Thematic Role Based Generation of UML Models from Real World Requirements

Tom Gelhausen, Walter F. Tichy
Institute for Program Structures and Data Organization (IPD)
University of Karlsruhe, GERMANY
gelhausen@ipd.uni-karlsruhe.de, tichy@ipd.uni-karlsruhe.de

Abstract

Model-driven development depends on good initial models. Creating these models by hand is a challenging task, because of complex specification documents and change requests. We propose a new internal representation based on thematic roles, especially designed for (but not limited to) requirements documents. The representation can be generated automatically out of annotated real-world specification text and can be used to generate UML models based on graph rewriting rules.

1 Introduction

Our goal is to automate the transformation of software requirements into the models that currently form the starting point of the Model Driven Development (MDD) process. This paper presents the "Software Engineers' Natural language Semantics Encoding" (SENSE), a model with the ability to directly represent the content of natural language from a modelers point of view. Such an intermediate representation enables the separation of various demanding subtasks: the parsing and disambiguation of natural language, reasoning upon the statements, and the transformations from statements into diagrams as depicted in Figure 1. In this paper, we present the intermediate representation, SENSE, by means of the IETF WHOIS Protocol Specification (Section 2). Crucial for the encoding of semantics is a new set of thematic roles (Section 3). 'Supergraphs', a fundamental formalism we developed for SENSE (Section 4) simplify the representation of complex circular structures of real world specifications and the declaration of transformations upon those structures.

1.1 What is SENSE?

The intention of SENSE is to encode *semantics* of natural language – neither *syntax* nor *pragmatics* (in the sense of semiotics). That is: on the one hand, SENSE abstracts from things like word order, active or passive voice, or tense. On the other hand, SENSE has no understanding for language entities, for example what a 'library' or a 'book' really is.

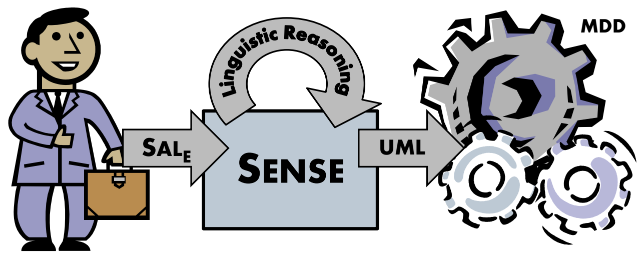

Figure 1. The Role of SENSE

This paper presents SENSE by means of a purpose-built annotation language, SAL_E. But SENSE is neither this annotation language nor the tool we have written doing the graph transformations: SENSE is a formalism, a way of representing the content of genuine natural language.

SENSE inherited the concept of n-ary associations with roles from Topic Maps [1], and the idea of self-assigning modifiers from Head-driven Phrase Structure Grammars (HPSG) [17]. SENSE is based upon supergraphs, an extension of hypergraphs where edges may not only connect to nodes but also to edges (cf. Section 4).

1.2 Related Work

There are numerous approaches on the general problem of transforming natural language into software artifacts, [2], [11], [12], [13], [19], for example, Moreno [16] discusses even more. The topic of this publication is SENSE – a formal representation of the content of natural language – along with transformations to and from this representation; so the nearest relative is the work of Dignum. His Conceptual Prototyping Language (CPL) [6] is based on Functional Grammar (FG) [7], SENSE is derived from HPSG. A major difference between SENSE and CPL is that CPL requires a verb for every association. This is undesirable as it requires the user to make up verbs in relations that do have an own verb: the term *Peter's house* is an example thereof. Furthermore, the real action or relation may be coded in a noun in conjunction with a meaningless verb as in *Peter* ***is*** *the* ***buyer*** *of the house* (instead of *Peter* ***buys*** *the house*). A verb centric encoding draws wrong conclusions in such cases. Intermediate representations such as CPL and SENSE have been rarely addressed.

Our work incorporates several concepts and approaches that can be regarded as state of the art. These are mentioned on the spot throughout the rest of this paper.

2 An Introduction to SENSE via SAL_E

This section introduces SENSE by means of a custom-made annotation language, the "SENSE Annotation Language for English" (SAL_E). It is compact and easy for a person to enter. An important design goal was that SAL_E should be minimally invasive: Predominantly, the initial specification text can be kept as is and only annotations need to be added.

The working example for this section is Chapter 2 ('Protocol Specification') of the IETF WHOIS Protocol Specification[1] [3]. Listing 1 shows the already encoded specification text. The following paragraphs explain the annotations.

