Knowledge-Based Business Process Modeling and Simulation

Dmitri Soshnikov
Department of Numerical Mathematics and Programming,
Moscow Aviation Institute
(State Technical University)
Moscow, Russia
e-mail: dsh@mailabs.ru

Sergey Dubovik
Department of Numerical Mathematics and Programming,
Moscow Aviation Institute
(State Technical University)
Moscow, Russia
e-mail: dse@mailabs.ru

Abstract

This paper presents an approach for modeling and simulation of certain domains of business processes, which can be classified as knowledge-based business processes. In those areas, knowledge is already partly formalized, and often structured functional models of the problem domain exists; however, it is complex enough to justify knowledge-based implementation of process execution.

The approach is based on extending structured functional decomposition model of business process with traditional knowledge representation techniques, such as AND/OR trees or production rules, that can be used to describe primitive processes on the lowest level of decomposition. In the process of modeling, both the structure of problem domain concepts and their dynamic properties are identified. A graphical notation is proposed, based on SADT/IDEF-0, extended with knowledge representation notations (graphical as well as textual) to define dynamic properties of lowest-level primitive processes. Process model in this graphical notation can be effectively translated into one-level graphical notation of extended conceptual graphs, as well as into production or production-frame knowledge representation that can be used in creating intelligent automation and/or expert systems. It is noted that forward-chaining logical inference in resulting knowledgebase is simulating business process execution, while backward chaining allows performing goal-based requirements analysis.

Proposed approach can be viewed as a complementary multi-level model of knowledge representation that is based on structured functional decomposition of knowledge field instead of abstraction. Thus, together with frame-based modeling, those techniques can form the basis of complete graphical representation of structural and dynamic knowledge of certain problem domains.

Proceedings of the 6th International Workshop on Computer Science and Information Technologies CSIT'2004
Budapest, Hungary, 2004

1. Introduction

There is no doubt that business process modeling and engineering constitute a very important area of modern information technology. Being very complex systems to study, modern business processes require specific techniques to develop process models that can handle complexity using some means of abstraction and/or decomposition, and that can further be used to simulate business process or apply some optimization techniques.

It can be noted that, while some business processes are inherently algorithmic, others have more complex and not strictly deterministic behaviour, where many decisions are taken non-deterministically based on expert **knowledge**. We would refer to the class of

those processes as to **knowledge-bases business processes**. Such processes appear in the areas of law consulting, notary actions, workflow modeling, selection of products and services on e-business portals, etc.

Unlike in other areas such as medicine, knowledge about business processes in many cases is already partly formalized in the form of free-text instructions, documents, guidelines, or even better structured functional diagrams. However, this knowledge is still complex enough to justify knowledge-based approach in describing and modeling those business processes.

While the use of knowledge-based techniques for enterprise business process modeling has been proposed previously by a number of authors (a good overview of related papers can be found in [1]), we present an approach that combines knowledge-based business process description with industry-standard functional decomposition approach for structured process modeling. On the other hand, the proposed approach for decomposition can be used in a wider context of knowledge acquisition in other problem domains, not directly related to enterprise process modeling. In addition, this approach provides out-of-the-box technology for business process simulation by logical inference in corresponding knowledgebase.

2. Knowledge-Based Business Process Modeling

2.1. Viewing Business Process Execution as Logical Inference

Let us consider the class of **monotonic** business processes, in which data can be attached to a data flow as a result of a business process, but cannot be altered afterwards[1]. From the hierarchical SADT/IDEF-0 diagram we can effectively construct a one-level diagram that would contain only atomic business processes B_i, and only the lowest level of decomposition of data flows D_j. This diagram would actually be a graph with vertices B_i and directed edges D_j[2]. We would also not distinguish between input, control and mechanism input arrows, since this distinction is not relevant to the execution semantics.

