Recent Results of the NLRE (Natural Language based Requirements Engineering) Project

Günther Fliedl, Christian Kop, Heinrich C. Mayr, Alexander Salbrechter, Georg Weber, Christian Winkler

University of Klagenfurt, Austria

Abstract

NLRE is a project that handles natural language requirements specifications to derive conceptual schemata. For this purpose an interlingua was introduced. Parts of the natural language specification are firstly extracted and collected in the interlingua schema. Afterwards all concepts of the interlingua schema are mapped to a conceptual schema. The current paper gives a rough overview of the project status and the tool support.

1. Introduction

Detecting and deriving those notions in a natural language discours which are crucial for modeling Conceptual Schemata is the computer scientist's most challenging task.

The NLRE Project was launched to minimize the gap between end-user requirements formulated mainly in spoken natural language and the more formal requirements specification (Conceptual Model) of the designer.

Direct mapping from natural language to conceptual modeling is highly complicated. Furthermore, design decisions as to *what is an attribute* vs. *what is a class* or *what is a state* and *what is an event* are dependent primarily on technical aspects rather than on given requirements. This is why we propose an intermediate step, called **KCPM** (**K**lagenfurt ***C**onceptual **P**redesign **M**odel*), which offers a minimal set of modeling notions particularly suitable for classifying requirements.

As an integrated part of the overall project working task, aimed at supporting a step by step automatic mapping from natural language to a conceptual model with the help of KCPM, the current project focusses on dynamic aspects of modeling, in particular on:

- the complexity of natural language as opposed to formal languages
- modeling options regarding dynamic aspects in conceptual modeling

As for dynamic modeling, the complexity of natural language (redundancy, ellipses, anaphoric/cataphoric elements etc.) is extremely challenging. Customers sometimes express *pre-* and *post conditions* along with *actions* packed into one single sentence. In other cases, one single requirement is referred to by means of several sentences. Thus, a tool for natural language analysis has to cope not only with the complexity of one sentence: it is expected to detect relationships between different sentences belonging together.

UML offers 5 different diagram types for dynamic modeling. Even if all dynamic notions are processeded to one simple intermediate KCPM schema, the question still remaining is to which conceptual dynamic model it should be eventually mapped. This is why the second working task within the NIBA project is concerned with tools and heuristics supporting the mapping to the respective model.

2. KCPM

The interlingua consists of a small set of modeling notions namely thing type, connection type, cooperation type, operation type, pre- and post condition and constraint. Thing types represent important domain concepts. A **thing type** is a generalization of a class and value type. That means we do not distinguish between classes and value types but leave the distinction to the mapping process. Thus "*customer*" as well as "*customer name*" is modeled as a thing type. Typical things (instances of thing types) are natural or juristic persons, material or immaterial objects, abstract notions. In textual requirements specifications they are usually referred to by noun phrases. Relationships between thing types are modeled by **connection types**. Dynamic aspects are modeled with operation types and cooperation types.
Operation types are used to model functional services, that can be called via messages (service calls). As such they may be seen as a generalization of the notions use case, activity, action, method, service etc. Each operation type is characterized by references to so-called **thing types**, which model the *actors* (executors and callers of the resp. operation type) and *service parameters*. UoD dynamics emerge from *actors* (thing types) performing operations under certain circumstances (*pre-conditions*) and thus creating new circumstances (*post-conditions*). This is captured by the KCPM concept of **co-operation type, a** term we adopted from object orientated business process modeling [KKM95]. A particular cooperation in that sense is an elementary step of a business process to which one or more actors contribute by (concurrently) executing operations (services).
Requirement specifications which cannot modeled by one of the modeling notions are modeled as constraints.

