A Data Dictionary as a Lexicon:

An Application of Linguistics in Information Systems

*J.F.M. Burg, R.P. van de Riet, S.C. Chang**

Department of Computer Science, Vrije Universiteit, Amsterdam
* Gemeenschappelijk Administratie Kantoor (GAK), Amsterdam
The Netherlands
{jfmburg, vdriet, thiel}@cs.vu.nl

ABSTRACT

Data Dictionaries (DD) contain crucial information about the (technical) meaning of words used in a certain company. In linguistics a lexicon contains syntactic and semantic information about words used in the society. In this paper we study the possibility of structuring a Data Dictionary as if it were a lexicon. The results of this study have successfully been applied to a Data Dictionary of a large Dutch company. It was possible to generate Natural Language (Dutch) sentences from the definitions of the words (concepts) in the lexicon, replacing the fixed strings originally put in the Data Dictionary.

Keywords : Lexicon, Data Dictionary, Linguistics, Natural Language Generation, Information and Communication System

1. Introduction

In the framework of the **LIKE** (Linguistic Instruments in Knowledge Engineering) project we have investigated the following problem: How can the construction of a **Data Dictionary (DD)** profit from Linguistic Knowledge? The experiences of this investigation will be described in this paper.

The LIKE project is a consortium of researchers from three disciplines: Linguistics, Business Administration and Computer Science and is focusing research around the theme: how linguistic instruments can be used profitably in the area of Knowledge Engineering. In a subproject **(LICS)** we investigate how such use can be made when designing and building Information and Communication Systems (ICSs for short). A part of that project, concerned with DDs, is the subject of this paper. A DD is normally used to define words occurring in the ICS, such as words for values, attributes, relations, programs, datastructures, triggers, etc. Mostly these words are used with a technical meaning which is not too far from their ordinary meaning: in an ICS where books can be borrowed by students, the words *borrow*, *book* and *student* are most probably used in accordance with their meaning: *book* and *student* for entities being related by the relationship *borrow*. It is quite natural to assume that these words also occur in the DD together with their meaning as expressed by the words *entity* and *relationship*. When also active components, such as transactions are described in the DD, it is likely that *student* is described as an active object which can initiate a *borrow* transaction and *book* as a passive object. When one compares this with what is usually written in an ordinary dictionary, the resemblance is striking. In a dictionary a student is described as a kind of person, while persons can borrow non-human things.

In an attempt to use the knowledge Linguists have put in dictionaries we have taken the following approach. Let the DD be considered as a Lexicon, i.e. a kind of dictionary where the words and meanings (concepts) are structured by *is_a, has_a* and other ordering notions. The contents are semantic and syntactic knowledge about ordinary and special words, typically dependent on the domain in which the DD is being used.

To make the experiment worth while an existing DD was chosen as the object for experimentation: the DD (called GBIS) of the Gemeenschappelijk Administratie Kantoor (**GAK**), which is an organization which governs the social laws in the Netherlands. Its databases rank amongst the biggest in the country.

The DD consists of more than 2500 entries which

CIKM '93 - 11/93/D.C., USA

© 1993 ACM 0-89791-626-3/93/0011 ....$1.50

are primarily being used to define in human-readable and understandable form, the meaning of the technical and non-technical words. These definitions consist of two parts: a *formal* one, which is a table of aspects of interest to the defined term, and a *verbal* one, which is a Natural Language sentence in which the term is defined using the aspects from the formal part.

The lexicon should have the same function as the current DD, i.e. it should be readable and understandable to humans. We have also, however, extended the DD with the capability to generate the verbal part of the definition automatically from the formal one. Originally this was not possible because the DD did not contain linguistic knowledge and secondly the formal part of the definition was not based on linguistic notions.

The main contribution of the research reported in this paper is, that we have investigated the structure of the Lexicon such, that on the one hand its contents can be extracted from an ordinary dictionary and on the other hand it is very easy to put technical words with their meanings and their relations to "ordinary" words and other technical words in it. In this way we made a Lexicon which can replace the current DD. In another project, which started recently, we investigate how the newly structured DD can be extended to be used in a tool which allows software-components to be reused.

In this paper we will describe the structure of the lexicon and we will show how it is used in GBIS. We structured the lexicon in a similar way to two other projects which were conducted in our group ([3] and [4]).

The paper first deals with Data Dictionaries in general and then shows some of the details of the DD in GAK. Next the requirements for the Lexicon will be studied. Then the structure of the Lexicon will be described, in particular the taxonomy chosen. Finally, we show how the lexicon can be incorporated in GBIS, the effectiveness of which will be demonstrated by some examples. The paper ends with conclusions and remarks about the research we are carrying out as a follow-up.