Listing 1. Chapter 2 of the WHOIS Protocol Specification in SAL_E

```
1  [ #A WHOIS_server|{AG,RECP} listens|ACT #on
2    TCP_port_43|PAT #for requests|HAB #from
3    WHOIS_clients|DON ]
4
5  [ [ #The WHOIS_client|{AG,DON} makes|ACT #a
6      text_request|{HAB} #to #the
7      @WHOIS_server|RECP ]|SUM
8    #then #the @WHOIS_server|{AG,DON} replies|ACT
9    #with text_content|HAB ]
10
11 [ #All @requests|PAT #are terminated|ACT #with
12   { ASCII_CR , #and #then ASCII_LF }|INST ]
13
14 [ [ #The response|OMN #might contain|ACT #more
15     #than #one
16     [ ^line|QUALII #of text|QUAL ]|PARS ]|CAU
17   #so [ #the presence|ACT #of { @ASCII_CR ; #or
18       @ASCII_LF }|PAT #characters ]|NOT
19   #does #not indicate|!ACT
20   [ #the ^end|HAB #of #the @response|POSS ]|STIM ]
21
22 [ #The @WHOIS_server|AG closes|ACT
23   [@WHOIS_server|POSS ^connection|HAB]|PAT
24   #as #soon #as [ #the output|PAT #is
25   finished|ACT ]|TEMP_POS ]
26
27 [ #The $closed @connection|STIM #is #the
28   indication|ACT #to #the @WHOIS_client|EXP
29   #that [#the @response|PAT #has #been
30   received|ACT]|NOT ]
```

2.1 Description of the Annotations

Let's look at the first sentence: *A WHOIS server listens on TCP port 43 for requests from WHOIS clients.* It contains two aspects: The first aspect relates to acting. Acting requires an agent (AG), an action (ACT) and something acted upon (PAT). The second aspect concerns passage, namely passing something (HAB) from a donor (DON) to a recipient (RECP). The **thematic roles**, AG, ACT, PAT, DON, HAB, and RECP, are appended to the corresponding terms. We say these entities 'play a certain role' in their clause. Compound[2] words are joined using the underscore character ('_'), as in *TCP_port_43.* Words prefixed with a hash symbol ('#') are **commented out**; they are either superfluous (like *#A...*) or their meaning is encoded in a semantic role (*#from* is encoded in DON).

To **delineate clauses**, they are enclosed in square brackets ('[ ]'), which makes it clear how clauses are nested. Regard the second sentence: The outer clause describes an action (ACT). This action has a precondition (sumptio, SUM). The subordinate clause in line 5 trough 7 is as a whole the precondition for the outer clause; thus the entire clause is annotated with the thematic role SUM.

However, it is not always the case that the subclause as a whole plays a role in the outer clause. For example a nested relative clause would characterize one of its entities, and it is that entity (not the relative clause) that plays a role in the outer clause. This is also the case in the subclause in line 14 through 16: The subclause is nested into an outer clause, but it is the word *line* that plays the role PARS in relation to the *response* (omnium, OMN). This **lifting** of the head[3] of a clause is indicated by a caret ('^').

The second sentence also shows the usage of a **reference**: Words prefixed with an '@' refer to previously introduced entities, words without it declare new entities. For instance, *WHOIS_client*, *text_request*, *text_content*, *makes*, and *replies* in the second sentence declare new entities, and

[1] We applied minor modifications to the text as these address coreference resolution, an issue beyond the scope of this paper: (a) in the last but one sentence the pronoun ('its') has been replaced by the noun it refers to ('WHOIS server') and (b) in the last sentence, two synonyms ('TCP connection' and 'client') have been replaced by the terms that was used earlier in the text ('connection' and 'WHOIS client').

[2] The rules of *word formation* say that *science fiction* for example is *one* word build from *two* 'morphemes'. A character-based tokenizer as in our ANTLR-parser is thus not able to identify *words* of the English language.

[3] as it is called in HPSG

@WHOIS_server refers to the entity that has already been introduced in the first sentence.

The third sentence contains a **set** which is indicated by braces ('{}'): *ASCII CR* and *ASCII LF* (line 12), both play the role of an instrument (INST) in the termination mentioned there. Another example for a set can be found in line 17 and 18: Again, *ASCII CR* and *ASCII LF* take a role together. But in this case, the delimiter symbol is a semicolon ('`;`') instead of a comma ('`,`'). These two examples show the general difference between entering AND- and OR-sets in SAL$_E$: In AND-sets the elements are separated by commas ('`,`'), OR-sets use a semicolon ('`;`'). Sets may also be nested.

The entities declared in a specification statement may have properties like a *traffic light* that is *red* or an *apple* that is *green*. These properties would normally be represented by node attributes in an ordinary graph. Yet the attributes that may be declared in a specification document are not known in advance, so the graph model cannot incorporate them. SENSE utilizes so called **dynamic attributes**: They have a name and a value like ordinary attributes. Normally, we only use boolean attributes denoted by their attribute names; they get the value `true` if specified, `false` when specified with an exclamation mark (see below) or else `undefined`. Dynamic attributes differ from ordinary attributes in that they store an additional context and that they do not need to be (and cannot be) declared a priori (i.e. in a meta-model).