The state of business process in this case can be defined by a function $\mathcal{S} : \{D_j\} \to \mathbb{T}$, where $\mathbb{T}$ is a domain of values with some order relation $\sqsubseteq$, which can also be naturally extended to the set of states itself $\mathfrak{S}$. We will say that D_j is active at state $\mathcal{S}$ if $\mathcal{S}(D_j) \neq \perp$.

Let us assume that each atomic business process is described by a function $B_i : \mathfrak{S}' \to \mathfrak{S}''$, where $\mathfrak{S}', \mathfrak{S}'' \subseteq \mathfrak{S}$ – sets of local states that involve only mappings from a subset of $\{D_j\}$ related to B_i. We will say that B_i is active at state $\mathcal{S} \in \mathfrak{S}$, if $B_i(\mathcal{S}) \neq \perp \in \mathfrak{S}''$, i.e. $\exists D_j \in \text{output}(B_i) \quad B_i(\mathcal{S})|_{D_j} \neq \perp$. Each function B_i can also be extended to the whole set $\mathfrak{S}$ in a natural way, by assuming $B_i(\mathcal{S})|_{D_j} = \mathcal{S}(D_j), \forall\, D_j \notin \mathfrak{S}'$. We will also assume B_i to be monotonic, which means that execution of atomic business process does not reduce the information contained in the state.

At each state $\mathcal{S}$, there is a set of active atomic business processes that can be executed, resulting in state being altered. During the execution, one of the active atomic processes is chosen in the process of **conflict resolution**, and applied. The whole process is repeated until the final state or fixpoint is reached. Thus, the execution of the whole process can be viewed as a composition $B_{i_1} \circ B_{i_2} \circ \ldots \circ B_{i_n}$, where the choice of B_{i_j} is not deterministic and is defined by some heuristic execution strategy. Due to finite number of atomic processes and monotonic nature of execution fixpoint is guaranteed to occur after the finite number of compositions[3]. Theoretically, from each initial state $\mathcal{S}_0$ there is a whole set of compositions that lead to a family of final states, and thus the set of possible outcomes of a business process from the initial state S_0 would be $\cup_{\ell=1}^{\infty} \mathfrak{B}_\ell$, where $\mathfrak{B}_\ell = B_{\ell_1} \circ \ldots \circ B_{\ell_{n_\ell}}(\mathcal{S}_0)$.

One notices the similarities between described execution model and forward chaining logical inference in production expert systems, which suggests that execution semantics of structured functional model can be captured using production knowledge representation. Indeed, if we consider each atomic process to have a set of discreet outputs and discreet inputs, it can be represented as a set of production rules — in which case business process execution would coincide with forward chaining inference in the collection of production rules, similar to [2]. Moreover, even when inputs and outputs are not originally discreet, they can be reduced to the discreet case by considering a factor set, as described in [2], or to some arithmetic continuous function of inputs.

2.2. Multi-Level Model Description

Business process execution semantics is conveniently described using one-level diagram, but the same ideas can be applied to trace process execution at any level of decomposition. At the top level, the whole business process would be described by one transformation function $B_1^{(0)}$ with a set of input and output values $D_j^{(0)}$, that would be decomposed on the next level using a set of lower-level functions $B_i^{(1)}$ and arcs $D_j^{(1)}$. If, during decomposition, the input / output arc $D_j^{(k)}$ is decomposed into a set of arcs $\{D_{j_l}^{(k+1)}\}_{l=1}^{n}$ with values from domains $\mathbb{T}_l$, then the value of $D_j^{(k)}$ would belong to

[1] In the process of finding the actual execution path the attached data can change during the backtracking, but once the execution path is found – each step of execution includes the data obtained on earlier steps, only providing additional knowledge.

[2] For the resulting diagram to be a graph, we would need to disregard any tunneled data flows, and in case the data flow arrow is split into two – represent it by two edges of the graph.