2. Traditional Data Dictionaries

A traditional DD contains definitions and descriptions of the primitive as well as user-defined data in the database it supports. In fact, it consists of a collection of *meta-data*, such as origins of data, data attributes, usage, location, format and relationships, i.e. data about the data produced, managed, exchanged and maintained in an organization [17]. This meta-data is used to maintain the constraints which are attached to the data from the database. The dictionary itself can also be considered as a database, because it is a repository of data and it is provided with software and procedures to create and maintain itself [21].

The traditional setting in which a Data Dictionary System (DDS), the software that manages the DD, operates is a DataBase Management System (DBMS) environment. The DD is *integrated* or *stand-alone*, depending on the strength of the relation with the DBMS. A stand-alone DDS has its own maintenance and reporting programs, while an integrated one is implemented as an application of and consequently dependent on a database or DBMS [14], [10]. The dictionary has the following functions in such a setting [20], [14]:

- the DD stores information about data and processes of interest to an organization.
- the DD describes the abstract conceptual structures from which the (or a possible) *internal schema* of the database can be derived.
- the DD serves as a common *reference point* for different users with varying views on the database.
- the DD plays an active role in the question and answer interface by locating semantic errors and providing *help and explanation* facilities.
- by constraining the possible uses of the predications, the DD contributes to maintaining data integrity.
- the use of a DD leads to a standardization of names, definitions and physical descriptions of the data elements used in programs.
- the DD describes which data is kept in which files, databases or schemas, it describes access-paths, reports of such and which modules and subprograms are included in which programs.

Examples of existing traditional DDs are *The Cullinet Integrated Data Dictionary* and the *IBM DB/DC Data Dictionary* [21], [14].

There is also a broader definition of a DD which says that a DD should store all the definitions of words used in a certain environment. An example is GBIS, the DD of GAK which stores definitions of words used in the social laws. This kind of data dictionary also contains NL-like sentences as strings. There is *no* knowledge about linguistic notions stored, there is just a field for the word itself, one for its description and other fields for the more technical information. This kind of DD is useful as a supporting tool in the software life cycle, because it achieves a certain degree of standardization in the usage of words.

GBIS is based on a term-defining method, **the aspects-method**, which consists of two parts: the *Formal* and the *Verbal* definition.

In the formal definition, the keywords, called *aspects*, are put in a framework (table). The verbal definition contains these aspects in the form of a Natural Language sentence (see Table 1).

There is an obvious problem: the same knowledge is represented in two different ways, which undoubtedly leads to inconsistent definitions, because there is no

automatic verification provided. And even when the two definitions are checked by hand, it is still a redundant way of storing information. By re-structuring the DD with linguistic knowledge and relations, this problem was tackled, because we were able to generate the verbal part automatically. Another advantage of our approach is the capability to check the input to the formal part for inconsistencies and other semantic errors, which was originally not possible. Our adjustments to the method will be explained in the next chapters.

1. Formal Definition	
Aspects	**Value**
Datum	code
Property Datum	-
First Object	person
Property Object	male
Attribute Object	nationality
Property Attribute Object	current
Second Object	administration
Property Object	-
Attribute Object	-
Property Attribute Object	-
First Role	register
Property Role	-
Third Object	GAK
Property Object	-
Attribute Object	R&D-department
Property Attribute Object	-
Second Role	work at
Property Role	-
2. Verbal Definition	
A code that represents the current nationality of a male person, who is registered by the administration, and who works at the R&D-department of GAK.	

Table 1. Complete GBIS-definition of the term: *nationality-code*

The way we see the DD is thus more **linguistically oriented**, such that the, extensive, knowledge from the field of linguistics can also be used in our own area, that of Information and Communication Systems (ICS's). We use linguistic knowledge, because this kind of knowledge describes everyday life in Natural Language. It is this everyday life that is more and more affected by the further developments in ICS's. We think that these developments and their effects are more controllable if we use linguistic knowledge at an early stage.

3. Traditional Lexicons

The **Lexicons** which have been designed and developed primarily by linguistic researchers contain information about words and the relations between those words. Linguists have distinguished two kinds of information about words: *syntactic* and *semantic* information. The content of these two kinds of information is shown in Table 2.

Kind of information	Consists of
Syntactic	category, stem, gender, irregular forms, etc. of the concept
Semantic	inter-concept relations essential characteristics stereo-typical examples

Table 2. Distinction between syntactic and semantic knowledge

When we talk about semantic knowledge (or information) about *words*, we mean actually knowledge about *concepts*. There is a linguistic difference between a *word* and a *concept*. A word is syntactic, a concept semantic, i.e. a concept is the *meaning* of a word. Consequently a word can be ambiguous (*book*), in contrast with a concept (*reading-book* or *to book*). The consequence of this is that a lexicon stores word-concept links. Where there is an ambiguous word (i.e. homonym) there should be enough knowledge available in the lexicon about the corresponding concepts and about the concepts used in the context (e.g. the verb), to determine which concept is meant by this specific word. Example: in "*The student borrowed a book.*", the word *book* refers undoubtedly to the concept reading-book. In general, a lexicon contains words as well as concepts, and an n-m relation between them.

