Knowledge Representation Concepts Supporting Business Process Analysis

Hans W. Nissen
Informatik V
RWTH Aachen
Ahornstraße 55
52056 Aachen
Germany

Georg V. Zemanek
USU Softwarehaus
Unternehmensberatung GmbH
Spitalhof
71693Möglingen
Germany

Abstract

Modeling and analysing business processes is a frequent job in professional consulting projects, but adequate commercial tools or even formal methods supporting this task hardly exist. This paper reports about the successful application of the knowledge representation system ConceptBase to this task. Based on generic modeling facilities on the one hand and powerful query mechanism on the other ConceptBase is able not only to represent and analyse the final complex model but also to support and record intermediate states together with transitions between them. Our experience has shown that a logic based knowledge representation language is *not* inconvenient for practical modeling tasks but even urgently needed to handle large and complex models in an adequate way.

1 Introduction

During the first KRDB workshop in 1994 a first contact between the information systems group of Informatik V at the RWTH Aachen and the consulting firm USU was established. Now, at the second workshop in this series, we can report on a successful cooperation project. Within this project the deductive object base manager ConceptBase [4] developed in the group of Matthias Jarke was used to model and reason about business processes.

Early phases of consulting projects concerning the introduction of sophisticated information systems include the analysis of existing business processes together with the derivation of requirements as the fundamental goal. For this task only rudimentary tool support or fragments of formal methods exist. Almost all existing tools contain a fixed view of the world, an extension of the supported concepts is not possible. As a consequence, the tools prescribe the analysis procedure and not vice versa.

The aim of this project was to develop concepts and a prototype for a comprehensive support of the current USU-PFR method used to capture information about the domain of interest. The result should be easy to use and to understand, such that also customers are able to use it, and on the other hand be powerful enough for a convincing analysis.

The next section describes the USU method to business process modeling together with some of the arising problems in tool support. The knowledge representation formalism Telos is introduced in section 3. Section 4 presents the application of Telos and ConceptBase to this task while the last section summarizes our experiences.

2 The USU Method to Business Process Modeling

Requirements are captured from multiple, sometimes unforeseen perspectives: content and structure analysis of existing documents, interviews with individuals describing their current situation and wishes, informal textual or visual conceptual models developed in planned or unplanned meetings of stakeholder groups, reverse analysis of existing systems, or goal analysis from a business or individual perspective. The study of each of these sources may lead to new questions, to be answered from new sources until a somewhat coherent picture of requirements emerges.

A typical USU consulting project follows the so called PFR method (Analysis of Presence and Future Requirements) [1] The aim of this method is to generate a shared and agreed understanding of the current business processes, the problems, and a first vision of the target system. The main part consists of two phases.

In a cooperative fashion a set of involved persons generate in a first phase a rough overview of the existing processes (mostly in terms of information exchange among organisational units). Based on the result of the first phase people working within the identified units describe in a second step in detail the sequence of their activities together with relationships to other persons in the organisation. This step has the goal of testing the initial vision against the existing and expected organizational context, and to elaborate it, both in terms of deepened *understanding* and in terms of more formal *representations* (e.g. in the form of activity sequences, data flow models, entity relationship diagrams or object models). This step also includes an analysis of exchanged media in order to capture hints for further process optimization.

From a representational viewpoint, the PFR methodology comprises a set of source perspectives as captured in the first two steps, and a set of result perspectives which represent the delivered requirements (with the intent of presenting them to users or to use them in subsequent design tasks). The details of these perspectives may change with the individual customers and projects

The source perspectives are:

- *The information exchange between organisational units.* This perspective aims to produce a visual overview of the current or future situation including the identification of weak spots of the process under investigation. It is represented in an informal collage style employing a fixed set of graphical symbols and pictograms. Although its semantics is a bit vague, it provides a valuable overview of the current situation and its limitations. This representation is not only used to produce a picture of the current situation, but also to visualize a first version of the target conception.
- *The individual activity sequence of stakeholders.* This perspective is captured for each stakeholder by individual interviews and describes in form of a detailed flow chart the sequence of activities, the required and produced information, and inter-relationships with other stakeholders. In the same way information from already existing workflow documents, as, e.g., the quality management handbook, is represented.
- *The structure of exchanged media.* This perspective identifies the pieces of information that reside on forms, documents and other kinds of media that are exchanged between stakeholders resp. organisational units. This breakdown of a medium into the pieces of information it carries is necessary for a detailed analysis of the activities performed by stakeholders.

