lüngen (2000).

In the field of the semantic web, the concept of inheritance is becoming increasingly popular (see for example Berners-Lee et al. (2001)), as is being able to describe a way of treating exceptions (see Binckley and Guha (2003) [2003])), but without a way of global inheritance.

8.4 Summary

In this chapter the use of corpora for the creation of higher order lexicons was described using a temporal calculus. This can be done by clustering to find generalizations but also by the application of inference rules. Different lexicons can be generated from different sorts of corpora, starting with simple wordlists and ending with sophisticated algorithms used for inferring lexical information from multiple annotations that are connected via a common time line. The use of additional inheritance rules applied according to the lexicon mesostructure can extend such a lexicon further to increase the coverage. The concept described in this chapter is too complex to be covered within this work, especially as large amounts of data is required. But this is already part of the perspectives of this approach.

9

Summary and perspectives

9.1 The Lexicon Graph in summary

Combining lexicons from different sources and extracting lexicons from multi-tier score annotations are two major problems related to language resources. The combination of lexicons can happen both when lexicon working groups are merged but also when for other reasons efforts are combined. This was problematic when the lexicon structures were very different, showing differences in the lexicon microstructure, in the lexicon mesostructure and the lexicon macrostructure. Existing lexicons in print and in electronic formats for applications show a huge variety of differences. However, a lot of them cover similar information, so that one lexicon can easily benefit from information contained in another lexicon. Pending issues in lexicography such as a conclusive model for ambiguity and hard cases such as homonymy and polysemy have to be treated appropriately as well. The problem of the combination of lexicons and the reuse of lexical resources for other then the intended purpose was to find a representation that is general enough to allow the representation of existing lexicons. This representation needed to be powerful enough to provide a way to avoid duplicates when merging lexicons, having an expressiveness to contain the structures of all the lexicons that are part of it. It was shown that a solution to these problems is the use of a very generic approach to the lexicon, seeing all bits of information in a lexicon as atomar and these atomar bits of information being related to each other. The result is the Lexicon Graph Model, a lexicon model that consists of a graph of lexical items as nodes and typed edges. Subgraphs are then the structures of the original lexicons. It is possible to combine different kinds of lexicons and extract lexicons from the common source that contains at least the same amount

of information, possibly even more as some additional information could be derived from another lexicon source. Additionally, different access structures are possible, e.g. by semantics, pronunciation, orthography, all based on the same lexical information, with the only prerequisite that these types of information exists in the lexicon graph.

The other problem mentioned above is the extraction of lexical information from multi-tier annotations. The reason for this is that a lot of linguistic information is contained in this form of annotations especially in contexts of spoken language. The expert knowledge could be harvested by experts writing a lexicon, but the formal foundation for this was not obvious. Combining the Lexicon Graph Model with the Annotation Graph Model (Bird and Liberman (2001)) provides a way to do so. Firstly. it shows that the formal foundation of corpus and lexicon are based on the same formalism. Secondly, by using a time based calculus, lexical information can be extracted from annotations which is immediately available in a Lexicon Graph format. For this creation of lexicons from annotations the conclusive description of the annotations using metadata is essential. Based on this metadata simple lexicons such as concordances for multimodal corpora can be created. Such a multimodal concordance was designed and implemented. Based on the lexicon graph model, thus it is possible to relate not only linear annotations such as texts, but also multi-tier score annotations to the lexicon and extract lexical information from this source of lexical knowledge.

9.2 Perspectives for the Lexicon Graph Model

The Lexicon Graph Model is a general model for lexicons, opening a lot of possibilities. The Lexicon Graph Model is at the beginning of the implementation and use, opening perspectives in at least the following areas:

1. The use of the model
2. The creation of lexicons with a time calculus
3. The enhancement of these created lexicons by inference rules
4. The extension of the concordance based on the Lexicon Graph Model.

The Lexicon Graph model can be used for the unification of existing, well-structured lexicons, i.e. lexicons that are available in an electronic format with explicit structural markup. Using the Lexicon Graph will allow the extension of these lexicons, resulting in a greater coverage. This includes also the use with existing ontologies that can be used as a lexical resource to be merged

with a lexicon graph. Implementations harvesting all available information — including inference — are the consequent next step in this development.

Another area of perspective is the lexicon creation based on annotations in multi-tier score formats. This allows the use of existing language resources from differing fields of linguistics, such as studies of endangered languages, studies of unwritten languages. Using the LG perspective enables the extension of lexicon studies to languages with lexical features not part of the orthographic systems such as lexical tone or morphology in syllable script languages. For this purpose, methods of statistical language resource analysis and constrain based approaches need to be combined. The reason for this is that the number of individual sequences that need to be processed, is too huge to process manually, but too small for relevant statistical correlations. It is not expected that the size of the available corpora can be increased significantly, as the annotation on multiple tiers is only possible manually or machine assisted, at least for some linguistic levels. The manual annotation however is expensive, that means that for generalizations constraints based on expert knowledge have to be taken into account as well.

The multimodal concordance can be used in further contexts, which opens more perspectives. This includes the use of the concordance for more data, i.e. the addition of further corpora. The use with more corpora also extends to the addition of video data using the same interfaces, which requires to build in a way of streaming the video data over the internet. This could be accomplished by using standard procedures that are only in the process of emerging. The addition of further functionalities for example by the combination of the concordance system with further analysis features for the audio signal is another option. The analysis of the signal could be extended by more providing more options to a user. Another additional feature for the multimodal concordance is the combination of constraints for more than one tier, e.g. a query for an lexical element l_1 of type t_1 with a parallel occurrence of a lexical item l_2 of type t_2. A typical example of such a query would be the search for a specific orthographic form of a word with a certain prosodic pattern, such as a word with a high tone.

The development of these perspectives and options is left to further research. All of these could be the starting point for more developments and extensions to this work.

Part III

Appendix: Auxiliary programs, resources and bibliography

10

Auxiliary programs and resources

Some programs written for proof of concept studies and mentioned in the previous chapters are referenced here. The interested reader can use them for reimplementation of similar systems. As these are many lines of code only important for the reimplementation, the code is published on a website. The programs, grammars and resources include:

- programs for merging two lexicon graphs
- an LG format to the dot graph format transformation function
- a Feature Structure Representation into LG transformation
- a lexicon generating program based on the TASX format
- an XSLT stylesheet for the extraction of time logics relations in TASX files
- some sample time logics relations extracted from TASX files
- an XSLT stylesheet to generalism over time relations
- sample generalizations over time relations
- the TASX document grammar and
- a metadata lexicon

All of these can be found on the website `http://www.lexicongraph.de/appendix`.

For completeness these programms are printed here in source code, however the reader is strongly advised to use the source code and programs as published on the website.