The need for context sensitive attributes comes from the observation that few attributes are really absolute in the real world. Regard the `$closed @connection|STIM` in line 27: The according connection is probably not always closed – in fact it has just been closed in the preceding sentence (line 22 to 25). So we require a context for every attribute to keep the information in which context it holds. This context is (initially) the subclause in which the attribute occurred. If no context is given, the attribute is omnipresent, i.e. it holds for all contexts in the discourse.

The syntax for the declaration of dynamic attributes is borrowed from natural languages: They accumulate the attributes (i.e. adjectives and adverbs) of an entity either in front or behind this entity. So in SAL$_E$, all dynamic attributes declared *in front* of entities (identifiable by an appended role declaration via vertical bar symbol and role name) pertain to the next following entity. If no entity declaration or entity reference follows the declaration of the dynamic attribute it pertains to the enclosing (sub-) clause.

Dynamic attributes cannot only be assigned to entities, they can also be assigned to other dynamic attributes: a *car* can be *fast*, but it can also be *terribly fast*. In this case, the adverb *terribly* does not describe the *car* but the adjective *fast*. If an entity is to be assigned more than one dynamic attribute (i.e. a list of adjectives), all these attributes are enumerated in front of the entity, separated by commas (or semicolons, depending on the type of the set) and enclosed by braces. If a dynamic attribute A (i.e. an adverb) is to pertain to another dynamic attribute B, A is placed right in front of B. So a `$terribly $fast car|PAT` is a car that is terribly fast and a `${terrible,fast}car|PAT` is a car that is both, terrible and fast.

Line 19 shows the notation of a **not** in SAL$_E$. It is marked by an exclamation mark ('`!`') and can be used in various situations: It can be used to mark a role as negative, for example a `FAV` is a favor and a `!FAV` is a disfavor (see Appendix A). The exclamation mark can also be used in sets to mark that an entity (or subset) is *not* a member. And finally, the exclamation mark can be used on dynamic attributes to denote that the entity does *not* have this property.

2.2 Transformations

SENSE has been designed to serve as an intermediate representation for the process of transforming natural language statements into software artifacts. This section serves to show the fitness of SENSE for this task, not to discuss the transformation from SENSE to software artifacts per se.

Our current workflow to create software artifacts from natural language specifications via SENSE is as follows: Firstly, the user annotates the plain text specification statements according to Section 2.1 using an ordinary text editor. The SAL$_E$-document thus created is automatically translated into a graph definition for the common graph rewrite system GrGen [10]: The annotated text is actually a supergraph definition (SENSE is a special kind of supergraph). A supergraph contains edges between edges, and an edge may connect more then two nodes or edges (cf. Section 4). It can be translated into an ordinary graph, where superedges are represented by special nodes. Figure 2 shows the translation of the WHOIS-supergraph into a regular graph. The first sentence is represented within the light gray nodes.

As we strive for the independence from the style of the input text, the next step could be the application of rules to normalize the SENSE-graph. An example where this seems both, eligible and feasible, can be found in the text: The fifth sentence defines an action (*The WHOIS server **closes** its connection...*) and the sixth sentence refers to that action (*The **closed** connection is the indication...*). Linguistically, the adjective of the sixth sentence is created by 'derivation' (a special kind of word formation) of the verb in the fifth. A rule reverting such derivations could reveal this connection. We call this process 'linguistic reasoning' (cf. Figure 1).

The final step is to generate the desired software artifacts from the graph. To demonstrate the fitness of our approach, we implemented a set of rules that transform a SENSE-graph into a class diagram. One of these rules can be seen in Listing 2. It creates an aggregation between two

Listing 2. A supergraph transformation rule

```
1  rule OMNPARS2agr {
2    pattern {
3      a:Class; b:Class;
4      z[x|OMN y|PARS];
5      if { x.NAME==a.NAME; y.NAME==b.NAME; }
6      negative {
7        :Aggregation[a|aggregate b|component];
8      }
9    }
10   replace {
11     a; b;
12     z[x|OMN y|PARS];
13     :Aggregation[a|aggregate b|component];
14   }
15 }
```

classes, if their corresponding entities were in a whole-part-relationship in SENSE. It is written in another custom-made language for creating rules containing superedges [4].