[3] Provided that only strictly monotonic processes are applied, i.e. $B_{i_1} \sqsubset B_{i_2} \sqsubset \ldots \sqsubset B_{i_n}$

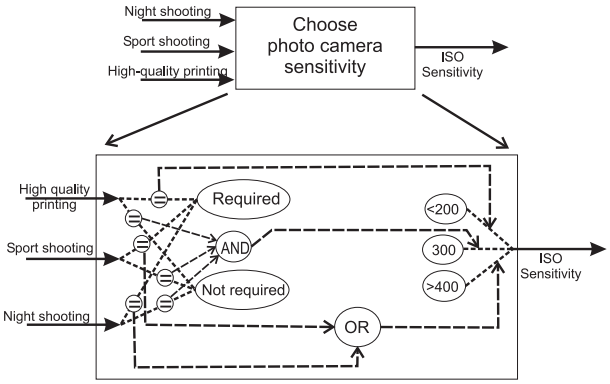

Figure 1. Atomic business process decomposition using AND/OR trees

$\mathbb{T}_1 \times \ldots \times \mathbb{T}_n$. Behaviour of each process function $B_i^{(k)}$ at certain level of decomposition can be described using the fixpoint approach above, and thus the behaviour of the whole system can be constructed step by step from the lowest level of decomposition towards the top-level transformation function.

2.3. Extended AND/OR Tree Notation

The approach presented above defines the execution behaviour of a business process given the functional description of lowest-level primitive processes B_i. To fully describe the system, the behaviour of those primitive processes should also be defined.

One of the ways is to outline a set of primitive processes with pre-defined behaviour, and allow using those at the lowest level of decomposition. However, this approach would force us to use structured functional decomposition below the usual level, which seems to be quite artificial and inconvenient.

Another approach proposed in this paper is to use some form of knowledge representation to describe the behaviour of an atomic process. Such knowledge representation can be graphical (in the form of AND/OR or decision trees) or textural (production rules, decision maps). AND/OR trees are particularly suited for this purpose, since leafs and root of AND/OR tree are boolean values corresponding to some conditions. By extending the notation with conditional and assignment constructions we can seamlessly integrate AND/OR tree notation into business process diagram as shown on Fig.1.

Proposed notation actually borrows from the idea of extending conceptual graphs with AND/OR trees in order to describe dynamic behaviour of the system [3, 4], and includes the following components:

- **Value vertices**, that can be either variable (input/output values of the process or some intermediate variables) or constant (representing some constant values from the problem domain).

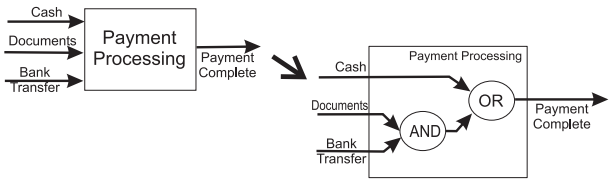

Figure 2. Simplified AND/OR Tree Notation

- **Comparison nodes** that connect value vertices and are labeled with comparison sign (=,<, etc.). Comparison nodes are activated when values assigned to the corresponding value nodes obey the comparison relation.
- **Conditional assignment** arcs connect value vertices and represent assignment of values based on certain condition.
- **AND/OR nodes and connectors** that actually form AND/OR tree starting on the comparison nodes, and ending on the conditional assignment arcs.

The semantics of this notation is quite natural: input values for the AND/OR tree are defined by the comparison nodes. AND/OR tree is computed in the normal way, and when the result is true — corresponding conditional assignment arc is activated, and the value is assigned to the value node.

An alternative way to describe the same behaviour is by using production rules that include input values on the left hand side of the conditional, and output value assignment on the right hand side. AND/OR tree notation is just a graphical representation of production rules, and knowledge engineer may chose whether graphical modeling or textual knowledge representation is more appropriate.