11

Merging two lexicon graphs

```
(: XQUERY for processing LG data :)
(: Thorsten Trippel August 2004 :)

declare namespace my="my-functions.uri";
declare variable $largelexinfile as xs:string external;
declare variable $smalllexinfile as xs:string external;

(: Function disambiguate-id: tests if an id is already
   in use somewhere else,  :)
(: and if so adds as many letters a to make it unique :)

declare function my:disambiguate-id
($ambiguousid as xs:string, $largelexitems as node()* )
as xs:string
  {
  if ($ambiguousid = $largelexitems/@lexid) then
    my:disambiguate-id(concat($ambiguousid,"a"),
                              $largelexitems)
  else ($ambiguousid)
  };

(: Transformation of the lexicon:
   Takes a large LG lexicon and a small lexicon. :)
(: Every element of the small one is tested against
   the large one and if not already in there is added.:)
(: Before the addition the ids are adjusted.
   The references from the relations are
   adjusted as well :)
```

```
(: Every lexitem of the large LG lexicon is copied if
   it is not part of the small lexicon :)
(: roles of lexicons can be interchanged, with
   consequences on performance, maybe :)


(: Definition of the lexicon files:)
let $largelex := doc($largelexinfile),
                 (:LG_complete.xml :)
$smalllex := doc($smalllexinfile)
                   (:LG_to_be_merged.xml:)

(: This is the lexitems from the large lexicon :)

let $largelexitems := $largelex/LG/lexitems/lexitem

(: These variable is used in processing the relations :)
let $largelexrelations :=
                         $largelex/LG/relations/relation
let $largelex_relation_target_id_type_concat :=
    for $aux11 in $largelexrelations
    return <concat_idref_type>{
concat($aux11/target/@idref,
$aux11/@type)}</concat_idref_type>

(: This is the processing of the lexical items :)
(: First all lexical items from the small lexicon are
    tagged if they are already in the large one;
    types are not taken into account here :)

let $potentiallexitemsmall :=
      <all_lexitems_smalllex>{
      (: Process all elements of the small lexicon
         into ones that are not redundant and others
         that could be candidates for redundancies :)

      for $potentiallexitem in
                       $smalllex/LG/lexitems/lexitem
      return
       if ($potentiallexitem/text() =
              $largelexitems/text()) then
        let $concataux :=
         for $largelexitems_test in $largelexitems
```

```
 where $largelexitems_test/text()=
                $potentiallexitem/text()
 return
           <concataux>
    {
           concat($largelexitems_test/text(),
                     $largelexitems_test/@type)}
           </concataux>
        return
 if (concat($potentiallexitem/text(),
           $potentiallexitem/@type)=$concataux) then
   for $largelexitems_test in $largelexitems
   where
             $largelexitems_test/text()=
                     $potentiallexitem/text() and
             $largelexitems_test/@type =
                     $potentiallexitem/@type
   return
<redundant>{<lexitem
  type="{$potentiallexitem/@type}"
  lexid="{$largelexitems_test/@lexid}"
  orglexid="{$potentiallexitem/@lexid}"
>
{$potentiallexitem/text()}</lexitem>}
</redundant>
     else
<nonredundant>
                  {
                   <lexitem
     type="{$potentiallexitem/@type}"
     lexid=
"{my:disambiguate-id(
                $potentiallexitem/@lexid,
                                $largelexitems)}"
     orglexid="
                             {$potentiallexitem/@lexid}"
   >
   {$potentiallexitem/text()}
                   </lexitem>}
</nonredundant>
      else
          <nonredundant>
           {<lexitem
```

```
    type="{$potentiallexitem/@type}"
    lexid="{my:disambiguate-id(
              $potentiallexitem/@lexid,
              $largelexitems)}"
           orglexid="{$potentiallexitem/@lexid}">
          {$potentiallexitem/text()}
          </lexitem>
  }
          </nonredundant>
        }
 </all_lexitems_smalllex>

(: Here the lexitems that are definitively not identical
are processed :)

let $alllexitems :=
  <lexitems>
    {
    (: Select lexitems having non identical texts
       and adjust identifiers :)
    for $nonredundant in
        $potentiallexitemsmall/nonredundant/lexitem
    return $nonredundant
    }
    {
  (: Here the typing is applied but only for the items
      that might be candidates, :)
  (: i.e. their content is in both lexicons :)
    for $largelexitem in $largelexitems
    return $largelexitem
    }
  </lexitems>

(: In the smalllexicon the identifiers in the relations
   need to be adjusted as well :)
let $modifiedrelations :=
  <relations>{
    for $target in
                $smalllex/LG/relations/relation/target
    return
      <relation type="{$target/../@type}" >
      {
      if ($potentiallexitemsmall//lexitem/@orglexid =
```

```
          $target/@idref) then
        for $alllexitems_4rel in
                          $potentiallexitemsmall//lexitem
where $alllexitems_4rel/@orglexid =
                                              $target/@idref
return
              <target idref="{$alllexitems_4rel/@lexid}"
       orgidref="{$target/@idref}"/>
        else <target idref="{$target/@idref}"
     orgidref="unique"/>
      }
      {
      for $source in $target/../source
      return
if ($potentiallexitemsmall//lexitem/@orglexid =
            $source/@idref)
        then
  for $alllexitems_4rel in
              $potentiallexitemsmall//lexitem
  where $alllexitems_4rel/@orglexid =
                                              $source/@idref
  return <source idref="
                               {$alllexitems_4rel/@lexid}"
     orgidref="{$source/@idref}"/>
  else <source idref="{$source/@idref}"
       orgidref="unique"/>
}
</relation>
      }
  </relations>

(: With the adjusted relations the relations can
   now be processed :)

let $allrelations :=
    <relations>
      {
(: First the intersection of both lexicons :)
for $largelexrelation in $largelexrelations
let $smalllex_intersect_rel :=
            $modifiedrelations/relation

let $large_relation_target_id_type_concat :=
```

```
    concat($largelexrelation/target/@idref,
                    $largelexrelation/@type)
let $smalllex_relation_target_id_type_concat :=
    for $smalllex_concat_aux in
                 $smalllex_intersect_rel/target
    return <concat_aux2>
    {concat($smalllex_concat_aux/@idref,
                     $smalllex_concat_aux/../@type)}
    </concat_aux2>
where $large_relation_target_id_type_concat =
              $smalllex_relation_target_id_type_concat
(: This is the intersection of the relations from the
     target perspective :)
return
          <relation type="{$largelexrelation/@type}">
     <target idref="
                     {$largelexrelation/target/@idref}"/>
                {
(: Now the source items need to be checked
   for redundancies :)
    let $idrefs :=
       for $aux1 in
                 $smalllex_intersect_rel/source/@idref,
$aux2 in $largelexrelation/source/@idref
return <source>{$aux1}{$aux2}</source>
    return
    for $aux3 in distinct-values($idrefs/@idref)
 return <source idref="{$aux3}"/>
       }
  </relation>
        }
{
(: Now the ones only present in
           the large lexicon :)
for $largelexrelation in $largelexrelations
let $large_relation_target_id_type_concat :=
    concat($largelexrelation/target/@idref,
                    $largelexrelation/@type)
let $small_relation_target_id_type_concat :=
 for $aux4 in $modifiedrelations/relation/target
 return
           <concat_idref_type>
           {
```

```
   concat($aux4/@idref,$aux4/../@type)}
            </concat_idref_type>
where not(
          $small_relation_target_id_type_concat/text() =
                  $large_relation_target_id_type_concat)
return $largelexrelation
}
{
(: Finally the ones only in the small lexicon :)
   for $smalllexrelation in
                       $modifiedrelations/relation/target
   let $small_relation_target_id_type_concat :=
       concat($smalllexrelation/@idref,
              $smalllexrelation/../@type)
   let $largelex_relation_target_id_type_concat :=
       for $aux5 in $largelexrelations
       return
         <concat_idref_type>{
   concat($aux5/target/@idref,
$aux5/@type)}</concat_idref_type>
   where not($small_relation_target_id_type_concat =
        $largelex_relation_target_id_type_concat/text())
   return $smalllexrelation/..
   }
 </relations>
(: Right here comes the output :)
return
  <LG>
    <lexitems>{
      for $outlexitems in $alllexitems/lexitem
      return
        <lexitem lexid="{$outlexitems/@lexid}"
             type="{$outlexitems/@type}">
 {$outlexitems/text()}
        </lexitem>
        }
    </lexitems>
{$allrelations}
{
(: Process everything else :)
for $everythingelse in $smalllex/LG/knowledge
 return $everythingelse}
</LG>
```

12

LG to dot transformation

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
      version="1.1">
  <xsl:output method="text" version="1.0"
                                    encoding="ISO-8859-1"
          omit-xml-declaration="no" indent="yes"/>
<xsl:template match="/LG">
      <xsl:text>digraph G {
</xsl:text>
      <xsl:apply-templates select="relations/relation"/>
      <xsl:text>}
</xsl:text>
</xsl:template>

<xsl:template match="relations/relation"
                                          name="relation">
  <xsl:for-each select="source">
   <xsl:variable name="sourceidref" select="@idref"/>
   <xsl:variable name="targetidref"
                             select="../target/@idref"/>
   <xsl:variable name="targetname"
     select="../../../lexitems/lexitem[
                                 @lexid=$targetidref]"/>
   <xsl:variable name="sourcename"
      select="../../../lexitems/lexitem[@lexid=
                                          $sourceidref]"/>
   <xsl:variable name="translate_from">"</xsl:variable>
```

```
   <xsl:variable name="translate_to">\"</xsl:variable>
   <xsl:text>"</xsl:text>
   <xsl:call-template name="replace-string">
      <xsl:with-param name="text" select="$sourcename"/>
      <xsl:with-param name="from"
                              select="$translate_from"/>
      <xsl:with-param name="to" select="$translate_to"/>
   </xsl:call-template>
   <xsl:text>"  -&gt; "</xsl:text>
   <xsl:call-template name="replace-string">
      <xsl:with-param name="text" select="$targetname"/>
      <xsl:with-param name="from"
                              select="$translate_from"/>
      <xsl:with-param name="to" select="$translate_to"/>
   </xsl:call-template>
     <xsl:text>" [label="</xsl:text>
     <xsl:value-of select="../@type"/>
     <xsl:text>"];
</xsl:text>
  </xsl:for-each>
</xsl:template>

<xsl:template match="lexitems/lexitem/@lexid"
     name="idrefresolve">
  <xsl:value-of select="../*"/>
</xsl:template>



<xsl:template name="replace-string">
  <xsl:param name="text"/>
    <xsl:param name="from"/>
     <xsl:param name="to"/>
     <xsl:choose>
       <xsl:when test="contains($text, $from)">
         <xsl:variable name="before"
               select="substring-before($text, $from)"/>
          <xsl:variable name="after"
                select="substring-after($text, $from)"/>
          <xsl:variable name="prefix"
                select="concat($before, $to)"/>

          <xsl:value-of select="$before"/>
```

```
          <xsl:value-of select="$to"/>
          <xsl:call-template name="replace-string">
           <xsl:with-param name="text" select="$after"/>
            <xsl:with-param name="from" select="$from"/>
            <xsl:with-param name="to" select="$to"/>
           </xsl:call-template>
         </xsl:when>
         <xsl:otherwise>
           <xsl:value-of select="$text"/>
         </xsl:otherwise>
      </xsl:choose>
  </xsl:template>
</xsl:stylesheet>
```

13

FSR to LG transformation

13.1 Lexical item listing function

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet
     xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    version="2.0">
<xsl:output method="xml" version="1.0"
encoding="utf-8" omit-xml-declaration="yes"
                                          indent="yes"/>

   <xsl:template match="/">
      <xsl:element name="lexitems">
         <xsl:apply-templates/>
      </xsl:element>
     </xsl:template>

   <xsl:template match="f">
      <xsl:element name="lexitem">
         <xsl:attribute name="type">FSR_class
                                       </xsl:attribute>
         <xsl:attribute name="lexid">
          <xsl:value-of select="./@name/generate-id()"/>
         </xsl:attribute>
         <xsl:value-of select="@name"/>
      </xsl:element>
      <xsl:apply-templates/>
   </xsl:template>
```

```
   <xsl:template match="fs">
      <xsl:apply-templates/>
   </xsl:template>

   <xsl:template match="str">
      <xsl:element name="lexitem">
         <xsl:attribute name="type">
            <xsl:value-of select="../@name"/>
         </xsl:attribute>
         <xsl:attribute name="lexid">
            <xsl:value-of select="generate-id()"/>
         </xsl:attribute>
         <xsl:value-of select="."/>
      </xsl:element>
   </xsl:template>

   <xsl:template match="plus">
      <xsl:element name="lexitem">
         <xsl:attribute name="type">FSR_binary
             </xsl:attribute>
         <xsl:attribute name="lexid">
            <xsl:value-of select="generate-id()"/>
         </xsl:attribute>
         <xsl:text>+</xsl:text>
      </xsl:element>
      <xsl:apply-templates/>
   </xsl:template>

</xsl:stylesheet>
```

13.2 Lexical relations from left to right

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet
     xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
     version="2.0">