Let's assume we allready created *class*-nodes for all entities. The rule shown in Listing 2 searches for two classes a and b and a clause z that contains at least two entities x and y, x playing the role of a whole (omnium, OMN) and y playing the role of a part (PARS). We require that the names match (line 5) to ensure a and b will be the right classes (otherwise they would match *any* class). Line 6 through 8 contain a negative application condition, i.e. a condition that will keep the rule from executing if it is fulfilled. We need it to ensure the rule is not executed if there already exists an aggregation edge between a and b; for instance this would be the case if this rule has already been applied to these two classes. Line 10 through 14 declare the replace-part of the rule: Everything that occurs in this part will either be will be created if it occurs for the first time or it will be kept if it already occurred in the pattern-part (line 2 through 9). Everything that does not reoccur is deleted from the graph. Thus, line 11 and 12 only repeat elements from the pattern-part to keep them from being deleted. Line 13 creates the element we seek for: It creates a superedge of the type *Aggregation*. This newly created edge connects the classes a and b, whereas a plays the role of the *aggregate* and b the role of the *component*. When applied to the SENSE-graph of the WHOIS specification, the resulting aggregation can be found in the parallelogram in the upper right corner of Figure 3.

Apparently, the rule in Listing 2 is formulated generically enough to be applicable not only in our example but also in cases where a whole-part-relationship is annotated in other specification texts. We expect that a modest set of rules will suffice for many application domains.

For this demonstration, we implemented 19 rules total (about 330 LOC, including initialization and clean up rules)

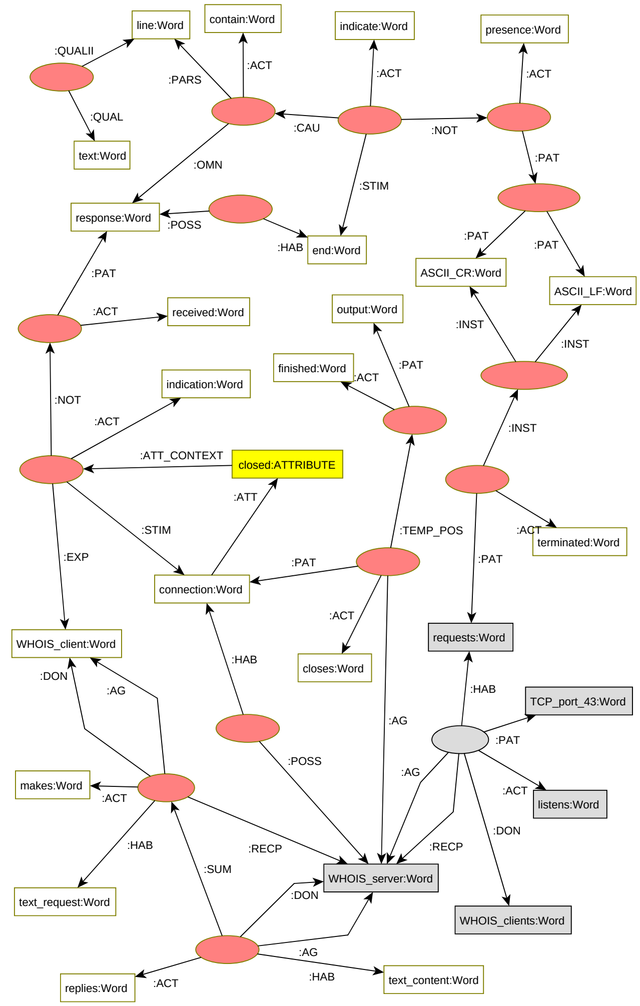

Figure 2. The internal representation of the WHOIS specification in GrGen

to create an exemplary class diagram from the working example. GrGen executes them in about 60 ms and creates the graph shown in Figure 3.

3 Thematic Roles

The semantics representable by SENSE are coded via graph contiguity and roles. Like Topic Maps [1], we need roles to assign semantics to n-ary associations. But in contradiction to Topic Maps, SENSE does not use an arbitrary user defined set of roles. This would hinder the development of a general computational transformation. We propose to use a common set based on *thematic roles*[4].

[4]'semantic roles', 'thematic roles', 'θ-roles', and 'theta-roles' are all found in literature and name the same thing