There are also a lot of different lexical semantic relations [11] possible between *concepts* (and *words*). For linguists, the most familiar relations are:

synonymy (promise and pledge)

homonymy ((river-) bank and (money-) bank)

antonymy (hot versus cold)

while computer-oriented scientists use mainly (for example [19]):

taxonomy (lion is a mammal)

kind (penguin is a kind of bird)

cause (to send causes to go)

part (petal is part of flower)

sequence (Monday is always followed by Tuesday)

Our objective is to use both kinds of relations.

The part of the lexicon which contains the semantic knowledge is often called *Concept Lexicon* ([20], [15], [7] and [4]) and its structure consists of three hierarchical levels, which are depicted in figure 1. The three levels are:

1. *Ontology*

   At this level all basic (theoretical) notions of lexical semantic specification are introduced. Ontology concepts are the primitives of the specifications on the lower levels. In this sense the Ontology constitutes the top of a lattice. The root of the lattice differentiates into *State of Affairs* (verbs), *Objects* (nouns) and *Attributes* (adjectives). These concepts are further differentiated as well, see [4], [7] and [9] for more details.

2. *Common Sense*

   The Common Sense contains general concepts defined in terms of Ontology primitives. The knowledge present at this level can be found in ordinary dictionaries.

3. *Terminology*

   This level contains domain specific concepts defined in terms of general concepts, augmented with knowledge that is specific for the domain. At this level the actual application of the Lexicon takes place.

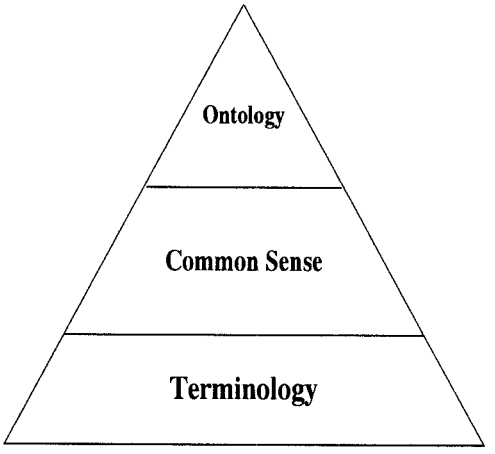

Figure 1. Levels in the Concept Lexicon

Later on, we shall use these ideas in a slightly adjusted form in an Information System. The traditional lexicons are used for:

- parsing and generating Natural Language text/messages.
- translating one language into another.
- reasoning by means of the semantic information. A very useful example is exploring the inheritance property of taxonomies, which will be discussed later.

Examples of these kinds of lexicons are described in [1] and [15].

4. Data Dictionary becomes Lexicon

We think lexicons should support the functions from the traditional data dictionaries as well as those from the traditional lexicons. To achieve this we had to define a lexicon-structure in which the necessary knowledge can be stored in a clear, readable and maintainable format. We have taken the structure of the traditional knowledge about data for granted, because research on this topic is extensive (see for example [21], [14]). This kind of knowledge can simply be incorporated in the structure we are going to present, by adding additional attribute-fields to the objects.

To complete the knowledge in the lexicon, we have added *articles*, *relative pronouns*, *demonstrative pronoun*, etc. This knowledge is necessary because the lexicon is supposed to generate and/or parse Natural Language sentences. In the future we will also add *inference-* and *expression-rules*. The latter define the meaning of words in so-called *Functional Grammar (FG)* [9].

5. Structure of the Lexicon

As proposed in several linguistic approaches to the structure of a lexicon [20], [15], we have made a distinction between **syntactic** and **semantic** information about words, which has been explained in Table 2. We have divided the semantic part as well. This was based on the *kind* of semantic knowledge, i.e. is it an *is_a* or a *has_a* relation and is the *has_a* relation *specific* or not?

As said before, a lexicon contains words as well as concepts, an n-m relation between them and some relations between concepts. In our approach however we have truncated the n-m relation to 1-n by forcing every specific problem domain to have its own definition of words and to store several meanings of the same word in separate parts of the lexicon (see the SCL-part of the lexicon). This restricts the meaning of a word to a specific one, in other words, a word points to just one concept. In the prototype we have implemented, we have truncated this relation to 1-1 by omitting synonyms (no two or more words for the same concept).