   <xsl:output method="xml" version="1.0"
        encoding="utf-8" omit-xml-declaration="yes"
        indent="yes"/>
<!-- Root element match feature structure -->
```

```
   <xsl:template match="/fs">
<!-- xsl:element name="relations" -->
       <xsl:apply-templates/>
<!-- /xsl:element -->
   </xsl:template>

<!-- we see a feature, we look at the value -->
 <xsl:template match="f">
    <xsl:variable name="sourceid">
       <xsl:value-of select="@name/generate-id()"/>
    </xsl:variable>
<!-- only the features that do not refer to other
     features -->
    <xsl:for-each select=".[not(@fVal)]">

<!-- Now let's look at the features which
     have embedded features -->
<!-- first without reentrancy -->
      <xsl:for-each select="fs/f[not(@fVal)]">
         <xsl:element name="relation">
            <xsl:attribute name="type">
               <xsl:text>is_instanced_by</xsl:text>
            </xsl:attribute>
            <xsl:element name="source">
               <xsl:attribute name="idref">
                <xsl:value-of
                 select="../../@name/generate-id()"/>
               </xsl:attribute>
            </xsl:element>
            <xsl:element name="target">
               <xsl:attribute name="idref">
                 <xsl:value-of
                          select="@name/generate-id()"/>
               </xsl:attribute>
            </xsl:element>
         </xsl:element>
      </xsl:for-each>

<!-- Well, now we have to look at the features with
     reentrancy -->
         <xsl:for-each select="fs/f[@fVal]">
            <xsl:variable name="fVal">
               <xsl:value-of select="@fVal"/>
```

```
      </xsl:variable>
      <xsl:for-each select="//fs[@id=$fVal]/f">
        <xsl:element name="relation">
          <xsl:attribute name="type">
            <xsl:text>is_instanced_by</xsl:text>
          </xsl:attribute>
          <xsl:element name="source">
            <xsl:attribute name="idref">
               <xsl:value-of select="$sourceid"/>
            </xsl:attribute>
          </xsl:element>

          <xsl:element name="target">
            <xsl:attribute name="idref">
              <xsl:value-of
                    select="@name/generate-id()"/>
            </xsl:attribute>
          </xsl:element>
        </xsl:element>
      </xsl:for-each>
    </xsl:for-each>
    <xsl:for-each select=".[not(fs)]">
      <xsl:element name="relation">
        <xsl:attribute name="type">
          <xsl:text>has_instance</xsl:text>
        </xsl:attribute>
        <xsl:element name="source">
          <xsl:attribute name="idref">
            <xsl:value-of
                    select="@name/generate-id()"/>
          </xsl:attribute>
        </xsl:element>
        <xsl:element name="target">
          <xsl:attribute name="idref">
            <xsl:if test="str">
              <xsl:value-of
                      select="str/generate-id()"/>
            </xsl:if>
            <xsl:if test="plus">
              <xsl:value-of
                     select="plus/generate-id()"/>
            </xsl:if>
          </xsl:attribute>
```

```
            </xsl:element>
          </xsl:element>
        </xsl:for-each>
      </xsl:for-each>

      <xsl:apply-templates/>

   </xsl:template>

   <xsl:template match="fs">
      <xsl:apply-templates/>
   </xsl:template>


<!-- Here we have the terminal symbols str and plus
 we can deal with them easily by referring to them and
looking at the feature -->
   <xsl:template match="str">
   </xsl:template>

   <xsl:template match="plus">
   </xsl:template>


</xsl:stylesheet>
```

13.3 Lexical relations from right to left

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
       version="2.0">
   <xsl:output method="xml" version="1.0"
        encoding="utf-8" omit-xml-declaration="yes"
        indent="yes"/>
<!-- Root element match feature structure -->
   <xsl:template match="/fs">
      <xsl:apply-templates/>
   </xsl:template>

<!-- we see a feature, we look at the value -->
```

```
   <xsl:template match="f">
      <xsl:element name="relation">
         <xsl:attribute name="type">
            <xsl:text>is_subcategory_of</xsl:text>
         </xsl:attribute>
         <xsl:element name="source">
          <xsl:attribute name="idref">
            <xsl:value-of select="@name/generate-id()"/>
          </xsl:attribute>
         </xsl:element>
         <xsl:element name="target">
           <xsl:attribute name="idref">
             <xsl:value-of
                    select="../../@name/generate-id()"/>
         </xsl:attribute>
        </xsl:element>
      </xsl:element>
      <xsl:apply-templates/>
   </xsl:template>

   <xsl:template match="fs">
<!-- This is for the inverse reentrancy -->
<!-- Referred fs need to have an id, so let's look at
     them -->
      <xsl:if test="@id">
         <xsl:variable name="fs_id">
            <xsl:value-of select="@id"/>
         </xsl:variable>

         <xsl:for-each select="//f[@fVal=$fs_id]">
            <xsl:variable name="target_id">
               <xsl:value-of
                          select="@name/generate-id()"/>
            </xsl:variable>
            <xsl:element name="relation">
              <xsl:attribute name="type">
                <xsl:text>is_subcategory_of</xsl:text>
              </xsl:attribute>
              <xsl:for-each select="//fs[@id=$fs_id]/f">
                <xsl:element name="source">
                  <xsl:attribute name="idref">
                    <xsl:value-of
                          select="@name/generate-id()"/>
```

```
                    </xsl:attribute>
                  </xsl:element>
               </xsl:for-each>
               <xsl:element name="target">
                 <xsl:attribute name="idref">
                   <xsl:value-of select="$target_id"/>
                 </xsl:attribute>
               </xsl:element>
            </xsl:element>
         </xsl:for-each>
      </xsl:if>
      <xsl:apply-templates/>
   </xsl:template>

<!-- Here we have the terminal symbols str and plus
 we can deal with them easily by referring to them and
 looking at  the feature -->
<!-- Strings are attribute value structures, easy to
     handle -->
   <xsl:template match="str">
      <xsl:element name="relation">
         <xsl:attribute name="type">
            <xsl:text>is_value_of</xsl:text>
         </xsl:attribute>
         <xsl:element name="source">
            <xsl:attribute name="idref">
               <xsl:value-of select="./generate-id()"/>
            </xsl:attribute>
         </xsl:element>
         <xsl:element name="target">
            <xsl:attribute name="idref">
               <xsl:value-of
                       select="../@name/generate-id()"/>
            </xsl:attribute>
         </xsl:element>
      </xsl:element>
      <xsl:apply-templates/>
   </xsl:template>

   <xsl:template match="plus">
      <xsl:element name="relation">
         <xsl:attribute name="type">
            <xsl:text>is_value_of</xsl:text>
```

```
         </xsl:attribute>
         <xsl:element name="source">
            <xsl:attribute name="idref">
               <xsl:value-of select="./generate-id()"/>
            </xsl:attribute>
         </xsl:element>
         <xsl:element name="target">
            <xsl:attribute name="idref">
               <xsl:value-of
                       select="../@name/generate-id()"/>
            </xsl:attribute>
         </xsl:element>
      </xsl:element>
      <xsl:apply-templates/>
   </xsl:template>

</xsl:stylesheet>
```

14

Lexicon generating program

14.1 Word frequency lexicon for TASX corpora

The following XQuery is a program creating a simple word frequency lexicon from a TASX corpus.

```
<frequencylexicon>{
let $file := doc("filename.xml")

for $layermeta in $file//session/layer/meta/desc
where $layermeta/name="layer name" and
                                    $layermeta/val="words"
return
  for $word in distinct-values($layermeta/../../event)
  order by $word
  return
    <lexentry>
      <word>
        {$word}
      </word>
      <absfrequency>
        {count(
          for $frequencywords in $layermeta/../../event
          where $frequencywords=$word
          return $word)
        }
      </absfrequency>
      <relfrequency>
        {count(
```

```
            for $frequencywords in $layermeta/../../event
            where $frequencywords=$word
            return $word
            )
            div
           count(
            for $frequencywords in $layermeta/../../event
            return $word)
          }
      </relfrequency>
    </lexentry>
}
</frequencylexicon>
```

The following is a part of the resulting lexicon:

```
<lexentry>
      <word>ab</word>
      <absfrequency>2</absfrequency>
      <relfrequency>0.001845018450184502</relfrequency>
   </lexentry>
   <lexentry>
      <word>Abend</word>
      <absfrequency>1</absfrequency>
      <relfrequency>0.0009225092250922 51</relfrequency>
   </lexentry>
   <lexentry>
      <word>Abends</word>
      <absfrequency>1</absfrequency>
      <relfrequency>0.0009225092250922 51</relfrequency>
   </lexentry>
   <lexentry>
      <word>aber</word>
      <absfrequency>7</absfrequency>
      <relfrequency>0.006457564575645756</relfrequency>
   </lexentry>
```