3. Natural Language Analysis

To cope with the complexity of natural language we have developed a POS tagger with shallow parsing features. The tagger also provides a smart verb classification which is the basis to distinguish between the main concepts of KCPM dynamic modeling (namely: cooperation type, operation type and condition). Furthermore it allows word stemming, lemmatizing, and morphosyntactic analysis. The tagged output is represented in XML. Figure 1 shows a typical output of the tagger.

```
Tagger
<text>
    <sentences>
    <sentence>
    <n3>
        <spz0 position="1">Der</spz0>
        <n0 position="2">Auftrag</n0>
    </n3>
    <v0 referTo="eintreffen" role="spz0" type="eV" position="3">trifft</v0>
    <pt0 partOf="3" position="4">ein</pt0>
    </sentence>
    <sentence>
    <n3>
        <spz0 position="1">Die</spz0>
        <n0 position="2">Auftragsabteilung</n0>
    </n3>
    <v0 referTo="prüfen" position="3" type="tvag2">prüft</v0>
    <n3>
        <q0 type="all" position="4">jeden</q0>
        <n0 position="5">Artikel</n0>
    </n3>
    <n3>
        <spz0 position="6">des</spz0>
        <n0 referTo="Auftrag" position="7">Auftrags</n0>
    </n3>
    </sentence>
```

Figure 1: Tagger Output

The output can be used in the following way to collect KCPM entries. We have to look at the main verb and it's verb class. Verb classes like *eV* (ergative verbs) are used to model a condition. Verbs of the class *tVag2* (agentive verbs) are used to model operation types. A detailed description of how operation types, conditions and cooperation types can be determined from verb classes is described in [Fl03].

4. Tool Support for KCPM modeling and mapping to UML

Concerning behavioral modeling, there is a tool for storing and managing KCPM schemata (insertion, update and deletion of operation types, cooperation types, conditions etc.) in a graphical representation (see Fig. 2) For analysis purposes theses tools allow to turn on or off the visibility of particular schema parts in order to reduce complexity and thus increasing transparency for validation. Furthermore, a tree view supports the presentation of several levels of abstraction.

A mapping feature is an integrated component of this tool. It transforms the KCPM schema to a XMI Schema for UML activity diagrams according to mapping heuristics given in [KM02]. The XMI Schema can then be visualized using the UML Tool "Poseidon for UML"[1].

[1] Poseidon for UML is a Trademark of Gentleware.

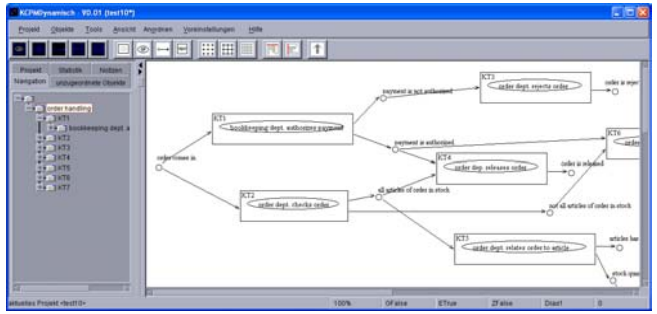

Figure 2: Screenshot of the KCPM tool for graphical modeling

5. Conclusion

Beside the tagging tool (fig. 1) and the tool for graphical modeling of KCPM notions (fig. 2), there are still other tools which are under development. Another "mapper" for transforming KCPM schemata to state diagrams will be developed in a next step. We will also provide the user with a web application allowing him to navigate through the requirements specifications, requirements documents and the KCPM schemata exploiting the advantages of a hypertext.

References

[Fl03] Fliedl, G.; Kop, Ch.; Mayr, H.C.: From Scenarios to KCPM Dynamic Schemas: Aspects of Automatic Mapping. In: Düsterhöft, A. Thalheim, B. (eds.): Natural Language Processing and Information Systems – NLDB'2003, GI Lecture Notes in Informatics, Vol. 29, pp 91 – 105.

[KKM95] R. Kaschek, C. Kohl, H.C. Mayr: Cooperations - An Abstraction Concept Suitable for Business Process Reengineering, in Györkös, J. et.al. (eds.) *Re-Technologies for Information Systems, Proc. ReTIS'95*, R. Oldenburg Verlag, Wien, 1995.

[KM02] Kop, Ch.; Mayr, H.C.: Mapping functional requirements: form natural language to conceptual schemata. In: Proceeding of the 6th IASTED International Conference Software Engineering and Applications, Cambridge USA, November 4-6, 2002, pp.82 – 87.