We have come to the following levels of the lexicon:

1. **Grammatical Lexicon (GL)**

   In this part of the lexicon *words* are stored, because grammatical knowledge simply belongs to words. The GL contains the syntactical information, such as *category* (noun, verb, adjective, etc.), *stem*, *irregular forms*, *gender*, etc. of the words. These properties are attached to the words in the form of attributes (i.e. *has_a*).

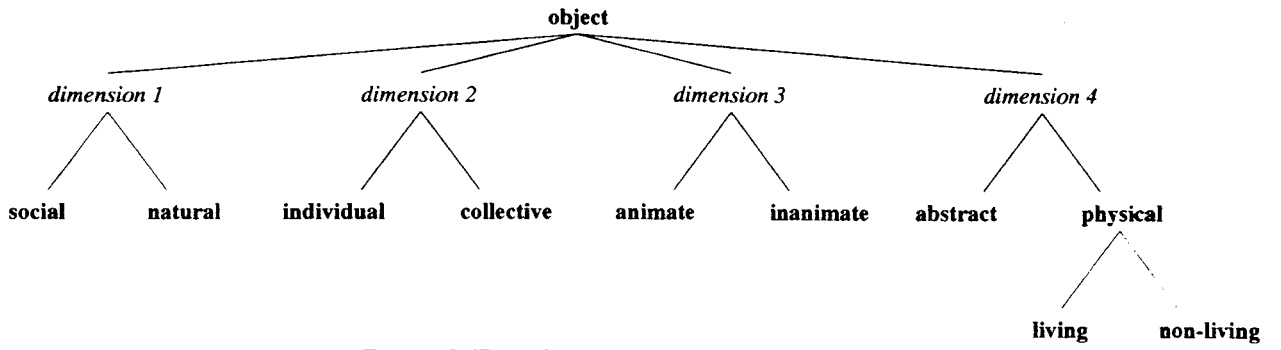

Figure 2. Top of taxonomy for objects (nouns)†

2. **Concept Lexicon (CL)**

As the name says, the CL contains information about *concepts* [15], because in this part the specific meaning of words is stored. This semantical meaning is divided in three parts, which are explained below.

2a. **Taxonomy Concept Lexicon (TCL)**

This contains the taxonomy relation (*is_a*) between concepts. For example: *a person is a mammal, a mammal is an animal, an animal is a living object.* This does not form a strict hierarchy. There are more dimensions at several levels of the (acyclic directed) graph. They are indicated as such, but do not occur explicitly in the lexicon of course. This problem is known as *cross-classification* [7] and can be implemented with multiple-inheritance (see, among others, [6]). For example: every object can be described according to four dimensions where every combination is possible (figure 2.†):

1. *natural versus social* (human-being versus police-officer)
2. *individual versus collective* (guilder versus money)
3. *animate versus inanimate* (company versus building)
4. *abstract versus physical* (time versus clock)

As figure 3. shows, this approach is very useful. The four 'base-classes' give the specific object its basic properties, whereas the combinations of the three new dimensions of *unemployment benefit* cover all the possible appearances of this term in a particular social law. These three new dimensions are derived from the social law-descriptions we used. *State of Affairs* are treated in a similar way, as figure 4. shows.

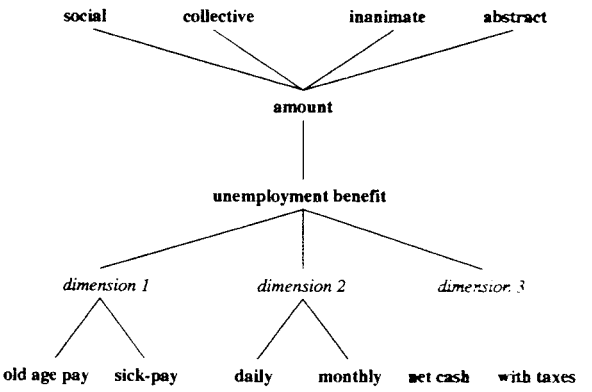

Figure 3. Part of taxonomy (inheritance-structure) for objects (nouns): *unemployment benefit*

2b. **Common Concept Lexicon (CCL)**

The CCL, which is indicated as *Common Sense* in figure 1., contains the knowledge everybody shares about concepts (common *has_a*). An example: *a person has a(n) age, length, etc.*

2c. **Specific Concept Lexicon (SCL)**

The SCL, the *Terminology*-part in figure 1., contains the knowledge specific for the problem domain from which the used concepts are collected (specific *has_a*). For example: *a person has a social security-number.*

A consequence of this approach is that for each problem domain there should be a different SCL, whereas the other components (GL, TCL and CCL) are universal.

† The words *dimension_1* etc. are just put there to explicitly show the different dimensions and therefor making the figure more readable. They are *not* incorporated in the lexicon.