Cross-perspective analysis applies these source perspectives and mainly consists of a comparison of the perspectives to detect discrepancies, modeling errors, gaps, and properties of the business process. The results of this comparison activity guide further interviews to clarify the inconsistencies and to complete the models. During these changes in individual perspectives, the corresponding derived knowledge about the conflicts has to be maintained, as old conflicts may disappear and new problems may surface. USU's experience in applying this method to a large number of projects has shown that an analysis by hand is a time-consuming and error-prone task. A supporting tool should therefore

- represent the information from all perspectives in a natural way (which may be different from customer to customer or from project to project) such that they can be easily communicated to stakeholders,
- enable the comparison of diagrams represented according to different (semi-)formal notations to detect discreparencies, modeling errors, gaps, etc., and maintain the detected relationships over time,
- be able to automatically generate function-oriented and data-oriented perspectives on the provided information to be used as a starting point in subsequent analysis and design steps.

Use of existing CASE tools proved unsatisfactory for these tasks, as they were too rigid in their hard-coded consistency analyses which were developed for other purposes.

Frequently, the set of perspectives has to be customized by aspects which are specific to a particular project but do not occur sufficiently often to include them in the standard methodology. Or a customer organization uses an existing methodology in subtly different ways than others.

What is needed, is a *simple* formalism which is *extensible* to the needs of specific methodologies or even application projects but still provides the formal background for *integrating* all the perspectives used. This combination of simplicity of basic formalism, extensibility, and formal integratability proved crucial to the success of ConceptBase.

3 The Knowledge Representation Language Telos

In this cooperation we used the deductive object manager ConceptBase. ConceptBase is a prototype system that is based on the knowledge representation language Telos [7]. Telos is especially designed to offer modelers the flexibility to define and use their particular understanding of the world, and to relate this understanding to that by others. Telos offers a simple generic data model that is extensible to specific application needs and provides mechanisms for perspective integration.

The kernel model of Telos consists of just two concepts: nodes and links. To allow any kind of formalization, we need a third concept, that of an assertion. Finally, to talk about different notations, we need at least one abstraction mechanism – classification – which enables us to talk about classes and their instances. The kernel of the Telos language is just that. All other language facilities can be bootstrapped from this kernel of nodes and arcs, assertions, and classification.

Tailoring Telos to specific application data models is done by first embedding the structural regulations of the language (i.e. its syntax) into Telos, second giving the new modeling constructs a formal semantics by defining appropriate rules and constraints, and third introducing the diagrammatic presentation of the language by assigning graphical type descriptions to the modeling constructs. In this section we concentrate on the structural and semantic extensibility.

Structural Extension. The infinite levels of classification available in Telos enable the creation of (meta) models. Such a meta model extends the admissible set of modeling constructs to the models considered on an abstraction level lower than the meta model. This technique can be used to integrate the structural part of other modeling languages and to make the modeling concepts of that language available. The meta model acts then as a conceptual model of the structural part (syntax) of the modeling technique.