2.4. Simplified AND/OR Tree Notation

In some cases, especially when performing rough process simulation, the actual values assigned to data flows are not important — it is essential to know that certain information is available to take some further actions or decisions. In this case, we can assume that atomic data flows can only be assigned true/false values, and in this case the AND/OR tree notation for atomic process description can be greatly simplified (see Fig.2).

It can be noted that business process model extended with simplified AND/OR tree notation to a certain degree resembles IDEF3 [5] model, with the semantics of Petri nets. However, IDEF3 graphical model is not strongly tied to IDEF0 structured functional decomposition diagram, in a sense that both provide slightly different view of the same process, but are not entirely complementary and cannot be transformed

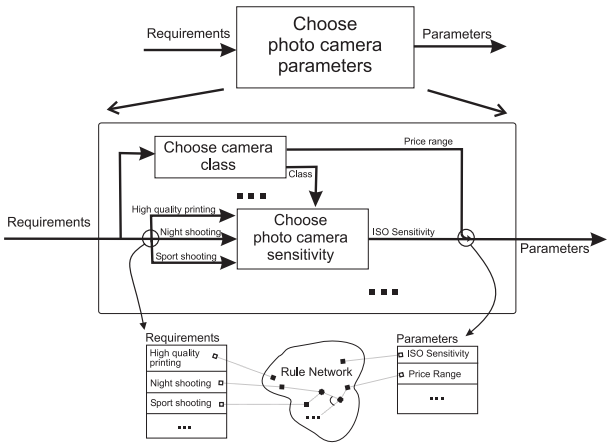

Figure 3. Frame induction from data-flow decomposition and multi-level business process analysis

into each other. In fact, the general recommendation for process modeling is to construct IDEF0 diagram first to gain the initial understanding of the process, and then proceed with developing IDEF3 model.

Proposed approach allows extending the original IDEF0 model with the execution semantics similar to IDEF3 / Petri nets, but without losing the structured functional decomposition approach to dealing with process complexity. Moreover, the result of such modeling can be represented as a knowledgebase, suitable for immediate simulation and/or usage in the decision support systems.

2.5. Extracting Domain Object Structures from Data Flow Decomposition

After business process model extended with AND/OR tree description of atomic processes has been completed, the knowledge of business process execution can be extracted in the form of production rules to form the expert system, which will model business process execution by forward-chaining inference. In this expert system, static knowledge of the problem domain will be described by a set of attribute-value pairs corresponding to atomic data flow arrows.

However, from the multi-level structural decomposition diagram it is also possible to extract the knowledge of domain concepts. Since data flow arrows represent flows and states of information entities in the business process, the decomposition of arrows naturally define inner structure of higher-level domain entities, as shown on Fig.3. On this figure, two frames are shown that represent input data flow and output data flow for the upper level diagram, and slots of those frames correspond to individual atomic data inputs for the lower-level diagram decomposed as AND/OR tree.

Since there may be many layers of decomposition in the business process model, it is not very clear how those can be represented as plain domain concepts with a

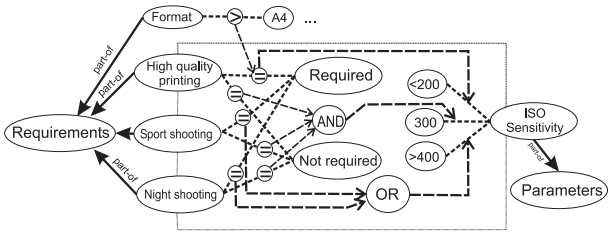

Figure 4. Transformation of business process diagram into extended conceptual graph

number of attributes. The most obvious way to handle this problem would be to manually define the level of decomposition at which to perform attribute grouping, or set some criteria for situations when attributes should be grouped together to form a concept.