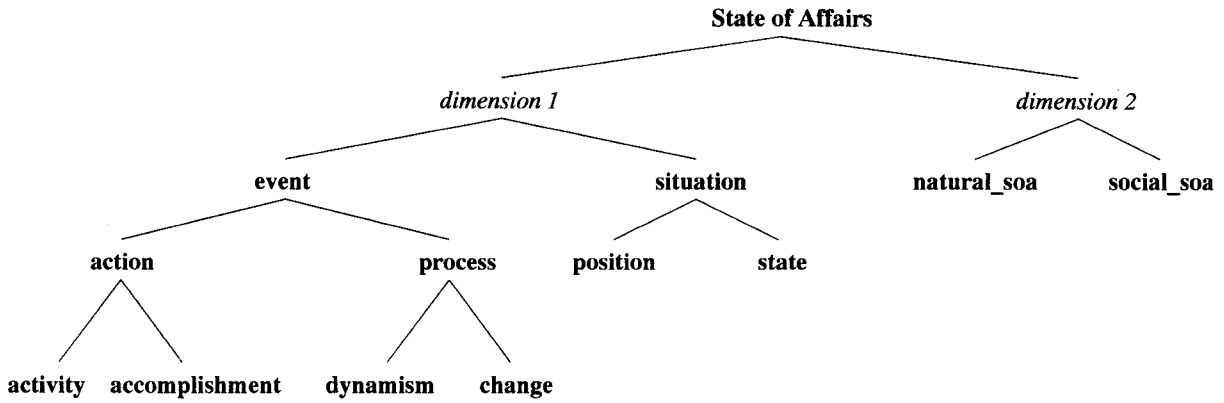

Figure 4. Top of taxonomy for State of Affairs (verbs)

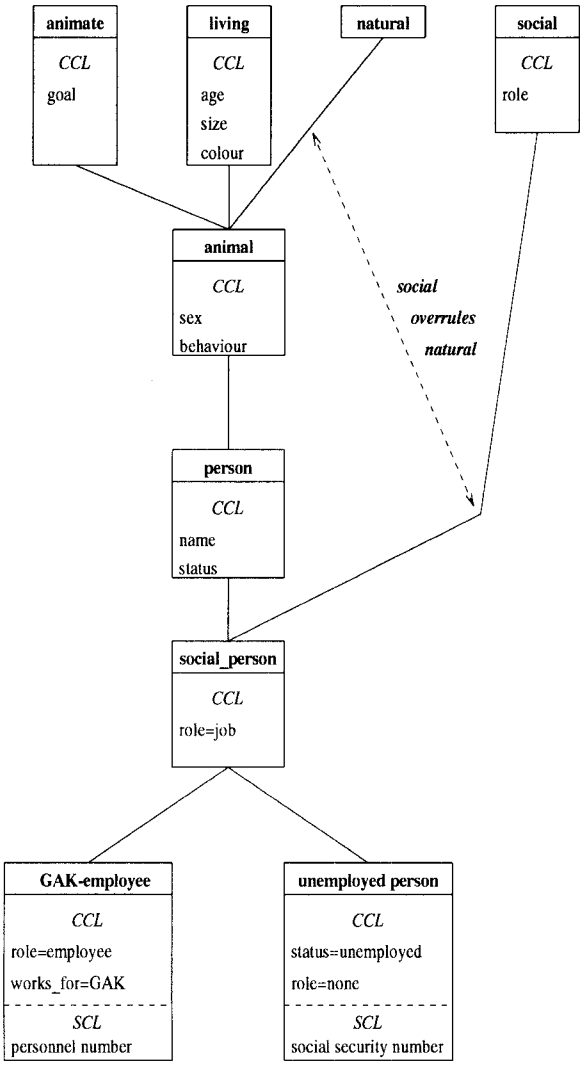

Figure 5. Two specific persons in the field of social laws

We have come to this structure by a *top-down* as well as by a *bottom-up* approach. The first, the top-down, approach consists of two parts:

1. we analyzed a number of existing lexicon-structures, taxonomies, hierarchies, ontologies and linguistic theories (see for example [20], [7] and [19]).
2. after these analyses, we developed our own taxonomy, based on the existing ones. The taxonomy is also based on the attributes contained in the concept-classes (see figure 5. and 6.).

We have complemented the top-down approach with a bottom-up one, to make sure our taxonomy and attributes-identification were useful. The concepts with which we started the bottom-up part are those, that are used by GAK to describe the social laws. These concepts have their own attributes as well, i.e. a person has a social security-number. These are purely specific for the problem domain, and appear in the SCL.