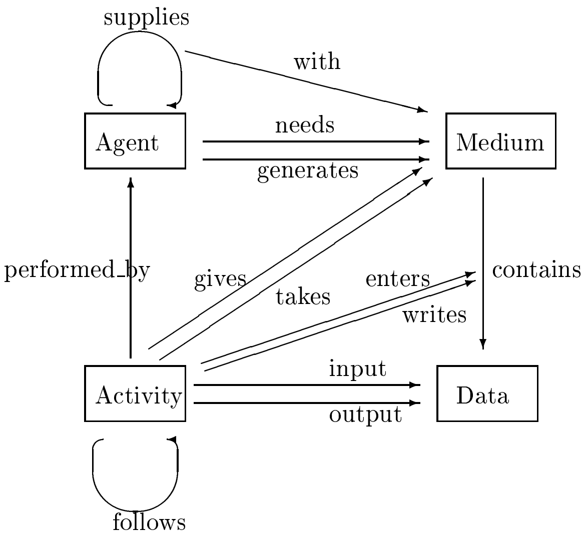

Figure 1: The Telos meta model for the PFR analysis method

Semantic Extension. Most implemented approaches to meta modeling cover the structural part well [6, 9] but offer semantic extension only within a predefined set of constraint types (e.g. cardinality constraints). Telos assertion objects make it possible to specify the semantics of language extensions as part of the corresponding meta model. The formal behavior, defined in the form of integrity constraints and deductive rules, can be directly attached to the corresponding class definition. In ConceptBase, semantic extensibility is assisted by so-called *meta formulas* [5]. We allow formulas to make statements across several instantiation level. Thus, they are able to specify the behavior of objects which reside two or more instantiation levels below the objects of the meta model.

4 Extension of Telos Towards the PFR Method

The meta model shown in figure 1 was derived from a cumulative analysis of the perspectives typically discussed in USU's RE projects. By emphasizing the relevant objects in the meta model, we show in the following how the different perspectives described in section 2 are captured in this meta model. Based on this description, we also present a number of query classes for analyzing conflicts among these perspectives. For each of these query classes, a set of possible explanations and related courses of action have been developed in order to help USU analysts in conflict resolution.

The meta model covers all PFR source perspectives. Figure 2 presents the individual perspectives and also visualizes the overlaps of them. Part (a) highlights the part of the meta model used to represent the *information exchange between organisational units*, as captured in the "collage" of the initial workshop We model an organisational unit as an abstract `Agent` who `supplies` another agent `with` a `Medium`. The earliest version of the meta model had

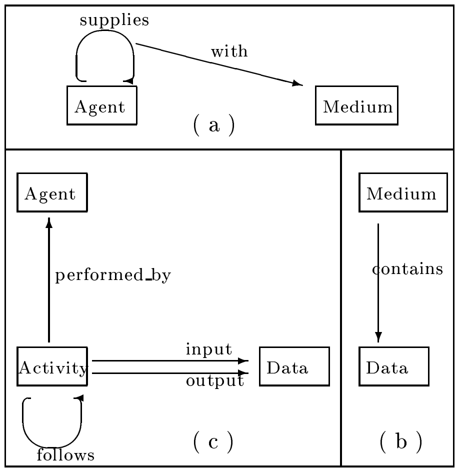

Figure 2: The PFR perspectives within the meta model

this simply as a data flow but, observing the participants of the first pilot project, we recognized that agents do not really exchange information, but the medium that act as the data or information carrier. A medium can be something persistent, like a piece of paper, a form or a disk, or a transient thing like the voice that carries words.

The model in part (b) therefore represents *the structure of exchanged media* by explicitly distinguishing the `Medium` and the `Data` it `contains`. This distinction is essential to talk about phenomena such as empty and completed forms, reading from and writing to a medium, replicating the same piece of data on multiple media manually or automatically, and agents that get a medium but perform no activity that needs or produces any data located on that media. For example, one project revealed that the same data was captured and re-captured several times in a workflow, with very good and expensive quality controls, except in the last step! Here, the meta model helped to explain why there was bad quality despite high quality control costs.

The conceptual model of the *individual activity sequence of stakeholders* is shown in part (c). An `Activity` is `performed_by` an `Agent`. A partial order on activities (workflow) is expressed by the `follows` relation. An activity is an atomic action that takes some information or `Data` as `input` and generates new `Data` as `output`. Our semantics of the `output` relation is very rigorous: The activity must create this data for the first time, i.e. no other activity can also create this data. Every piece of data is created exactly once. The motivation behind this is that the data once created gets never lost.

As indicated in the description above, the perspectives are strongly interrelated by overlaps and redundant information. The USU application projects identified more than 70 constraints describing the consistency of the captured information. This includes consistency of knowledge *within* an individual perspective as well as the consistency *between* different perspectives.