14.2 Time relations extracted from TASX annotations

14.2.1 XSLT stylesheet: Time logics relations in TASX files

The following XSLT stylesheet produces the list of segments of a TASX file with their related segments from other tiers, using the time calculus relations

from Allen and Ferguson (1994). The relations *before* and *after* are not processed.

```
<xsl:stylesheet
     xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
     version="1.1">
  <xsl:output method="xml" version="1.0"
       encoding="utf-8" omit-xml-declaration="yes"
       indent="yes"/>

  <xsl:template match="/">
      <xsl:apply-templates/>
  </xsl:template>

  <xsl:template match="session">
   <xsl:element name="struct_rel">
     <xsl:apply-templates select="layer"/>
   </xsl:element>
  </xsl:template>

  <xsl:template match="layer">
   <xsl:variable name="layerid" select="@l-id"/>
     <xsl:for-each select="event[@start &lt; @end and
                                                text()]">
       <xsl:variable name="startevent" select="@start"/>
       <xsl:variable name="endevent" select="@end"/>
       <xsl:variable name="element" select="text()"/>
       <xsl:variable name="eventid" select="@e-id"/>

       <xsl:for-each select="../../layer[not(@l-id =
                                          $layerid)]">
         <xsl:for-each select="event[@start &lt; @end
                                        and text()]">
           <xsl:variable name="meta2">
             <xsl:copy-of select="meta"/>
           </xsl:variable>
           <xsl:variable name="OUTPUT_OF_PARAMS">
             <xsl:element name="baselayerid">
               <xsl:value-of select="$layerid"/>
             </xsl:element>
             <xsl:element name="baseeventid">
               <xsl:value-of select="$eventid"/>
             </xsl:element>
             <xsl:element name="basestartevent">
```

```
                <xsl:value-of select="$startevent"/>
              </xsl:element>
              <xsl:element name="baseendevent">
                <xsl:value-of select="$endevent"/>
              </xsl:element>
              <xsl:element name="baseelement">
                <xsl:value-of select="$element"/>
              </xsl:element>
              <xsl:element name="complayer">
                <xsl:value-of select="../@l-id"/>
              </xsl:element>
              <xsl:element name="compeventid">
                <xsl:value-of select="@e-id"/>
              </xsl:element>
              <xsl:element name="compstartevent">
                <xsl:value-of select="@start"/>
              </xsl:element>
              <xsl:element name="compendevent">
                <xsl:value-of select="@end"/>
              </xsl:element>
              <xsl:element name="compelement">
                <xsl:value-of select="text()"/>
              </xsl:element>
   </xsl:variable>

<!-- 1. Case: Identity of start times-->
            <xsl:if test="$startevent=@start">
 <!-- Case 1.1: start identical, end smaller than
                 compared value-->
              <xsl:if test="$endevent &lt; @end">
      <xsl:element name="relationpair">
                <xsl:copy-of select="$OUTPUT_OF_PARAMS"/>
       <xsl:element name="relation">
                   <xsl:attribute name="rel">
                    <xsl:text>starts</xsl:text>
                   </xsl:attribute>
                 </xsl:element>
              </xsl:element>
            </xsl:if>
<!-- Case 1.2: start and end identical -->
              <xsl:if test="$endevent = @end">
               <xsl:element name="relationpair">
                <xsl:copy-of select="$OUTPUT_OF_PARAMS"/>
```

```
                <xsl:element name="relation">
                 <xsl:attribute name="rel">
                   <xsl:text>ident</xsl:text>
                 </xsl:attribute>
                </xsl:element>
               </xsl:element>
              </xsl:if>
<!-- Case 1.3: start identical, end greater than comp.
                value -->
              <xsl:if test="$endevent &gt; @end">
               <xsl:element name="relationpair">
                <xsl:copy-of select="$OUTPUT_OF_PARAMS"/>
                 <xsl:element name="relation">
                  <xsl:attribute name="rel">
                   <xsl:text>started_by</xsl:text>
                  </xsl:attribute>
                 </xsl:element>
               </xsl:element>
               </xsl:if>
             </xsl:if>

<!-- 2. Case: event start before the one compared to
        it -->
           <xsl:if test="$startevent &lt; @start">
<!-- Case 2.1 event is completely before the one
        compared  to: not dealt with -->
             <!-- <xsl:if test="$endevent &lt; @start">
                  <xsl:element name="relationpair">
                  <xsl:copy-of select="$OUTPUT_OF_PARAMS"/>
                  <xsl:element name="relation">
                  <xsl:attribute name="rel">
                  <xsl:text>before</xsl:text>
                  </xsl:attribute>
                  </xsl:element>
                  </xsl:element>
                  </xsl:if>
              -->

<!-- Case 2.2 event finisehs when the one compared to
      it starts -->
              <xsl:if test="$endevent = @start">
               <xsl:element name="relationpair">
                <xsl:copy-of select="$OUTPUT_OF_PARAMS"/>
```

```
                <xsl:element name="relation">
                 <xsl:attribute name="rel">
                  <xsl:text>meets</xsl:text>
                 </xsl:attribute>
                </xsl:element>
               </xsl:element>
              </xsl:if>
<!-- Case 2.3 event finishes while the one compared to
       it is already running -->
              <xsl:if test="$endevent &gt; @start">
<!-- Case 2.3.1 event finishes before the one compared
                to -->
             <xsl:if test="$endevent &lt; @end">
              <xsl:element name="relationpair">
               <xsl:copy-of select="$OUTPUT_OF_PARAMS"/>
                <xsl:element name="relation">
                 <xsl:attribute name="rel">
                  <xsl:text>overlaps</xsl:text>
                 </xsl:attribute>
                </xsl:element>
              </xsl:element>
             </xsl:if>
<!-- Case 2.3.2 -->
             <xsl:if test="$endevent &gt; @end">
              <xsl:element name="relationpair">
               <xsl:copy-of select="$OUTPUT_OF_PARAMS"/>
                <xsl:element name="relation">
                 <xsl:attribute name="rel">
                  <xsl:text>contains</xsl:text>
                 </xsl:attribute>
                </xsl:element>
               </xsl:element>
             </xsl:if>
<!-- Case 2.3.3 -->
             <xsl:if test="$endevent = @end">
              <xsl:element name="relationpair">
               <xsl:copy-of select="$OUTPUT_OF_PARAMS"/>
                <xsl:element name="relation">
                 <xsl:attribute name="rel">
                 <xsl:text>finished_by</xsl:text>
                 </xsl:attribute>
                </xsl:element>
              </xsl:element>
```

```
            </xsl:if>
<!-- end Case 2.3 -->
           </xsl:if>
<!--  end Case 2 -->
          </xsl:if>
<!-- 3. Case event beginns after the one compared
        with -->
         <xsl:if test="$startevent &gt; @start">
<!-- Case 3.1 -->
          <xsl:if test="$startevent &lt; @end">
<!-- Case 3.1.1 -->
           <xsl:if test="$endevent &lt; @end">
            <xsl:element name="relationpair">
             <xsl:copy-of select="$OUTPUT_OF_PARAMS"/>
             <xsl:element name="relation">
              <xsl:attribute name="rel">
               <xsl:text>during</xsl:text>
              </xsl:attribute>
             </xsl:element>
            </xsl:element>
           </xsl:if>
<!-- Case 3.1.2 -->
           <xsl:if test="$endevent =@end">
            <xsl:element name="relationpair">
             <xsl:copy-of select="$OUTPUT_OF_PARAMS"/>
             <xsl:element name="relation">
              <xsl:attribute name="rel">
               <xsl:text>finishes</xsl:text>
              </xsl:attribute>
             </xsl:element>
            </xsl:element>
           </xsl:if>
<!-- Case 3.1.3 -->
           <xsl:if test="$endevent &gt; @end">
            <xsl:element name="relationpair">
             <xsl:copy-of select="$OUTPUT_OF_PARAMS"/>
             <xsl:element name="relation">
              <xsl:attribute name="rel">
               <xsl:text>overlapped_by</xsl:text>
              </xsl:attribute>
             </xsl:element>
            </xsl:element>
           </xsl:if>
```

```
          </xsl:if>
<!-- Case 3.2 -->
          <xsl:if test="$startevent = @end">
            <xsl:element name="relationpair">
             <xsl:copy-of select="$OUTPUT_OF_PARAMS"/>
             <xsl:element name="relation">
              <xsl:attribute name="rel">
               <xsl:text>met_by</xsl:text>
              </xsl:attribute>
             </xsl:element>
            </xsl:element>
          </xsl:if>
<!-- Case 3.3 -->
<!-- not dealt with
           <!--
             <xsl:if test="$startevent &gt; @end">
             <xsl:element name="relationpair">
             <xsl:copy-of select="$OUTPUT_OF_PARAMS"/>
             <xsl:element name="relation">
             <xsl:attribute name="rel">
             <xsl:text>after</xsl:text>
             </xsl:attribute>
             </xsl:element>
             </xsl:element>
             </xsl:if>
              -->
          </xsl:if>
        </xsl:for-each>
       </xsl:for-each>
      </xsl:for-each>
   </xsl:template>

   <xsl:template match="meta">
   </xsl:template>

</xsl:stylesheet>
```

14.2.2 Sample time logics relations extracted from TASX files

Some sample relations from this function from the ModeLeX corpus are given below.

```
   <relationpair>
```

```
    <baselayerid>hands</baselayerid>
    <baseeventid/>
    <basestartevent>815.18</basestartevent>
    <baseendevent>906.88</baseendevent>
    <baseelement>17m,7A,sy</baseelement>
    <complayer>esel_einfache_antefinal.words</complayer>
    <compeventid>163</compeventid>
    <compstartevent>88.053064304977752</compstartevent>
    <compendevent>88.895005999999995</compendevent>
    <compelement>Eselein</compelement>
    <relation rel="contains"/>
   </relationpair>
   <relationpair>
    <baselayerid>hands</baselayerid>
    <baseeventid/>
    <basestartevent>815.18</basestartevent>
    <baseendevent>906.88</baseendevent>
    <baseelement>17m,7A,sy</baseelement>
    <complayer>esel_einfache_antefinal.words</complayer>
    <compeventid>164</compeventid>
    <compstartevent>88.895005999999995</compstartevent>
    <compendevent>89.969133999999997</compendevent>
    <compelement>aufgezogen</compelement>
    <relation rel="contains"/>
   </relationpair>
```