The example shows that different knowledge about the problem domain can be extracted from the extended structured functional diagram:

- Structural knowledge about problem domain concepts in the form of frames
- Procedural dynamic knowledge in the form of production rules

2.6. Extended Conceptual Graph Notation

While extended structured functional model can be translated to production-frame knowledge representation and used in the development of software systems, it can also be represented in the uniform notation of conceptual graphs [6, 7] extended with AND/OR trees to express the explicit dynamic properties or domain concepts. In fact, according to the proposed methodology individual atomic business processes are already modeled using the similar notation, and constructing the extended conceptual graph model for the whole process largely consists of merging different atomic process descriptions together (see Fig.4). In addition, this notation allows expressing multi-layered concept composition in the form of part-of relation, which is essentially more expressive than the frame definition.

Extended conceptual graph notation has been described in [3,4]. The advantage of this notation is its uniformity, that allows natural translation of the model into the tuple stack abstract machine with clear execution semantics [4], that can form the basis of implementation of reasoning engine on multiple platforms. The relationship between extended functional model, extended conceptual graph model and tuple abstract machine is shown on Fig.6.

3. Knowledge-Based Business Process Simulation

Using the modeling technique described above, knowledge-based business process model can be

constructed as a structured functional decomposition diagram extended with AND/OR tree models of low-level primitive processes. This model can be then effectively used to simulate business process execution by the process of logical inference in some knowledge-based system driven by the knowledge-base derived automatically from the graphical process model. Process simulation can be part of the investigation of process behaviour directed to some model optimizations, or it can be used in real-life decision support systems or expert systems.

3.1. Translation into Knowledge Representation Models

The AND/OR tree graphical model can be effectively translated into several knowledge representation models suitable for logical inference, such as production knowledge representation or predicate calculus. For example, one of the statements generated from Fig. 1 might look like

$$(\exists X) \quad \texttt{sensitivity}(X) \wedge X > 400 \leftarrow \texttt{sport\_shooting(required)} \vee \vee \texttt{night\_shooting(required)}$$

or, as a production rule,

```
if sport shooting is required
or night shooting is required
then sensitivity is more than 400
```

If the whole model uses simplified AND/OR tree notation, it can be expressed as simple proposition calculus statements, in the following manner (see Fig.2):

$$\begin{aligned} \texttt{payment\_complete} &\leftarrow \texttt{cash} \vee p_1 \\ p_1 &\leftarrow \texttt{documents} \wedge \texttt{bank\_transfer} \end{aligned}$$

In addition, the model contains information about higher level domain concepts that can also be captured in the knowledge representation formalism, for example

```
part_of(camera_parameters, sensitivity)
part_of(requirements, night_shooting)
...
```

However, this approach is particularly fruitful when production-frame knowledge representation [8] is used to represent the model — in which case atomic data flow values become frame slots that are grouped into higher-level concepts represented by frames. In this case individual production rules become attached to frame slots of the corresponding frames, in the following manner:

```
frame requirements contains
  sport shooting is (required, not required)
  ...
end
if requirements(sport shooting) is required
or requirements(night shooting) is required
then parameters(sensitivity) is more than 400
```

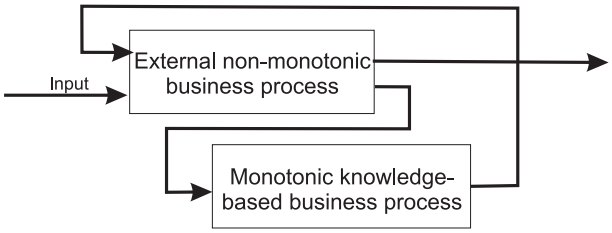

Figure 5. Using external business process to avoid monotonicity constraint

3.2. Logical Inference and Process Execution Order

As it has been pointed out above, the process of forward chaining logical inference in production knowledge based system is simulating business process execution, i.e. when certain data flow values become available — a set of applicable business processes become active and can be executed.