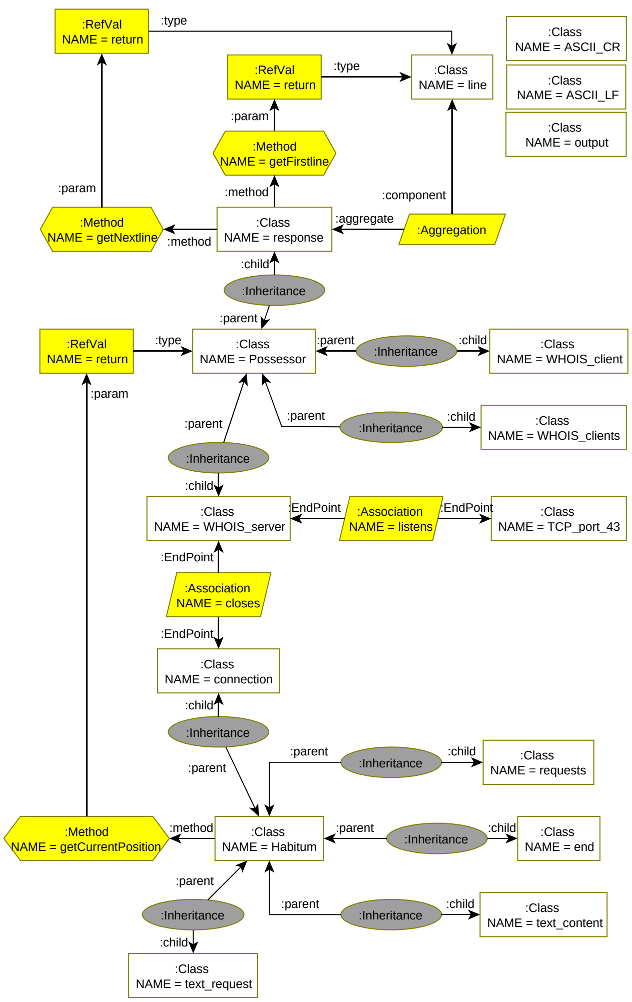

Figure 3. Result of the Transformation from SENSE into a Class Diagram

Some of the approaches mentioned in Section 1.2 are based on the interpretation of the *syntactic roles* (i.e. subject, predicate, and object) of a statement. These roles are usually easy and safe to determine automatically. Yet, Fillmore demonstrated that syntactic roles have no semantic meaning [8] and may thus lead to misinterpretations. Stimulated by his work, linguists developed the theory of thematic roles [18]: They are nowadays widely accepted as link between the syntax and semantics of natural language.

Take the sentence *The key is used by the janitor to open the door*: The *key* is an instrument (INSTR, see Appendix A), a tool that is used by someone to do something. That person acting is the *janitor* (AG). Finally the *door* is the object being treated (PAT). A purely syntactic approach would lead to a different analysis here.

It is a common assumption that thematic roles may occur at most once in a clause to make sense. It is important to notice that this **does not forbid** (a) that an entity may take more than one role and (b) that more than one entity can take a certain role. Our example in Section 2 has an example for the former (the *WHOIS server* in line 1 takes two roles) and the latter (*ASCII CR* and *ASCII LF* both take the role INST in line 12). The latter is possible if and only if the entities occur lexically grouped and form a pragmatical unit.

A question arising at this point is whether thematic roles are automatically identifiable. The answer is yes – to a certain extend. In all investigated languages thematic roles are mostly encoded through a combination of apposition(s) and case (sometimes verb-dependent). The Babel-System, a HPSG-based full parser for natural language [14], demonstrates the feasibility of an automatic identification. However, it provides a very coarse set of roles, unsuitable for our system.

3.1 A new Set of Thematic Roles

Some other approaches mentioned in Section 1.2 take thematic roles into account[5]. The main problem of these approaches is that their sets of thematic roles are relatively coarse. An orderly classification of real world natural language statements is rather difficult with them.

For SENSE, we developed a new set of 49 thematic roles, shown in Appendix A. Among them are several new roles like ‘comparand’ (a thing being compared with), or ‘intentio’ (like ‘causa’ an impetus for an action, but with a different temporal relation). The set was derived from sets found in linguistics literature ([15] and [18]). It was then complemented by the roles directly identifiable through the 21 Hungarian cases found in [20] and all possible combinations of appositions and cases in German. Finally, it has been ‘deictically’ closed, as described below.

3.1.1 What is ‘Deixis’?

Deixis is a linguistic concept. In natural language, some words rely on their context: What is meant by the word *there* can only be understood in conjunction with the sentence(s) around it. The theory of deixis explains how such words are to be interpreted. A deixis spans a referral space for the listener’s orientation in the discourse. Multiple deixis span a multidimensional room. In each deixis, one distinguishes different localities: proximal, medial, and distal. For the personal deixis, *I* and *we* is proximal, *you* is medial and *he*, *she*, *it*, and *they* is distal.

[5] These approaches might thus as well benefit from the new set of thematic roles presented in this paper.

3.1.2 Deictic Closure

The basic idea about the *deictic closure* is that if there is a *there*, there must also exist a *here*. We mapped the roles we identified to that point into as few[6] different deixis (or 'deictic axes') as possible, particularly by giving a negative extension to some of these axes. Afterwards, we tried to complement each position on the axes. Very often the proximal was missing – 'missing' in the sense that we did not recognize the concept until then; the role itself was preexisting, of course. In most of these cases there was not even a word for the proximal concept, so we needed to make up many artificial names. We coined the word *idem ipso* to subsume the proximals; the postfix '-II' in the aberration of a role name indicates that it is proximal. The deictic closure of the initial set led to a set of about 40 roles.