14.2.3 XSLT stylesheet: Generalising over time relations

For the extraction of generalisations another script is used, the XSLT implementation is the following, using a baseline parameter to avoid the inclusion of singular relations.

```
<xsl:stylesheet
     xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
      version="1.0">
     <xsl:output method="xml" version="1.0"
          encoding="utf-8" omit-xml-declaration="yes"
          indent="yes"/>
    <!-- gen_lex_times.xsl
       by Thorsten Trippel
       Department of Linguistics and Literary Studies,
       Bielefeld University
       March 2003
```

```
      XSL Stylesheet for transforming a set list of
      time related annotation entities and grouping
      them.

      A baseline is taken as a global parameter.
   ...

      Tested with saxon, xalan,sablotron and xsltproc;
      call from commandline (Linux/Unix) with;
      DOS-systems in the appropriate way.

   Xalan:
   Call with
   java -cp /usr/share/java/xalan2.jar
         org.apache.xalan.xslt.Process
         -in XMLFILE  -xsl gen_lex_times.xsl
         -param baseline 3

   Saxon: call with
   java -cp /usr/share/java/saxon.jar
            com.icl.saxon.StyleSheet
         XMLFILE gen_lex_times.xsl baseline=3

   XSLTPROC:
   xsltproc -param baseline 3 gen_lex_times.xsl XMLFILE

   Sablotron
   sabcmd gen_lex_times.xsl XMLFILE \$baseline=3
   Requires XSLT 1.0
      -->
  <xsl:param name="baseline"/>
   <xsl:key name="wordsandrelations"
            match="relationpair"
            use="concat(baseelement/text(),'_',
               compelement/text(),'_',relation/@rel)"/>
  <xsl:template match="//struct_rel">
    <xsl:element name="lexicon">
      <xsl:for-each select="relationpair[
        baseelement and
        generate-id()=generate-id(
        key('wordsandrelations',concat(baseelement,'_',
                 compelement,'_',relation/@rel))[1])]">
```

```
        <xsl:if test="count(key('wordsandrelations',
                concat(baseelement/text(),'_',
                compelement/text(),'_',relation/@rel)))
                &gt; $baseline">
   <xsl:element name="lexentry">
            <xsl:attribute name="number_of_relations">
             <xsl:value-of select="count(key('
                                wordsandrelations',
              concat(baseelement/text(),'_',
              compelement/text(),'_',relation/@rel)))"/>
            </xsl:attribute>
            <xsl:attribute name="relation">
              <xsl:value-of select="relation/@rel"/>
            </xsl:attribute>
            <xsl:element name="baseelement">
              <xsl:value-of select="baseelement"/>
            </xsl:element>
            <xsl:element name="compelement">
              <xsl:value-of select="compelement"/>
            </xsl:element>
            <xsl:element name="baselayer">
             <xsl:value-of select="baselayerid"/>
            </xsl:element>
            <xsl:element name="complayer">
              <xsl:value-of select="complayer"/>
            </xsl:element>
   </xsl:element>
         </xsl:if>
       </xsl:for-each>
     </xsl:element>
   </xsl:template>
</xsl:stylesheet>
```

14.2.4 Sample generalisations over time relations

This results in a classified lexicon entry such as the following. It gives the segments with their related elements, a reference to the layer where they are derived from, the temporal relation and the number of times that this relation was found in the sample set of relations.

```
<lexentry number_of_relations="4" relation="contains">
      <baseelement>17m,7A,sy</baseelement>
      <compelement>PPOSAT</compelement>
```

```
      <baselayer>hands</baselayer>
      <complayer>esel_einfache_antefinal.POS</complayer>
   </lexentry>
<lexentry number_of_relations="13" relation="contains">
      <baseelement>17m,7A,sy</baseelement>
      <compelement>PPER</compelement>
      <baselayer>hands</baselayer>
      <complayer>esel_einfache_antefinal.POS</complayer>
   </lexentry>
<lexentry number_of_relations="6" relation="contains">
      <baseelement>17m,7A,sy</baseelement>
      <compelement>APPR</compelement>
      <baselayer>hands</baselayer>
      <complayer>esel_einfache_antefinal.POS</complayer>
   </lexentry>
```

15

TASX document grammar

Quite frequently the TASX data format was used for the annotation. This is the document grammar in DTD format for the TASX format.

```
<!--
      tasx DTD 28.8.2002
      Version 0.6

      Dr. Ulrike Gut,
      Dr. Jan-Torsten Milde,
      Thorsten Trippel

      Universitaet Bielefeld
      Fakultät für Linguistik und Literaturwissenschaft
-->

<!-- ENTITIES -->

<!-- session besteht aus beliebig vielen Schichten und
     einer globalen Kommentarebene -->
<!ELEMENT tasx          (session+)>
<!ELEMENT session  ((meta*, layer)*)>

<!-- Beschreibungschicht -->
<!ELEMENT layer ((meta*, event)*)>
<!ELEMENT event (#PCDATA|meta*)>

<!-- Metadaten -->
<!ELEMENT meta         (desc*)>
```

```
<!ELEMENT desc (name,val)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT val (#PCDATA)>

<!-- ATTRBUTES -->
<!ATTLIST meta
         m-id CDATA #REQUIRED
         access CDATA #IMPLIED
         level CDATA #IMPLIED
>

<!ATTLIST session
         s-id CDATA #REQUIRED
         day NUMBER #REQUIRED
         month NUMBER #REQUIRED
         year NUMBER #REQUIRED
>

<!ATTLIST layer
         l-id CDATA #REQUIRED
>

<!ATTLIST event
         e-id CDATA #REQUIRED
         start CDATA #REQUIRED
         end CDATA #REQUIRED
         mid NUMBER #IMPLIED
         len NUMBER #IMPLIED
>
```

16

Metadata lexicon

The following is a part of the DATR representation of the metadata of the corpus *Das Eselein* from the ModeLex corpus. The content is normalized for reserved characters and capitalization due to the DATR conventions for encoding of terminal symbols.

```
MODELEX_ESELEIN:
<>==Session.

METATRANSCRIPT:
<> == .

Session:
<> == METATRANSCRIPT
<Name> == esel_ger
<Title> == eselein
<Date> == 2000\-01\-30
<annotation date>==2000\-01\-30
<Description> == german fairy tale by grimm
<Organisation> == Contact
<Project Title> == Project.

MDGroup:
<> == Session.

Location:
<> == MDGroup
<Continent> == europe
<Country> == germany
```

```
<Address> == bielefeld.

Project:
<> == MDGroup
<Name> == modelex
<Project Title> ==
                 theorie und design multimodaler lexika.

Contact:
<> == Project
<Name> == dafydd gibbon
<Email> == gibbon@spectrum\.uni\-bielefeld\.de
<Organisation> == fakultät für linguistik
und literaturwissenschaft.

Collector:
<> == MDGroup
<Name> == ulrike gut.

Contact1:
<> == Collector
<Name> == ulrike gut
<Email> == ulrike\.gut@anglistik\.uni\-freiburg\.de
<Organisation> ==
              englisches seminar\, universität freiburg.

CommunicationContext:
<> == Content
<Interactivity> == non\-interactive
<PlanningType> == planned
<Involvement> == elicited.

Genre:
<> == Content
<Interactional> == story\-telling
<Discursive> == unspecified
<Performance> == oral\-poetry.

MODELEX_LAYER_META_COGEST2_FUNCTIONS:
<part of session> == modelex_eselein
<> == MODELEX_ESELEIN
<part of layer> == function
<layer name> == function
```

```
<layer_type> == cogest2 functions
<annotator name> == karin looks
<annotator native language> == german
.
```

A sample query resulting in a catalog entry with the annotators name, date, description, etc. is given in the following.

```
MODELEX_LAYER_META_COGEST1:<annotator name>.
MODELEX_LAYER_META_COGEST1:<annotation date>.
MODELEX_LAYER_META_COGEST1:<Description>.
MODELEX_LAYER_META_COGEST1:<Organisation>.
MODELEX_LAYER_META_COGEST1:<Project Title>.
MODELEX_LAYER_META_COGEST1:<layer_type>.
MODELEX_LAYER_META_COGEST1:<annotation revision>.
```

A sample query for the annotator of the CoGesT 2 tier is the following:

```
META_COGEST2_FUNCTIONS:<annotator name>
```

The result of this query is

```
MODELEX_LAYER_META_COGEST2_FUNCTIONS:<annotator name>
==> karin looks
```

References

Aliprand, J., Allen, J., Becker, J., Davis, M., Everson, M., Freytag, A., Jenkins, J., Ksar, M., McGowan, R., Muller, E., Moore, L., Suignard, M., and Whistler, K., editors (2003). *The Unicode Standard Version 4.0*. Addison-Wesley.

Allen, J. F. and Ferguson, G. (1994). Actions and events in interval temporal logic. Technical Report TR521. http://www.cs.rochester.edu/u/james/, checked December 2006.

Arntz, R. and Picht, H. (1989). *Einführung in die Terminologiearbeit*. Olms, Hildesheim, Zürich, New York.

Atkins, S., Bel, N., Bertagna, F., Bouillon, P., Calzolari, N., Fellbaum, C., Grishman, R., Lenci, A., MacLeod, C., Palmer, M., Thurmair, G., Villegas, M., and Zampolli, A. (2002). From resources to applications. Designing the multilingual ISLE lexical entry. In *LREC 2002*, pages 687–693, Las Palmas de Gran Canaria.

Atkins, S., Bel, N., Bouillon, P., Charoenporn, T., Gibbon, D., Grishman, R., Huang, C.-R., Kawtrakul, A., Ide, N., Lee, H.-Y., Li, P. J. K., McNaught, J., Odijk, J., Palmer, M., Quochi, V., Reeves, R., Sharma, D. M., Sornlertlamvanich, V., Tokunaga, T., Thurmair, G., Villegas, M., Zampolli, A., and Zeiton, E. ((undated)). Standards and best practice for multilingual computational lexicons and MILE (the multilingual ISLE lexical entry). Deliverable d2.2-d3.2 isle computational lexicon working group, International Standards for Language Engineering (ISLE), Pisa.

Babylon-Builder (2002–2004). Babylon-Builder. Software, information at http://www.babylon.com/, checked December 2006.

Babylon (undated) (undated). *Building a corporate Glossary*. Babylon Ltd., 10 Hataasiya Street, Or Yehuda, 60212, Israel, version 1.0 edition. Manual for the GLS format used for the Babylon system.

Barnard, D. T., Burnard, L., Gaspart, J.-P., Price, L. A., Sperberg-McQueen, C., and Varile, G. B. (1995). Hierarchical encoding of text: Technical problems and SGML solutions. In Ide, N. and Véronis, J., editors, *The Text Encoding Initiative: Background and Context*, pages 211–231. Kluwer Academic Publishers.