However, it also makes sense to apply backward chaining inference to the same knowledgebase — in which case we start with the desired goals of business process execution, and discover which parts and values of the initial data are required to achieve those goals. In this manner, backward chaining inference can be very useful in both the process analysis / simulation, and in real decision support systems / expert systems. For example, if the business process model describes the process of obtaining certain document in the legal system, then the application of backward chaining inference would allow us to determine which initial documents are required for our particular case.

3.3. Overcoming Monotonicity Constraints

The constraint of knowledge-based business process being strictly monotonous can be very strict for real-life situations. Partially non-monotonic processes can be modeled as a combination of knowledge-based process and some external process that is not based on logical inference, eg. algorithmic (see Fig.5). This external process can "drive" the cycles of logical inference that take place in the underlying knowledge-based process, and thus describe more complex non-monotonic behaviour of the overall system.

As an example, consider the following way to reduce non-monotonic inference to monotonic in an interactive expert system (similar to the approach used in Yandex Guru buying advisor [9]). At each step, the set of user answers is maintained, and the user has an option to remove or change any of his previous answers. Logical

inference is started from scratch at each step, leading the system to some point at which another answer is needed from the user; in which case it is stored in the answer set and the inference is started from scratch. This approach can be well described using the technique presented here.

4. Structural Functional Decomposition as Knowledge Extraction and Representation Technique

It is universally accepted that the bottleneck in the development of knowledge-based systems lies in knowledge extraction from the expert and its formalization and representation in the form of a knowledgebase. In most of the cases, the knowledge of the expert is not formalized and is often very complex, thus knowledge engineer needs some techniques to deal with this complexity. Most commonly used technique is abstraction, which is typically realized through inheritance in frame-based systems and ontologies.

However, there are also other ways of decomposition that can be used to deal with complex processes — in particular, **structured functional decomposition** that is widely used in business process analysis. In functional decomposition, **processes** connected with some **data flows** are considered and structurally decomposed into more primitive processes, which also results in the decomposition of data flows. Typical methodologies of structured functional decomposition are SADT [10] and IDEF-0 [11].

The use of structured functional decomposition techniques is particularly convenient because they allow for natural graphical notation for analysis. In ontology modeling hierarchy diagram is used to express inheritance relationship between domain and abstract concepts, but there are no effective means to graphically describe and/or decompose dynamic properties of domain objects, expressed in terms of axioms, rules, etc. Traditional graphical notations used in this context are decision trees and AND/OR trees, but they do not allow for natural graphical decomposition. We show that structured functional decomposition can be used to cluster domain knowledge into relatively small independent entities, whose behaviour can be further expressed using traditional knowledge representation techniques. Thus, proposed technique provides a convenient graphical notation for expressing functional decomposition of domain knowledge, in the same manner as inheritance hierarchy provides the notation to express abstraction relation between domain concepts.

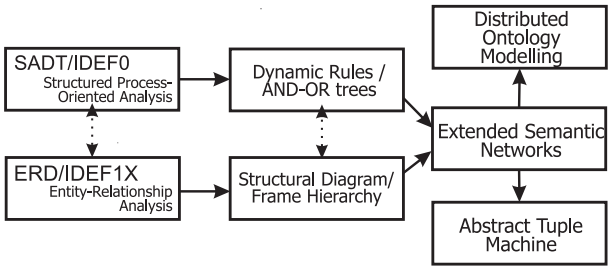

Figure 6. Set of related methodologies used for knowledge extraction and process modeling

5. Methodology for the Development of Knowledge-based Systems for Business Process Modeling

Presented approach for functional knowledge decomposition can form the basis for the methodology for the development of knowledge-based systems in the outlined class of problem domains. The methodology consists of several related graphical modeling techniques (see Fig.6) that can be used in some combination to describe the behaviour of problem domain concepts and use this description in the development of knowledge-based information systems.