In figure 5. and 6. other parts of the lexicon-structure are displayed. They show two *is_a*-paths that are contained in the TCL-part of the lexicon. The attributes displayed in the boxes are syntactical (GL) or semantical (CCL or SCL). One of the possible uses of this lexicon (-structure) is the following: When a GAK-employee, who is working with GBIS, has to define a term which contains the verb *pay*, our prototype can decide if the employee entered *enough* objects (three objects required) and if the objects have the *right (sub-) type*. More examples can be found at the end of the next chapter.

As figure 5. shows, it is possible to *overrule* some inherited properties. The classes *animal* and *person* are *natural*, but when a persons role in society is examined, the *social* information about this person overrules the natural aspects. This makes only sense if the overruling and overruled class are each others counterparts.

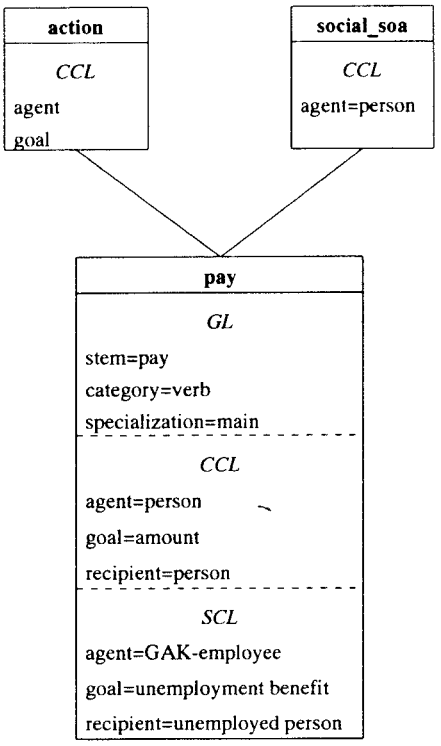

Figure 6. The role *pay*

A very important distinction has to be pointed out to understand the usability of the taxonomy we described. This distinction concerns *subtype* versus *instance*, both often described as *is_a*. The taxonomy as shown in the figures is purely a subtype-taxonomy. This means that a certain class inherits all the attributes from its ancestors, but it is still a class and not instantiated. When we are talking about an instance of a class, this instance contains all the attributes from its class, including the inherited ones, and values are put into most attributes.

An example in which this distinction is crucial is a *proper name*, which is an instance of a specific class (for example: John is an instance of person), but not a subtype. Such an instance should not inherit the grammatical knowledge for nouns from the GL, because the concept 'John' is not a normal noun (there is no plural form) but a proper noun. John inherits directly from the latter class, which has its own properties. When we simply say "*John is_a person*", this would mean that *John* inherits everything from *person*, including the knowledge from normal nouns. Therefore, we have to make the distinction between *is_a* (subtype) and *is_an_instance_of* (instance). This difference is analogous to the difference between the contents of a database (DB) and its data-dictionary. A DD contains the types of the concepts, whereas a DB contains instances of these types. In this paper we are only referring to the subtype-taxonomy.

6. Lexicon as part of an ICS

After having developed a structure for the lexicon, we shall now apply it.

The existing method (i.e. the framework for the formal definition) is adjusted in such a way that the verbal definition can be generated automatically. This generation can only take place when additional linguistic knowledge from the lexicon is used.

We had to adjust the framework because the existing aspects were not used consistently, and had therefore no strongly-defined linguistic meaning (such as noun, adjective, object, subject, etc.). This strongly-defined linguistic meaning is crucial for automatic NL generation.

In the new method, aspects have the meaning as shown in Table 3.

The rules we used to check if the aspects were right, were:

1. the aspects should be known to the lexicon, i.e. there exists an entry in the lexicon.
2. datum and objects should be nouns, i.e. their root-class should be noun.
3. the roles should be verbs.
4. the properties of datum, of objects and of object-attributes should be adjectives.
5. the properties of the roles should be adverbs.

The knowledge that is needed for these rules is available in the lexicon. These kinds of checks we have called *syntactic checks*.

There are also *semantic checks*, and these concern the meaningful combination of verb and nouns, i.e. the subject and the object belonging to a certain verb should be of a specific class. For example:

the verb *'give'* requires two persons as active elements (giver and recipient) and a passive object (which is given). The sentence *"The bike gives a person to a book"*, although syntactically correct, is semantically incorrect!

These semantic checks are exactly the reason why the taxonomy is so important. Because of the fact that a *GAK-employee* and a *woman* are both persons and a *book* is a physical object, the sentence *"The GAK-employee gives a book to a woman"* is also semantically correct.