Barras, C., Manta, M., Antoine, F., and Galliano, S. (1998-2003). Transcriber – a tool for segmenting, labeling and transcribing speech. Software, download at http://trans.sourceforge.net, checked December 2006.

Bell, J. and Bird, S. (2000). A preliminary study of the structure of lexicon entries. In *Workshop on Web-Based Language Documentation and Description*, Philadelphia, USA.

Berners-Lee, T. and Fischetti, M. (1999). *Weaving the Web*. Harper, San Francisco.

Berners-Lee, T., Hendler, J., and Lassila, O. (2001). The Semantic Web a new form of web content that is meaningful to computers will unleash a revolution of new possibilities. *Scientific American*.

Biber, D., Conrad, S., and Reppen, R. (1998). *Corpus linguistics : investigating language structure and use*. Cambridge University Press, Cambridge.

Binckley and Guha (2003). RDF vocabulary description language 1.0: RDF schema. http://www.w3.org/TR/rdf-schema, checked December 2006.

Bird, S. (1995). *Computational phonology: a constraint-based approach*. Studies in natural language processing. Cambridge University Press.

Bird, S. and Liberman, M. (2001). A formal framework for linguistic annotation. *Speech Communication*, 33(1,2):23–60.

Bird, S., Ma, X., Lee, H., Maeda, K., Randall, B., Zayat, S., Breen, J., Martell, C., Osborn, C., and Dick, J. (–2004). AGTK: Annotation graph toolkit. Software, download at http://agtk.sourceforge.net/.

Bird, S. and Simons, G. (2002). Seven dimensions of portability for language documentation and description. In *Proceedings of the Workshop on Portability Issues in Human Language Technologies*, Paris. Third International Conference on Language Resources and Evaluation, European Language Resources Association.

Bird, S. and Simons, G. (2003). Seven dimensions of portability for language documentation and description. *Language*, 79(3):557–582.

Black, A. W., Clark, R., Richmond, K., and King, S. (2004). Festival speech synthesis system. TTS project, available at http://www.cstr.ed.ac.uk/projects/festival/,checked December 2006.

Boag, S., Chamberlin, D., Fernandez, M. F., Florescu, D., Robie, J., and Siméon, J. (2003). XQuery 1.0: An XML query language. http://www.w3.org/TR/xquery/ , checked December 2006. W3C Working Draft 02 May 2003.

Boersma, P. and Weenink, D. (–2004). Praat. http://www.praat.org, checked December 2006.

Bole-Richard, R. (undated). Je lis et j'écris l'éga. Available at the Ega-Resource site at Universität Bielefeld.

Boyer, L., Danielsen, P., Ferrans, J., Karam, G., Ladd, D., Lucas, B., and Rehor, K. (2000). Voice eXtensible Markup Language (VoiceXML). http://www.w3.org/TR/voicexml/ , checked December 2006. W3C Note 05 May 2000.

Bray, T., Hollander, D., and Layman, A. (1999). Namespaces in XML. http://www.w3.org/TR/REC-xml-names/, checked December 2006. W3C Recommendation.

Bray, T., Sperberg-McQueen, J. P. C. M., and Malerand, E. (2000). Extensible markup language (XML) 1.0 (second edition). http://www.w3.org/TR/REC-xml, checked December 2006. W3C Recommendation.

Brockhaus (1996 – 1999) (1996 – 1999). *Brockhaus. Die Enzyklopädie in 24 Bänden*. Bibliographisches Institut und F. A. Brockhaus AG, Mannheim.

Byrd, R. (1991). Computational lexicology for building on-line dictionaries: The wordsmith experience. *Computational Lexicology and Lexicography*, VI(I):117 – 137.

Büchel, G. and Schröder, B. (2001). Verfahren und Techniken in der computergestützten Lexikographie. *Lexicographica: Series maior*, 107:7–28.

Calzolari, N. and McNaught, J. (1996). Synopsis and comparison of morphosyntactic phenomena encoded in lexicons and corpora a common proposal and applications to European languages. Technical report, Expert Advisory Group on Language Engineering Standards (EAGLES).

Carson-Berndsen, J. (1998). *Time Map Phonology*. Text, Speech and Language Technology. Kluwer Academic Publishers, Dordrecht.

Chomsky, N. (1957). *Syntactic Structures*. Mouton, The Hague.

Chomsky, N. (1970). Remarks on nominalization. *Readings in English Tranformational Grammar*.

Chomsky, N. (1995). *The minimalist program*. MIT Press, Cambridge, Massachusetts; London, England.

Codd, E. F. (1970). A relational model of data for large shared data banks. *Communications of the Association for Computing Machinery*, 13(6):377–387.

Crowther, J., editor (1995). *Oxford Advanced Learner's Dictionary of Current English*. Oxford University Press, Oxford, fifth edition edition.

Crystal, D. (1967). Word classes in English. *Lingua*, 17:24–56.

Cutter, C. A. (1904). *Rules for a dictionary catalogue*. U.S. Bureau of Education. Special report on public libraries. Washington Government Printing Office, Washington, 4. ed. edition.

Daelemans, W. and Durieux, G. (2000). Inductive lexica. In van Eynde, F. and Gibbon, D., editors, *Lexicon Development for speech and language processing*, pages 115–139. Kluwer Academic Publishers, Dordrecht.

de Saussure, F. (1916). *Cours de linguistique générale*. Payot, Lausanne/Paris.

DeRose, et. al. (2001). XML linking language (XLink) version 1.0. http://www.w3.org/TR/rdf-schema, checked December 2006.

Draper, D. (2004). Mapping between XML and relational data. In Katz, H., editor, *XQuery from the Experts*, chapter 6, pages 309–334. Addison Wesley.

Drosdowski, G., Müller, W., Scholze-Stubenrecht, W., and Wermke, M., editors (1996). *Duden, Rechtschreibung der deutschen Sprache*. Bibliographisches Institut and F. A. Brockhaus AG, Mannheim, 21st edition.

Dublin Core (2003) (2003). DCMI metadata terms. http://dublincore.org/documents/2003/03/04/dcmi-terms/.

Dutch, R. A., editor (1962). *Roget's Thesaurus of English words and phrases*. Longmans, Green and Co Ltd., London and Colchester.

Dutoit, T. (1997). *An introduction to text-to-speech synthesis*. Text, Speech and Language Technology. Kluwer Academic Publishers, Dordrecht.

Ekman, P. and Friesen, W. V. (1978). *Facial action coding system*. Consulting Psychologists Press, Palo Alto.

ELAN (–2004). EUDICO linguistic annotator. Software, download at http://www.mpi.nl/tools/elan.html.

Encarta (2004). Encarta. CD-ROM by Microsoft Corporation. Encylopaedia.

Entropic Research Laboratory (1998). Xwaves. Software. no longer maintained.

Erlenkamp, S. (2000). *Syntaktische Kategorien und lexikalische Klassen. Typologische Aspekte der Deutschen Gebärdensprache*. Lincom Europa, München.

Eurodicautom (1973 – 2004). Eurodicautom: European Terminology Database. http://europa.eu.int/eurodicautom/, checked December 2006.

Evans, R. and Gazdar, G. (1996). DATR: A language for lexical knowledge representation. *Computational Linguistics*, 22(2):167–216.

Farrar, S. and Langendoen, D. T. (2003). Markup and the GOLD ontology. In *Workshop on Digitizing and Annotating Text and Field Recordings*, LSA Insitute, Michigan State University. Published at http://emeld.org/workshop/2003/langendoen-paper.pdf.

Fellbaum, C., editor (1999). *WordNet: an electronic lexical database*. MIT Press, Cambridge, Mass.

Fontenelle, T. (2000). Introduction: Dictionaries, thesauri and lexical-semantic relations. *International Journal of Lexicography*, 13(4):229–231.

Frath, P. (2003). Rules and intentionality in the generative lexicon. In *Proceedings of the Second International Workshop on Generative Approaches to the Lexicon*.

Freed, N. and Borenstein, N. (1996a). Multipurpose internet mail extensions (mime) part one: Format of internet message bodies. RFC 2045, IETF.

Freed, N. and Borenstein, N. (1996b). Multipurpose internet mail extensions (mime) part two: Media types. RFC 2046, IETF.

Gibbon, D. (1996). ILEX / DATR online compendium. List of DATR related resources, available at http://www.spectrum.uni-bielefeld.de/DATR/compendium.html, checked December 2006.

Gibbon, D. (2000). Computational lexicography. In van Eynde, F. and Gibbon, D., editors, *Lexicon Development for speech and language processing*, pages 1–42. Kluwer Academic Publishers, Dordrecht.

Gibbon, D. (2002a). Compositionality in the inheritance lexicon: English nouns. In Behrens, L. and Zaefferer, D., editors, *The Lexicon in Focus: Competition and Convergence in Current Lxicology*, pages 145–185. Peter Lang, Frankfurt a. M.

Gibbon, D. (2002b). Prosodic information in an integrated lexicon. In Bel, B. and Marlien, I., editors, *Proceeding of the Speech Prosody 2002 Conference, 11–13 April 2002*, pages 335–338, Aix-en-Provence. Laboratoire Parole et Langage.

Gibbon, D. (2005). Spoken language lexicography: an integrative framework. *Forum Translationswissenschaft*, 5:247–289.

Gibbon, D., Ahoua, F., and Connell, B. (2001). Ega endangered language documentation. http://www.spectrum.uni-bielefeld.de/langdoc/EGA/, checked December 2006.

Gibbon, D. and Lüngen, H. (2000). Speech lexica and consistent multilingual vocabularies. In Wahlster, W., editor, *Verbmobil: Foundations of Speech-To-Speech Translation*, pages 296–307. Springer, Berlin, Heidelberg, New York.

Gibbon, D., Mertins, I., and Moore, R., editors (2000a). *Handbook of Multimodal and Spoken Dialogue Systems: Resources, Terminology and Product Evaluation*. Kluwer Academic Publishers, Dordrecht.