3.1.3 Evaluation of the Set of Thematic Roles

We evaluated the deictically closed set by testing it with human subjects on manually chosen sentences from software specification documents, fairy tales and news. Each participant was introduced to the topic of thematic roles for about half an hour. Then, the participant assigned thematic roles from the given list to 23 sentences of ascending complexity. Afterwards, we interviewed the participant in a semi-structured interview on that task. Incorporating the results, we revised the list. We repeated this sequence with new subjects until no further roles were proposed or estimated to be dispensable: The first seven iterations lead to changes in the list, the following four iterations did not.

This preliminary study provided a suitably complete set of thematic roles for our work. Yet, using the set in our application, we identified three roles that might be missing: a role named 'status' indicating that a verb marks a state rather than an action; and the two roles PRE and SUCC expressing a temporal relation. Whether these are truly new roles or only aspects of existing ones, and whether there are further ones, is currently under investigation.

4 Supergraphs

We chose to regard the task of translating natural language statements into software artifacts as a compiler construction problem. We use the more general graph rewrite systems instead of the conventional term rewrite systems. Their advantage is their inherent support for circular structures which often occur in natural language as well as in software models.

A problem about ordinary graphs is that their expressive power is far below that of natural language. Even the simplest style of speech enables us to relate more than two objects: The first sentence of the WHOIS specification (*A WHOIS server listens on TCP port 43 for requests from WHOIS clients*) already relates five objects[7]. Yet there is no single structure in ordinary graphs that is able to *directly* express higher order relations; we would need to switch to hypergraphs. But it is as simple to exceed the capabilities of hypergraphs as well: The fourth sentence of that same specification (*The response might contain more than one line of text so the presence of ASCII CR or ASCII LF characters does not indicate the end of the response*) relates a precondition with an action – and that precondition is a relation itself. This sentence would require us to draw an edge between nodes and another edge. But hypergraphs are not capable of connecting edges to edges – supergraphs can.

4.1 The Formalism – Informal

A supergraph consists of a set of nodes, a set of edges, and a set of roles. The set of nodes and the set of edges together form the set of 'connectible objects'. An edge connects one to many connectible objects. Each 'endpoint' of an edge (in hypergraphs they are called 'tentacle') is marked by one element from the roles set – this is the role the adjacent connectible object plays in the relation. Nodes and edges may have attributes. They have an identity so equality in attributes is not a problem and we can have multiple edges between the same choices of connectible objects. Recapitulating, supergraphs are attributed multi-graphs that allow edges with arbitrary numbers of tentacles connecting not only nodes but also edges. The formal description can be found in [5].

The basis of our supergraph rewrite system is an ordinary graph rewriting system named GrGen [10]. Thus, the information that a supergraph can represent is also representable by ordinary graphs. The major difference is that a supergraph is considerably more compact than an ordinary graph containing the same information.

4.2 Adaption for SENSE

The concept of supergraphs is an extension of an ordinary graph formalism, not aimed at representing natural language in particular. It was necessary to restrict and extend it somewhat:

There are two restrictions of the general concept of a supergraph in SENSE: First, we fixed the set of roles (see Section 3). Second, we require that every role occurs at most once in one edge. This is due to the common assumption that a thematic role may occur at most once in a phrase to make sense. Multiple tentacles may end at the same connectible object though. An object can thus occur in multiple relations.

[6] The goal is to obtain a *concise* set of roles

[7] a server, a port, a request, a client, and an action

There are two extensions of the general concept of a supergraph in SENSE: Firstly, we added context sensitive attributes. Secondly, a tentacle is allowed to connect to AND-, OR-, and NOT-sets of connectible objects. The need for context sensitive attributes has been discussed in Section 2.1. At the same place, the need for sets of entities taking a role together has been demonstrated. So the introduction of sets of connectible objects as possible endpoints for tentacles is straightforward.

5 Work in Progress

The usage of a real, independently created texts as in Section 2 reveals some issues that still need to be solved:

More Context. Reasonable processing of modal verbs, quantifiers, and number needs context information. We currently work on the theoretical integration into SENSE as well as a good syntactical integration into SAL_E.

Knowledge. Incorporating world-knowledge would enable reasoning upon a SENSE-graph. Seeing that **`replies`**`|ACT` (line 8) and **`text_content`**`|HAB` (line 9) occur in one relation it would be nice to automatically conclude **`text_content`** = **`response`**. This way, we intend to reach a kind of 'normal form'. This is desirable to further reduce the dependency on style.

Coreference resolution. Our implementation lacks coreference resolution. Whether this resolution is done best in (or before) SAL_E or in SENSE remains to be seen.