Aspects	Meaning	Linguistic category
Datum	1. definable item 2. subject in main clause	noun
Property Datum	1. property of datum	adjective
First Object	1. object of which the datum is the attribute 2. object in main clause and subject in relative clause	noun
Property Object	1. property of object	adjective
Attribute Object	1. characteristic of an object	noun
Property Attribute Object	1. property of attribute of object	adjective
Second/Third Object	1. object that has a relation with first object 2. object or logical subject † in relative clause	noun
First/Second Role	1. relation between first and second/third object	verb
Property Role	1. property of role	adverb
Actor	which object is logical subject of the relative clause *first*: active relative clause *second/third*: passive relative clause	

Table 3. (Linguistic) meaning of the aspects.

For the automatic generation of the sentence, we need additional information in the form of articles, forms of the verbs used, etc. The following knowledge is retrieved from the lexicon:

1. the gender of the nouns, which determines the right article, relative and demonstrative pronoun.
2. the articles, relative and demonstrative pronoun are also retrieved from the lexicon.
3. the forms of the verbs depend on the *actor* as defined in the formal definition. If the actor is the first object, then the relative clause is active, so the verb should be in the third person singular. If the actor is the first or second object, then the relative clause is passive, and the verb should be in the past perfect form.

6.1. Implementation

We have opted for an *Object Oriented (OO)* approach (see for example [5] and [18]), because the OO-notions are very useful in representing words and their properties in computer-based systems. There is a natural correspondence between words and objects on the one hand and word-properties and attributes on the other hand. Besides this, we have used OO-inheritance to interrelate words.

The Natural Language generating part of the lexicon is implemented in Prolog. This SUN-windows based prototype has the following main functions:

1. Applying a user interface to the aspects-method to make entering the formal definition easy.
2. Verifying this formal definition for concepts:
   - not known to the lexicon
   - of the wrong category (e.g. a verb has been entered where a noun is expected)
   - that cannot be related (e.g. a *bike* cannot *give*)
3. Generating the verbal definition belonging to the entered formal one. This requires:
   - placing the right determiners, relative and demonstrative pronouns
   - finding the right form of verb determined by the actor of the sentence
   - determining the right auxiliary verb according to the kind of main verb (situation or event)
   - filling the sentence according to the given rules for a verbal definition
4. Supplying the user with auxiliary functions like:
   - entering a concept in the lexicon when it is not present
   - viewing the lexicon concepts and attributes
   - showing information about the aspects-method and how it should be used

† In linguistic theories a distinction is made between a *logical* and a *syntactic* subject. The form (third person singular, past perfect, etc.) of the verb depends on the syntactic subject, which is not necessarily the actor of the action (the logical subject). In the sentence *"a person is registered by the administration"* the person is the syntactic and the administration the logical subject.

Aspects	Example 1 (correct)	Example 2 (correct)	Example 3 (incorrect)	Example 4 (incorrect)
Datum *Property Datum*	number	name	name	name
First Object *Property Object* *Attribute Object* *Property Attribute Object*	person	person male surname	**male**	**person** male surname
Second Object *Property Object* *Attribute Object* *Property Attribute Object*	GAK	administration	administration	**R&D-department**
Role *Property Role*	work at	register	register	**marry**
Actor	first	second	second	second

Table 4. Examples of correct and incorrect formal definitions

6.2. Examples

To demonstrate the above mentioned function we shall present some examples (Table 4.). These examples also show how incorrect formal definitions are treated.

Verbal definition of Example 1

This formal definition is correct, and the relative clause is in an active format because the first object is the actor of the verb.

Number that represents a person,
who works at GAK.

Verbal definition of Example 2

This formal definition is also correct, but the relative clause is in a passive format because the second object is the actor of the verb.

Name that represents the surname of a male person,
who is registered by the administration.

Verbal definition of Example 3

There is no correct verbal definition, because the first object is not a noun, but an adjective. This formal definition contains a syntactic error.

Verbal definition of Example 4

There is no correct verbal definition in this case either, because a male person cannot be married to the R&D-department. Although this word is legal in the examined problem domain, its class is not a sub-class of *person*, which is the only class of objects that can occur in the relation *marry*, so this word cannot be used in this specific role. The inheritance-structure as defined in the TCL-part of the lexicon is used here to extract the ancestor-classes, of the current aspects, for which constraints exist in the lexicon. These constraints consist mainly of the restrictions on the semantic functions of verbs (i.e. *a book cannot give*). This formal definition contains a semantic error.

7. Extension of the Lexicon to be used for a re-usable software library

Currently, we (i.e. a number of students) are working with GAK on the problem of how to extend the Lexicon described in this paper to a knowledge base which acts as a library containing software components.

The idea is a simple one: the old GAK data dictionary, GBIS, contains the technical description of all the terms used in the information systems. As we argue in this paper, such a data dictionary should be looked at from a linguistic point of view, as a Lexicon where the words are connected to each other to reflect their meaning. This fits very well with the use of a common method, the *faceted classification scheme* (see [16]), which is used to store and find software components.