In Telos, we can formally include consistency checks by attaching integrity constraints to the appropriate objects of the meta model. As a consequence, the system will reject every update that violates one of these constraints. This rigid consistency enforcement strategy is not well suited for RE workers: The distributed knowledge acquisition process and the overlapping perspectives lead to numerous conflicts, which then always have be solved before inserting new information into the knowledge base. This delay hampers the analyst and the whole acquisition and analysis process. It also forces perspective reconciliation to take place outside the system, and without traceability.

In contrast, Telos query classes offer a more flexible way to analysis and enforcement. Queries are represented as classes (i.e they are first class objects in a Telos model) and the answers become the virtual instances of that class. Applied to our problem, the answers to the query are interpreted as consistency violations.

USU did not only formulate queries to detect errors within and between perspectives, but also to analyse the properties of the finally reconciled business process model. This includes questions like "What is the trace of form X305 ?", to detect the reason for the long handling time of the form X305. All together USU produced over 80 query classes. To further support the analyst, we developed guidelines for applying the queries. For each query class, they include a set of possible answer interpretations in the light of business processes as the application domain together with appropriate repair suggestions. In addition, we established a sequence of the queries that proved to be reasonable within our experiment projects.

4.1 Some Analysis Examples

In this subsection we present some concrete examples of query classes and answer interpretations. We first give a brief to the syntax of query classes: A query class is formulated in the Telos frame syntax, and has the following form:

```
QueryClass <name> isA <superclasses> with
  attribute
    <answer attributes>
  constraint
    <condition>
end
```

We can distinguish four important parts:

1. The name of the query class is given by `<name>`.
2. The `<superclasses>` part specifies the superclasses of the query class. The set of possible answer objects of the query are then restricted to the common instances of the superclasses. If this part is omitted, `Object` becomes the superclass which enables all objects of the knowledge base to join the answer set.
3. The `<answer attributes>` part defines the attributes of the answers to the query. The attributes either already exist in the knowledge base, or are deduced during query evaluation.
4. The `<condition>` part contains the query condition which can be an arbitrary closed formular. The symbol `this` used within the condition refers to potential answer objects, i.e., the instances of the superclasses.

Analysis of a single perspective
Consider the activity sequence perspective. The query class below realizes the constraint that data can only be used by an activity (indicated by the `input` relation) after it was created (via the `output` relation). The query deduces data that are used as `input` before they are produced.

```
QueryClass Data_UsedBeforeProduced isA Data
   with  attribute
    early_user : Activity
  constraint
    c : $ (early_user input this) and
           (producer output this) and
        (producer trans_follows this) $
end
```

The query class uses the `trans_follows` relation, which denotes the transitive closure of the `follows` relation and is deduced by a Telos recursive rule. The answer can be interpreted as

1. an error, if the interviewed agent indicated a wrong sequence.
2. an error, where the interviewer misinterpreted a statement and modeled an `input` relation to `Data` instead of a `gives` relation to `Medium`.
3. nothing else, since this model represents a real existing and running process where data cannot be used before it is produced.

Analysis of interrelationship among multiple perspectives
In the consulting projects we often detected contradictions between the high-level information exchange perspective acquired mainly from managers and the detailed activity sequence perspective captured from the real working agents. An often violated interrelationship states that the medium flow among agents must correspond to the data demand of agent's activities, i.e. the supplied media must contain some data that is required by an activity and, conversely, all required data must be contained on some delivered medium. The following query class implements the first part and deduces all media that is supplied to an agent who performs no activity that needs any data carried by that medium.

```
QueryClass NotUsedMedium isA Medium with
  attribute
     not_user : Agent
  constraint
     c : $ (supply in Agent!supplies)
             and (supply to not_user)
             and (supply with this)
             and not exists (
       (action performed_by not_user)
       and (this contains info)
       and ((action input info)
       or (action output info))    ) $
end
```

The answers are the media together with the agent who gets the media but does not use it. They can be interpreted as follows:

1. There exists a mismatch between the captured perspectives: the management and the concrete employees view the process in different ways. Further clarification interviews are necessary to reconcile the contradicting views.
2. The model is correct and the agent actually gets and sends the medium without any interest on the data. In this case the business process can be further improved by optimizing the media flow.
3. The model is correct and the business process is ok, but the activity that works on the medium does not require any information from the medium.