5.1. Entity-Relationship Models

During the development of information systems SADT/IDEF0 modeling is typically paired with Entity-Relationship modeling using IDEF1X/ERD notations [12]. Entity-Relationship diagram represents data structure of domain concepts, and is typically obtained from the decomposition of data flow arrows, much in the same way as we suggested capturing domain concepts in terms of frames. However, in addition to structure of problem domain concepts, entity-relationship diagram also defines relationships between concepts, including their cardinality and other properties. Those relationships are not directly represented on the structured functional diagram.

As it has been described in [4], relational database model with entity relationships can be translated into the frame model, in which each domain entity is represented by a frame, and relationships are transformed into frame relations (see Fig.7). This frame model can be enriched by production rules generated from structured functional diagram forming the ontology of domain concepts, and all actual domain data, stored in the relational database, can then be seamlessly used in reasoning by being represented as a family of pseudo-frames that inherit from ontology concepts. Thus, using ERD diagram in conjunction with functional modeling can more fully describe problem domain and create database structure that can be used together with procedural dynamic knowledge in the production-frame representation.

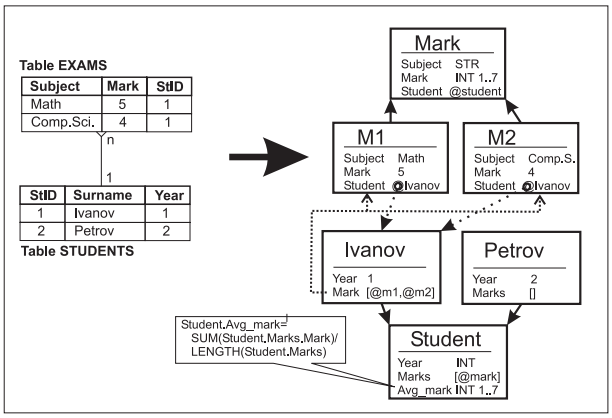

Figure 7. Relationship between relational database and frame structure

5.2. Distributed Ontology Modeling and Process Inheritance

Once we have transformed structured functional model into production-frame knowledge representation, it is possible to use frame inheritance and bring the notion of inheritance to business processes, which leads to the notion of business process ontologies [13]. If the knowledge-based business process is fully defined by a set of frames and attached production rules, it should be possible to inherit from those to obtain slightly modified and refined version of the business process. For example, if the generic process of photo camera selection and sale is defined, it is possible to derive a specialized process that would take into account discounts offered by some particular manufacturers.

Also, it is possible to use the technology of distributed frame hierarchy [4] to use remote business process repositories in the process of logical inference. Such a technology would be useful in the modern mobile environments because if would provide seamless access to reasoning from mobile devices such as cell phones and PDAs, inside the uniform framework of frame-based reasoning.

Another possibility of distribution in business process descriptions is spatial distribution, where different subprocesses have to be carried out at distant locations. In this case it is possible to separate the parts of a knowledgebase responsible for different subprocesses, and have the whole process execution be modeled by a set of cooperating knowledgebases with remote references. It is also quite natural to support the idea of spatial process distribution in the graphical structured functional model by providing some additional labeling to specify the location of each data flow and/or process execution.

It has to be noted, however, that inheritance is natural for frame diagram, but not for the structured functional decomposition diagram itself, which is inherently non-inheritable. This is mainly because inheritance is defined in terms of *concepts*, and functional diagram focuses on *processes* that operate on those concepts. Thus, process inheritance can be realized through concept inheritance, but because structured functional diagram concentrates on processes and not concepts — this cannot be naturally incorporated into functional decomposition diagram.

6. Directions for Further Research

Structured functional decomposition model typically contains several layers of decomposition of processes and data flows, which have to be transformed into planar attribute-value or frame-based model, which loses part of the decomposition semantics. It would be interesting to see whether some other models, like complex hyper-graph structures or hypermatrices can be used to capture that semantics, in conjunction with other models, or by itself, by providing uniform consistent execution semantics much in a same way as extended conceptual graph notation proposed in this paper.