A library of software components consists first of all of a collection of these components, but, in the second place it has an access mechanism, through which one can search for appropriate components. That mechanism is based on descriptions, in the form of key-words, aspects, etc. In other words: words. Words which have to occur in the Lexicon anyway, because they have a technical meaning. In the case of GAK, we would have aspects like: to register, to administer, to pay, a person, a payment, month, WAO, etc. With these aspects we would like to search the knowledge base for a software component which takes care of the monthly payment to a person who is registered

as entitled to receive a certain social benefit WAO.

In using simple key-words we benefit from the fact that these words are connected: payment can only be carried out by an administrator to a person, every so often, e.g. per month. That information can be put in a natural way in the Lexicon, the same Lexicon which is able to generate (simple) sentences, as described in this paper.

In another paper we will report on this work. It remains to say that the structure of the (object-oriented) database for the software components is ready as is the query mechanism.

8. Conclusions

We have shown in this paper that the idea to use linguistic knowledge for structuring a DD is a viable one. It is possible to use the existing general purpose taxonomies and combine them with domain dependent taxonomies into a useful DD, called a Lexicon. From a DD built in this way it is possible to automatically generate Natural Language sentences to be read by human beings. The advantage of our approach to the Information System GBIS is twofold: Firstly, the stored definitions are syntactically and semantically correct. Secondly, the verbal and formal definitions of one term are mutual consistent, because only the formal one has to be entered, while the verbal one is automatically generated out of it. This generation of Natural Language sentences makes the system more efficient as well, because entering a new or adjusting an existing definition means only handling the formal part of it, which saves time and money.

References

[1] Akkerman, E., Masereeuw, P., Meijs, W., "Designing a computerized lexicon for linguistic purposes", Rodopi, Amsterdam, 1985

[2] Black, W.J., "Acquisition of Conceptual Data Models from Natural Language Descriptions", Proceedings of the 2nd Conference of the European Chapter of the ACL, Copenhagen, 1987

[3] Buitelaar, P., Riet, van de, R.P., "The use of a lexicon to interpret ER-diagrams: a LIKE project", Proceedings of the 11th International Conference on the Entity-Relationship Approach, Karlsruhe, 1992

[4] Buitelaar, P., Riet, van de, R.P., "A feasibility study in linguistically motivated object-oriented conceptual design of information systems", IR-293, Amsterdam, 1992

[5] Coad, P., Yourdon, E., "Object oriented analysis", Prentice Hall, Englewood Cliffs, 1990

[6] Daelemans, W., "Inheritance in Object-Oriented Natural Language Processing", Proceedings of the Workshop for Inheritance in Natural Language Processing, Tilburg, 1990

[7] Dahlgren, K., "Naive Semantics for Natural Language Understanding", Kluwer, Boston, 1988

[8] Dalianis, H., "A Method for Validating a Conceptual Model by Natural Language Discourse Generation", Advanced Information Systems Engineering, Proceedings of the 4th International Conference CAiSE'92, Springer-Verlag, Manchester, 1992

[9] Dik, S.C., "Functional Grammar", North-Holland, Amsterdam, 1978

[10] Duyn, van, J., "Developing a Data Dictionary System", Prentice-Hall, Englewood Cliffs, 1982

[11] Evens, M., "Relational models of the lexicon: Representing knowledge in semantic networks", Cambridge University Press, Cambridge, 1988

[12] Lefkovits, H.C., Sibley, E.H., Lefkovits, S.L., "Information Resource/Data Dictionary Systems", QED Information Sciences, Wellesley, 1983

[13] Leong-Hong, B.W., Plagman, B.K., "Data Dictionary/Directory Systems: Administration, Implementation and Usage", Wiley-Interscience publication, New York, 1982

[14] Mayne, A., "Data Dictionary Systems: A technical review", NCC Publications, Manchester, 1984

[15] Nirenburg, S., Raskin, V., "The subworld concept lexicon and the lexicon management system", Computational Linguistics, Volume 13, 1987

[16] Pietro-Diaz, R., "Implementing Faceted Classification for Software Reuse", Communications of the ACM, volume 34, number 5, 1991

[17] Pirri, F., "Data Dictionary Design: a Logic Programming Approach", Proceeding of the 11th International Conference on the Entity-Relationship Approach, Karlsruhe, 1992

[18] Rumbaugh et al., "Object oriented modelling and design", Prentice-Hall, 1991

[19] Sowa, J.F., "Conceptual Structures", Addison-Wesley, New York, 1984

[20] Weigand, H., "Linguistically motivated principles of knowledge base systems", Foris, Dordrecht, 1989

[21] Wertz, C.J., "The Data Dictionary: Concept and uses", North-Holland, Amsterdam, 1986