Gibbon, D., Mertins, I., and Moore, R., editors (2000b). *Handbook of multimodal and spoken dialogue systems: resources, terminology, and product evaluation*, chapter SAMPA. Kluwer Academic Publishers, Dordrecht/New York.

Gibbon, D., Moore, R., and Winski, R., editors (1997a). *Handbook of Standards and Resources for Spoken Language Systems*, chapter SL corpus collection, pages 119–145. Mouton de Gruyter, Berlin and New York.

Gibbon, D., Moore, R., and Winski, R., editors (1997b). *Handbook of Standards and Resources for Spoken Language Systems*. Mouton de Gruyter, Berlin.

Gibbon, D., Moore, R., and Winski, R., editors (1997c). *Handbook of Standards and Resources for Spoken Language Systems*, chapter SL corpus design. Mouton de Gruyter, Berlin and New York.

Gibbon, D. and Strokin, G. (1998). ZDATR version 2.0, conforming to DATR standard library, RFC. http://www.spectrum.uni-bielefeld.de/DATR/Zdatr20.distribution/zdmanual/, checked December 2006.

Gibbon, D. and Trippel, T. (2000). A multi-view hyperlexicon resource for speech and language system development. In *Proceedings of the Second International Conference on Language Resources and Evaluation*, pages 1713–1718, Athens.

Gibbon, D., Trippel, T., Hell, B., and Gut, U. (2001 –2004). Theory and design of multimodal lexica. http://www.spectrum.uni-bielefeld.de/modelex/ , checked December 2006.

Gut, U. (2003). Learning the prosody of a foreign language. http://www.phonetik.uni-freiburg.de/leap/ , checked December 2006.

Gut, U., Looks, K., Thies, A., Trippel, T., and Gibbon, D. (2002). CoGesT conversational gesture transcription system version 1.0. Technical report, Universität Bielefeld, Universität Bielefeld. http://www.spectrum.uni-bielefeld.de/modelex/publication/techdoc/cogest/.

Halpern, J. (1993). *NTC's New Japanese English Character Dictionary*. National Textbook Company, Lincolnwood (Chicago).

Handke, J. (1995). *The Structure of the Lexicon: Human versus Machine*. Mouton de Gruyter, Berlin.

Hartmann, R. R. K. (2001). *Teaching and Researching Lexicography*. Applied Linguistics in Action. Pearson Education, Harlow.

Hausmann, F. J. and Wiegand, H. E. (1989). Component parts and structures of general monolingual dictionaries: A survey. In Hausmann, F. J., Reichmann, O., Wiegand, H. E., and Zgusta, L., editors, *Wörterbücher : ein internationales Handbuch zur Lexikographie*, Handbücher zur Sprach- und Kommunikationswissenschaft, chapter IV, pages 328 – 361. de Gruyter, Berlin/New York.

Heinrich, I. (2004). Entwicklung eines Konzepts zur Erkennung und Bereinigung doppelter Datensätze in einer Terminologiedatenbank. Master's thesis, Hochschule Anhalt für angewandte Wissenschaften (FH), Köthen.

Hell, B. (2003). *PERL API für die TAMINO*. Universität Bielefeld, Bielefeld.

Herbert Hasenbein, C. S. (2002). Online-Übersetzer. *CT Magazin für Computertechnik*, 13:170ff.

Ide, N., Lenci, A., and Calzolari, N. (2003). RDF instantiation of ISLE/MILE lexical entries. In *Proceedings of ACL 2003*, Sapporo.

IMDI Team (2003). Metadata elements for lexicon descriptions. MPI internal version, ISLE Metadata Initiative (IMDI).

ISLE Metadata Initiative Catalogue Metadata (2001). Metadata elements for catalogue descriptions draft proposal version 2.1. http://www.mpi.nl/world/ISLE/documents/draft/IMDI_Catalogue_2.1.pdf.

ISLE Metadata Initiative Session Metadata (2001). Metadata elements for session descriptions, draft proposal version 2.5. http://www.mpi.nl/world/ISLE/documents/draft/ISLE_MetaData_2.5.pdf.

ISO 10646 (2000). Information technology – universal multiple-octet coded character set (ucs) – part 1: Architecture and basic multilingual plane. Technical report, ISO/IEC.

ISO 12200:1999 (1999). Computer applications in terminology – machine-readable terminology interchange format (MARTIF) – negotiated interchange. Technical report, ISO.

ISO CD 24610-1:2003 (2003). Language resource management — feature structures —part 1: Feature structure representation. Technical report, ISO. Committee Draft.

ISO DIS 24610-1:2004 (2004). Language resource management — feature structures —part 1: Feature structure representation. Technical report, ISO. Draft International Standard.

Johnson, C. R., Fillmore, C. J., Petruck, M. R. L., Baker, C. F., Ellsworth, M., Ruppenhofer, J., and Wood, E. J. (2002). Framenet: Theory and practice. http://www.icsi.berkeley.edu/~framenet/book/book.html, checked December 2006. Version 1.0, checked May 28, 2003.

Kaplan, R. M. (1995). The formal architecture of lexical-functional grammar. In Dalrymple, M., Kaplan, R. M., III, J. T. M., and Zaenen, A., editors, *Formal issues in lexical functional grammar*, Stanford. CSLI Publications.

Kay, M. (2004a). Saxon basic edition . Available via http://www.saxonica.com, checked December 2006 Version 8.0 B.

Kay, M. (2004b). XQuery, XPath and XSLT. In Katz, H., editor, *XQuery from the Experts*, chapter 6, pages 145–184. Addison Wesley.

Kennaway, J. R. (2003). Experience with and requirements for a gesture description language for synthetic animation. In *Proceedings of the 5th International Workshop on Gesture and Sign Language Based Human-Computer Interaction*, Genova, Italy.

KIF (1998). Knowledge interchange format. http://logic.stanford.edu/kif/, checked December 2006.

Kilgarriff, A. (2001). Generative lexicon meets corpus data: the case of non-standard word uses. In *The Language of Word Meaning*, pages 312–328. Cambridge University Press.

Kipp, M. (2000-2003). Anvil — annotation of video and spoken language. Software, download at http://www.dfki.de/~kipp/anvil/, checked December 2006.

Kranstedt, A., Kopp, S., and Wachsmuth, I. (2002). MURML: A multimodal utterance representation markup language for conversational agents. In *Proceedings of the Workshop Embodied Conversational Agents - let's specify and evaluate them at Autonomous Agents and Multi-Agent Systems (AAMAS02)*, Bologna, Italy.

Labov, W. (1972). *Sociolinguistic Patterns*. University of Pennsylvania Press, Philadelphia.

Lamport, L. (1994). *A Document Preparation System: LaTeX*. Addison Wesley, second edition edition.

Lassila and Swick (1999) (1999). Resource description framework (RDF) model and syntax specification. http://www.w3.org/TR/REC-rdf-syntax, checked December 2006.

Lay, W. M. and Petrarca, A. E. (1970). *Modified double KWIC coordinate index: refinements in main term and subrodinate term selection*. Technical report series. Computer and Information Science Research Center, Ohio State University, Columbus, Ohio.

Leech, G. and Wilson, A. (1996). Recommendations for the morphosyntactic annotation of corpora. Technical report, Expert Advisory Group on Language Engineering Standards (EAGLES).

Leech, G. N. (1993). Corpus annotation schemes. *Literary and Linguistic Computing*, 8(4):275–281.

Lenci, A. (2003). Computational lexicons and the semantic web. Presentation at EUROLAN 2003.

MacEnery, T. and Wilson, A. (2001). *Corpus Linguistics: an introduction*. Edinburgh Textbooks in Empirical Linguistics, Edinburgh.

Martell, C. (2002). Form: An extensible, kinematically-based gesture annotation scheme. In *LREC Proceedings*, pages 183–187.

Matkins, M., editor (1994). *Collins English Dictionary*. HarperCollins Pulishers, third edition.

McKelvie, D. and Thompson, H. S. (1997). Hyperlink semantics for standoff markup of read-only documents. In *Proceedings of SGML Europe'97*, Barcelona.

McNeill, D. (1992). *Hand and Mind: What Gestures Reveal about Thought*. University of Chicago Press, Chicago and London.

Mehler, A. (2000). *Textbedeutung*. PhD thesis, Universität Trier, Trier. November.

Melby, A. K. (1995). *The possibility of language: a discussion of the nature of language with implication for human and machine translation*. John Benjamins, Amsterdam, Philadelphia.

Mengel, A., Dybkjaer, L., Garrido, J., Heid, U., Klein, M., Pirrelli, V., Poesio, M., Quazza, S., Schiffrin, A., and Soria, C. (2000). Mate dialogue annotation guidelines. Technical report, Multilevel Annotation, Tools Engineering (MATE). MATE Deliverable D2.1.

Messinger, H., editor (1978). *Langenscheidt GroSSwörterbuch Englisch-Deutsch*. Langenscheidt, München.

Milde, J.-T. and Gut, U. (2002). The TASX-environment: an XML-based toolset for time aligned speech corpora. In *Proceedings of LREC 2002*, pages 1922—1927, Las Palmas.

Morrel-Samuels, P. and Krauss, R. M. (1992). Word familiarity predicts temporal asynchrony of hand gestures and speech. *Journal of Experimental Psychology: Learning, Memory, and Cognition*.

Murray, J. A. H., Bradley, H., Craigie, W. A., and Onions, C. T., editors (1970). *The Oxford English dictionary*. Oxford University Press, Oxford.

Nolan, F., Ladefoged, P., Maddieson, I., Barry, M., Ball, M., MacMahon, M., and Esling, J., editors (1999). *Handbook of the International Phonetic Association*. Cambridge University Press, Cambridge.

OED online (2004). Oxford English dictionary. http://www.oed.com , checked December 2006.

Portele, T., Krämer, J., and Stock, D. (1995). Symbolverarbeitung im Sprachsynthesesystem HADIFIX. In *Proceedings of 6. Konferenz Elektronische Sprachsignalverarbeitung*, pages 97–104, Wolfenbüttel.