Tools. An automatic preprocessing step identifying sentences and clauses boundaries as well as commenting out functional words would be desirable and feasible.

Suitability for other languages. An inherent problem of SAL_E is that relations whose participants are not lexically located next to each other in the text cannot be expressed as easily as nested declarations. They need extra statements.

Transformations. A rule set for the transformation from SENSE to UML is work in progress.

6 Conclusion and Outlook

This paper presented three major contributions: Firstly, we presented a new formalism for the representation of the content of natural language (SENSE) along with a custom-made annotation language (SAL_E), and we showed the applicability to a real world specification document. Secondly, we presented a new set of thematic roles; designed for SENSE – but possibly useful in other applications as well. Thematic roles and graph contiguity, that is how semantics are encoded in SENSE. Thirdly, we presented supergraphs, a fundamental formalism we developed for SENSE. Based on well-known graph rewriting techniques, supergraphs enable a simplified declaration of transformations upon complex circular and recursive data structures.

SENSE has been designed to serve as an intermediate representation for the process of transferring natural language statements into software artifacts. It provides several extension points for future work: (a) development of a natural language parser emitting SENSE, (b) development of a natural language generator accepting SENSE, (c) development of dictionary-based and world-knowledge-based reasoning, and (d) development of transformation rules from and to UML or Story Diagrams [9], to mention but a few.

References

[1] M. Biezunski, M. Bryan, and S. R. Newcomb. Topic Maps. ISO/IEC Standard 13250:1999, ISO, Dec. 1999.

[2] J. F. M. Burg and R. P. van de Riet. COLOR-X Event Model: Integrated Specification of the Dynamics of Individual Objects. pages 146–157, 1995.

[3] L. Daigle. WHOIS Protocol Specification. RFC 3912, The Internet Engineering Task Force (IETF), Sept. 2004.

[4] O. Denninger. Erweiterung des Kantenkonzepts deklarativer Graphersetzungssysteme von Einfachkanten über Hyperkanten zu Superkanten. Master's thesis, Universität Karlsruhe, 2007.

[5] O. Denninger, T. Gelhausen, and R. R. Geiß. Supergraphs – Simplifying Model Transformation for Advanced UML Structures. Preliminary version, submitted to "Applications of Graph Transformation with Industrial releVancE" (AGTIVE), July 2007. see `http://www.ipd.uka.de/Tichy/publications.php?id=135&lang=en`.

[6] F. Dignum and R. P. van de Riet. Knowledge base modelling based on linguistics and founded in logic. *Data Knowledge Engineering*, 7:1–34, 1991.

[7] S. C. Dik. *The Theory of Functional Grammar. Part I: The Structure of the Clause.* Flores Publications, Dordrecht, 1989.

[8] C. J. Fillmore. Towards a modern theory of case. In D. A. Reibel and S. A. Schane, editors, *Modern Studies in English*, pages 361–375. Prentice Hall, 1969.

[9] T. Fischer, J. Niere, L. Torunski, and A. Zündorf. Story Diagrams: A New Graph Rewrite Language Based on the Unified Modeling Language and Java. In *TAGT'98: Selected papers from the 6th International Workshop on Theory and Application of Graph Transformations*, pages 296–309, London, UK, 2000. Springer-Verlag.

[10] R. Geiß, G. V. Batz, D. Grund, S. Hack, and A. M. Szalkowski. GrGen: A Fast SPO-Based Graph Rewriting Tool. In A. Corradini, H. Ehrig, U. Montanari, L. Ribeiro, and G. Rozenberg, editors, *Graph Transformations - ICGT 2006*, Lecture Notes in Computer Science, pages 383 – 397. Springer, 2006. Natal, Brasil.

[11] R. Goetz and C. Rupp. Psychotherapy for System Requirements. *International Conference on Cognitive Informatics*, 00:75, 2003.

[12] H. M. Harmain and R. J. Gaizauskas. CM-builder: An automated NL-based CASE tool. In *Automated Software Engineering*, pages 45–54, 2000.

[13] H. Liu and H. Lieberman. Toward a programmatic semantics of natural language. In *Visual Languages and Human Centric Computing, 2004 IEEE Symposium on*, pages 281–282, Washington, DC, USA, 26-29 Sept. 2004. IEEE Computer Society.

[14] S. Müller. *Deutsche Syntax deklarativ. Head-Driven Phrase Structure Grammar für das Deutsche*. Number 394 in Linguistische Arbeiten. Max Niemeyer Verlag, Tübingen, 1999.

[15] S. Müller. *Head-Driven Phrase Structure Grammar – Eine Einführung*. Number 17 in Stauffenburg Einführungen. Stauffenburg Verlag, Tübingen, Germany, 2007.