In practice, we often observed the problem described in 2. As an example of interpretation 4, a secretary collected the monthly reports of the employees of a department to give them as one piece to the manager of that department.

5 Conclusions

The applicability of ConceptBase and Telos to the task of business process modeling and analysis has been successfully proved within this project. We developed a specialized knowledge representation tool containing an adequate meta model, over 80 analysis queries together with predefined answer interpretations and guidelines how to use this system within further projects. The world model (i.e., the meta model) can easily be tailored to specific application needs, and the modeler can individually decide when to use which predefined queries for checking and analysing purposes. It exactly fits the methods used in the company, without precluding future evolution of these methods or customization to individual projects. The information exchange, document structure, and activity sequence can be represented within one meta model; a number of useful observations about the practicality of modeling features (e.g. distinguishing media and data, granularity of modeling required) were made.

An *extensible formal language* like Telos is able to provide a valuable complementary support for informal, teamwork-oriented methods. Since we can tailor the language to the specific application needs, we must also be able to formulate specific analysis queries - a fixed set of predefined queries as provided by most CASE environments will not fulfill this. This requires a powerful declarative assertion and query language based on a well-defined formal semantics.

The experiences confirm the usefulness of *requirements freedoms* and explicit tolerance of inconsistencies within and across multiple viewpoints, as postulated by researchers such as Balzer [2], Feather and Fickas [3], Finkelstein and colleagues [8]. This may seem in contrast to the old paradigm of consistently refining an initially consistent specification – the only known way to create provably correct software. However, recall that we are concerned with an early phase of analysis; its end result should still be consistent so that the consistent refinement approach may still be used in conjunction with our approach.

Conflicts during analysis force discussions and increase the understanding of the domain under investigation. Exactly for that reason we developed a meta model that potentially includes a lot of conflicts. A systematic way of developing such meta models in general is a subject of further research.

References

[1] P. Abel, "Description of USU-PFR analysis method", *Technical Report USU*, 1995.

[2] R. Balzer. Tolerating inconsistency. In *Proc. of the 13th ICSE*, 1991.

[3] M.S. Feather and S. Fickas. Coping with requirements freedom. In *Proc. Intl. Workshop Development of Intelligent Information Systems*, 1990.

[4] M. Jarke, R. Gallersdoerfer, M. Jeusfeld, M. Staudt, S. Eherer "ConceptBase - A Deductive Object Manager for Meta Databases" *Journal of Intelligent Information Systems* **4**(2), 1995.

[5] M.A. Jeusfeld. *Update Control in Deductive Object Bases.* PhD thesis, University of Passau (in German), 1992.

[6] K. Lyytinen, P. Kerola, J. Kaipala, S. Kelly, J. Lehto, H. Liu, P. Marttiin, H. Oinas-Kukkonen, J. Pirhonen, M. Rossi, K. Smolander, V.-P. Tahvanainen, and J.-P. Tolvanen. MetaPHOR: Metamodeling, principles, hypertext, objects and repositories. Technical Report TR-7, University of Jyväskylä, December 1994.

[7] J. Mylopoulos, A. Borgida,M. Jarke, and M. Koubarakis, "Telos: a language for representing knowledge about information systems", in *ACM Trans. Information Systems* **8**(4), pp. 325–362, 1990.

[8] B. Nuseibeh, J. Kramer, and A. Finkelstein. Expressing the relationships between multiple views in requirements specification. In *Proc. of ICSE 93*, 1993.

[9] M. Saeki, K. Iguchi, K. Wen-yin, and M. Shinohara. A meta-model for representing software specification & design methods. In N. Prakash, C. Rolland, and B. Pernici, editors, *Proc. of the IFIP WG8.1 Working Conference on Information System Development Process*, Como, Italy, 1-3 September 1993.

[10] G.V. Zemanek, "Project USU–ConceptBase: The results" *Technical Report USU*, 1995.