Prillwitz, S., Leven, R., Zienert, H., Hanke, T., and Henning, J. (1989). *HamNoSys. Version 2.0; Hamburg Notation System for Sign Languages. An Introductory Guide*. Signum, Hamburg.

Pustejovsky, J. (1995). *The Generative Lexicon*. MIT Press, Cambridge (Ma), London.

Quasthoff, U. (1997). Der deutsche Wortschatz — interaktive Sammlung. CD-ROM, Universität Leipzig, Institut für Informatik, Leipzig. DFG Projekt LAPT und DA, ca. 2.5 Mio wordforms, German.

Renear, A., Dubin, D., and Sperberg-McQueen, C. M. (2002). Towards a semantics for XML markup. In *Proceedings of the 2002 ACM symposium on Document engineering*, pages 119 – 126.

Richter, F. (1995 - 2004). German — English dictionary. http://dict.tu-chemnitz.de/, checked December 2006. Working Draft.

Rosenberg, E. L. (1997). The study of spontaneous facial expressions in psychology. In Ekman, P. and Rosenberg, E. L., editors, *What the Face Reveals*, pages 3–17. Oxford University Press, New York and Oxford.

Sag, I. and Wasow, T. (1999). *Syntactic Theory*. CSLI Publications, Stanford.

Sag, I. A., Wasow, T., and Bender, E. M. (2003). *Syntactic Theory*. CSLI Publications, Stanford, 2nd edition.

Sager, J. C. (1990). *A Practical Course in Terminology Processing*. John Benjamins Publishing Company, Amsterdam/ Philadelphia.

Salffner, S. (2004). Learning EGA: A linguistic frame of reference for primers for endangered languages. Master's thesis, Universität Bielefeld.

SARA (2002). SGML Aware Retrieval Application (SARA). Sofware for searching the British National Corpus(BNC), Available via http://sara.natcorp.ox.ac.uk/lookup.html, checked September 2004.

Sasaki, F., Witt, A., and Metzing, D. (2003). Declarations of relations, differences and transformations between theory-specific treebanks: A new methodology. In Nivre, J., editor, *Proceedings of the The Second Workshop on Treebanks and Linguistic Theories (TLT 2003)*. Växjö universitet.

Schiller, A., S.Teufel, and Thielen, C. (1995). *Guidelines für das Tagging deutscher Textcorpora mit STTS*. Universität Stuttgart, Institut für maschinelle Sprachverarbeitung and Universität Tübingen, Seminar für Sprachwissenschaft. Draft.

Schmidt, I. and Müller, C. (2001). Entwicklung eines lexikographischen Modells: Ein neuer Ansatz. *Lexicographica: Series maior*, (107):29–52.

Schmidt, T. (– 2004). EXMARaLDA. Software, download at http://www.rrz.uni-hamburg.de/exmaralda/, checked December 2006.

Scott, M. (1996-). Wordsmith tools. Oxford University Press. Concordancing Software for Windows.

Sharoff, S. (forthcoming). *Meaning as Use: a communication-centered approach to lexical meanings*.

Shieber, S. M. (1986). *An Introduction to unification-based approaches to grammar*. Number 4 in CSLI Lecture Notes. CSLI Publications, Stanford.

Shoebox (2002). The linguist's shoebox. Software,.

Silverman, K., Beckman, M. E., Pitrelli, J., Ostendorf, M., Wightman, C., Price, P., Pierrehumbert, J., and Hirschberg, J. (1992). ToBI: a standard for labelling English prosody. In *ICSLP-92*, volume 2, pages 867–870.

Simons, G. and Bird, S. (2002). OLAC metadata. http://www.language-archives.org/OLAC/metadata.html, checked December 2006.

Sinclair, J. (1991). *Corpus, Concordance, Collocation*. Describing English Language. Oxford University Press, Oxford.

Sjölander, K. and Beskow, J. (2000). Wavesurfer - an open source speech tool. In *Proceedings of ICSLP 2000*, Beijing, China.

Sjölander, K. and Beskow, J. (2004). Wavesufer. Software. Centre for Speech Technology (CTT) at KTH, Stockholm.

Sperberg-McQueen, C. M. and Burnard, L., editors (2001a). *TEI Guidelines*, chapter Terminological Databases. TEI Consortium.

Sperberg-McQueen, C. M. and Burnard, L. (2001b). TEI P4 guidelines for electronic text encoding and interchange. http://www.tei-c.org/P4X/index.html, checked December 2006.

Sperberg-McQueen, C. M. and Burnard, L., editors (2001c). *The XML Version of the TEI Guidelines*, chapter Print Dictionaries. The TEI Consortium, TEI P4 edition. http://www.tei-c.org/P4X/index.html, checked December 2006.

Spevack, M. (1973). *The Harvard concordance to Shakespeare*. Olms, Hildesheim.

Steininger, S. (2001). Labeling gestures in SmartKom - concept of the coding system. Technical report, LMU Munich, Munich.

Storrer, A. (2001). Digitale Wörterbücher als Hypertexte. *Lexicographica: Series maior*, (107):53–69.

Summers, D., editor (1992). *Dictionary of English Language and Culture*. Longman.

Tamino (2003). Tamino XML Server. Version 4.1.1.

TASX-Annotator (2002–2004). TASXAnnotator. Software, download at http://medien.informatik.fh-fulda.de/tasxforce.

Terrell, P., Kopleck, H., Holtkamp, H., and Whitlam, J., editors (1995). *Collins German Dictionary*. HarperCollins Pulishers, second edition edition.

Thompson et al. 2001 (2004a). XML Schema Part 1: Structures second edition. http://www.w3c.org/TR/xmlschema-1/, checked December 2006.

Thompson et al. 2001 (2004b). XML Schema Part 2: Datatypes second edition. http://www.w3c.org/TR/xmlschema-2/, checked December 2006.

Toolbox (2004). Field linguist's toolbox. Software, http://www.sil.org/computing/toolbox/index.htm, checked December 2006.

TreeTagger (1996). Treetagger - a language independent part-of-speech tagger. Software, download from http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/ , checked December 2006.

Trippel, T. (1999). Terminology for spoken language systems. Master's thesis, Universität Bielefeld, Fakultät für Linguistik und Literaturwissenschaft, Bielefeld.

Trippel, T. (2004). Metadata for time aligned corpora. In *Proceedings of the Workshop: A Registry of Linguistic Data Categories within an Integrated Language Repository Area, at LREC 2004*, Lisbon. ELRA.

Trippel, T. and Baumann, T. (2003). Metadaten für multimodale Korpora. Technical Document 3, Bielefeld University, Bielefeld. ModeLex Technical Document, Research Group Text Technological Modelling of Information.

Trippel, T. and Gibbon, D. (2001). PAX- an annotation based concordancing toolkit. In *Proceedings of the IRCS Workshop on Linguistic Databases*, pages 238–244, Philadelphia.

Trippel, T. and Gibbon, D. (2002). Annotation driven concordancing: the PAX toolkit. In *Proceedings of LREC 2002*, pages 1568—1572, Las Palmas.

Trippel, T., Gibbon, D., and Sasaki, F. (2004a). Consistent storage of metadata in inference lexica: the metalex approach. In *Proceedings of LREC 2004*, Lisbon. ELRA.

Trippel, T. and Salffner, S. (2002). *Recording Instructions*. Bielefeld University, Bielefeld, memo 3.1 edition.

Trippel, T., Sasaki, F., Hell, B., and Gibbon, D. (2003). Acquiring lexical information from multilevel temporal annotations. In *Proceedings of Eurospeech 2003*, Geneva.

Trippel, T., Thies, A., Milde, J.-T., Looks, K., Gut, U., and Gibbon, D. (2004b). CoGesT: a formal transcription system for conversational gesture. In *Proceedings of LREC 2004*, Lisbon. ELRA.

van Benthem, J. F. A. K. (1983). *The logic of time*. D. Reidel Publishing Company, Dordrecht.

van Eynde, F. and Gibbon, D., editors (2000). *Lexicon Development for Speech and Language Processing*. Kluwer Academic Publishers, Dordrecht.

Walsh, N. (1999, 2000, 2001, 2002, 2003). *DocBook: The Definitive Guide*. O'Reilly and Associates, Inc., Sebastopol, CA, USA.

Wells, J. C. (1990). *Longman Pronunciation Dictionary*. Longman, Harlow.

Witt, A. (2002a). Meaning and interpretation of concurrent markup. In *Proceedings of the ALLC / ACH 2002 - Joint International Conference of the Association for Literary and Linguistic Computing and the Association for Computers and the Humanities*, Tübingen. http://www.uni-tuebingen.de/cgi-bin/abs/abs?propid=40, checked December 2006.

Witt, A. (2002b). *Multiple Informationsstrukturierung mit Auszeichnungssprachen. XML-basierte Methoden und deren Nutzen für die Sprachtechnologie.* PhD thesis, Universität Bielefeld.

Witt, A., Lüngen, H., and Gibbon, D. (2000). Enhancing speech corpus resources with multiple lexical tag layers. In *Proceedings of LREC 2000*, pages 403–408, Athens.

Wittenbrink, H. and Köhler, W., editors (2003). *XML*, chapter Einführung – Was ist XML. TEIA Lehrbuch Verlag GmbH, Berlin.

Wittenburg, P., Gibbon, D., and Peters, W. (2001). Metadata elements for lexicon descriptions. IMDI Technical Report, ISLE Metadata Initiative (IMDI). Draft Proposal Version 1.0.

Wüster, E. (1991). *Einführung in die allgemeine Terminologielehre und terminologische Lexikographie*. Romanistischer Verlag, Bonn, 3rd edition.

XSLT (1999). XSL Transformations (XSLT) version 1.0. http://www.w3.org/TR/xslt, checked December 2006.

Zipf, G. K. (1935). *The psycho-biology of language*. Houghton Mifflin Company.

Index