[16] A. M. Moreno. Results of the application of a linguistic approach to object-oriented analysis. *International Journal of Software Engineering and Knowledge Engineering*, 8(4):449–459, December 1998.

[17] C. Pollard and I. A. Sag. *Head-Driven Phrase Structure Grammar*. University of Chicago Press and CSLI Publications, Chicago, Illinois, 1994.

[18] G. Rauh. *Tiefenkasus, thematische Relationen und Thetarollen*. Gunter Narr Verlag, Tübingen, Germany, 1988.

[19] C. Rolland and C. Proix. A Natural Language Approach for Requirements Engineering. In P. Loucopoulos, editor, *Proceedings of the Fourth International Conference CAiSE'92 on Advanced Information Systems Engineering*, volume 593, pages 257–277, Manchester, United Kingdom, 1992. Springer-Verlag.

[20] B. Szent-Ivanyi. *Der ungarische Sprachbau*, pages 45–50. Buske, 1995.

A Thematic Roles

actus (ACT) an action executed by a thing or person, see also: agens, patiens

agens (AG) the acting thing or person, see also: actus, patiens

beneficiens (BEN) (signed) the thing or person benefiting (+) or drawing back (-) from an action, see also: fautor, favor

causa (CAU) (signed) the cause of an action (+) or a fact regardless of which an action happens (-), see also: actus, intentio

comes (COM) a companion, see also: dux

comparand (COMP) an element being compared with, see also: compariens

compariens (COMP-II) the element being compared, see also: comparand

contrarius (CONT) an opponent, see also: contrariens

contrariens (CONT-II) the element that has an opponent, see also: contrarius

creator (CREA) (signed) the thing or person that creates (+) or destroys (-) something, see also: opus

destinatio (DEST) a target, a goal, a destination (insufficient for classification, add further roles to be more specific), see also: origo, positio

dimensio (DIM) an extend, see also: locus, tempus

donor (DON) a thing or person that emits something, see also: habitum, possessor, recipient

dux (DUX) a thing or person that is accompanied, see also: comes

experior (EXP) a thing or person experiencing something (i.e. through sensory perception), see also: notio, stimulus

fautor (FAU) (signed) a thing or person acting in favor (+) or adverse (-) for somebody or something else, see also: beneficiens, favor

favor (FAV) (signed) a favor (+) or disadvantage (-), see also: beneficiens, fautor

fictum (FIC) a role somebody or something plays, see also: fingens

fingens (FIN) the thing or person playing a role, see also: fictum

frequens (FREQ) the frequency or points in time of an action, a 'path in time', see also: tempus, limes

habitum (HAB) a property, something 'had', held, or owned, see also: donor, possessor, recipient

instrumentum (INSTR) (signed) a tool, an aid being used (+) or especially *not* being used (-) for an action, see also: actus, modus

intentio (INT) the intention or purpose of an action, see also: actus, causa

limes (LIM) a path (in space), see also: locus, frequens

locus (LOC) a location, a place, a 'distance' in conjunction with DIM, (insufficient for classification, add further roles to be more specific), see also: dimensio, destinatio, limes, origo, positio

modus (MOD) a mode in which an action is executed, see also: actus, instrumentum

notio (NOT) a notion, an image, an idea, a sound that is transmitted (i.e. through sensory perception), see also: experior, stimulus

omnium (OMN) a whole that has (or consists of) parts, see also: pars

opus (OPUS) (signed) a thing that is created (+) or destroyed (-) by an action, see also: creator

origo (ORIG) a source, a beginning, a starting point (insufficient for classification, add further roles to be more specific), see also: destinatio, positio

pars (PARS) a part of a whole, see also: omnium

patiens (PAT) a thing or person being affected by an action, see also: actus, agens

positio (POS) the current position (in the sense of a reference point) of an element (not only local!) (insufficient for classification, add further roles to be more specific), see also: destinatio, origo

possessor (POSS) the current possessor, the person currently 'having' something, see also: donor, habitum, recipient

qualitas (QUAL) the quality, the consistence, the nature, the flavor of an object, see also: qualifitiens

qualifitiens (QUAL-II) a qualified object, see also: qualitas

recipient (RECP) the recipient of an object, see also: donor, habitum, possessor

stimulus (STIM) a stimulating (i.e. through sensory perception) thing or person, see also: experior, notio

substitutus (SUBS) a thing or person that is substituting another thing or person, see also: substituens

substituens (SUBS-II) a thing or person that is substituted, see also: substitutus

sumptio (SUM) the precondition for an action, see also: actus

tempus (TEMP) a time, a date, a 'period' in conjunction with DIM (insufficient for classification, add further roles to be more specific), see also: dimensio, destinatio, frequens, origo, positio

thema (THE) a theme, see also: thematiens

thematiens (THE-II) an element that has a theme, see also: thema

(End Of List.)