From the practical viewpoint, it would be interesting to consider whether some other reference-based translation into families of frames may be used, which would allow preserving each level of decomposition of domain concepts into more primitive ones. Also, it may make sense to represent actual processes by separate frames linked together using references, that shall give more flexibility to process inheritance. Another possibility would be to consider the subsumption semantics of frame-based systems and see how it can be formulated in terms of knowledge-based business process model.

One of the important constraints of the knowledge-based modeling described in this paper is the monotonicity constraint. While we have proposed some ways to deal with it using external processes that are not modeled in the knowledge-based fashion, it does not seem to suffice for all real-world problems that might require more flexible process behaviour. To support non-monotonic business processes, some knowledge representation and logical inference models with truth maintenance would be required, and the study of suitability of those models for process modeling is an important continuation of the work presented here.

Another prominent direction of research is the development of effective graphical models for business process inheritance that would be logically separated from frame-based representation. The idea of creating business process repositories that are not only based on design patterns, but on actual inheritance should allow much more flexible reuse of knowledge in the area of process modeling.

7. Conclusion and Related Work

We have presented a technique based of structured functional decomposition that can be used for knowledge extraction and modeling of processes in certain

problem domains. There are many methodologies developed both for detailed business process analysis (IDEF3, UML, etc.) and for knowledge representation and extraction (IDEF5, etc.). However, presented technique has an advantage of being based on an industry-standard method of process decomposition that is both widely used for analysis of existing processes, and is very well known. Thus, it can be used to extend existing process models with lower-level knowledge-based models of atomic processes to obtain a knowledgebase that would model process execution.

Modeling technique described in this paper has been partially implemented in a prototype of Dixi CASE modeling tool on Microsoft Visio 2003 platform. The tool supports SADT modeling of business processes, and further decomposition of atomic processes using free-text set of rules in a simple production language, and in a graphical notation of extended AND/OR trees. This process model can be converted into production-frame knowledge representation in JULIA and LIMA formats [8], which can be directly used for the creation of Java- and .NET-based knowledge-based reasoning systems respectively. Since those reasoning engines support seamless integration with relational databases, it makes the proposed set of tools ideal for modeling data-intensive business processes with the access to data stored in the databases, building business process simulation and execution layer on top of the exisiting data infrastructure.

As an example of using the proposed technique, a knowledgebase for the selection of digital cameras in an internet shop has been created using described methodology, and the reasoning engine with web service interface has been implemented using LIMA .NET production-frame reasoning compiler. A web-client and PDA native client have been developed, allowing to perform the process of consultation from a mobile platform. The corresponding project has been presented on the Microsoft Imagine Cup 2004 competition, and was awarded the second place in the CIS Regional Final.

Modeling technique based on functional decomposition fits well into the infrastructure of other data and knowledge representation models as presented on Fig.6. That family of models can be used to describe properties of objects and processes in the problem domain fully, and translate them into representation (production-frame or extended conceptual graphs/ tuple abstract machine) suitable for automated reasoning, either for the actual process automation, or for the process simulation.

Acknowledgements

The material presented in this paper is based on research partly supported by the grant of President of Russian Federation No.MK-2569.2004.9, and by the grant of Russian State University of Business and Innovative Technologies.

References

1. Giaglis G.M. "A Taxonomy of Business Process Modeling and Information Systems Modeling Techniques". *International Journal of Flexible Manufacturing Systems*, 2001.

2. Soshnikov D. "Interoperability Semantics in Distributed Frame Hierarchy". In: *Proc.* 4th *Computer Science and Information Technology Workshop*, 2002.

3. Dubovik S., Soshnikov D. "Using Semantic Networks Extended with AND/OR Trees for Structural and Dynamic Knowledge Representation in Knowledge-Based Systems." *Trudy MAI*, 2003; No.13. (In Russian)

4. Soshnikov D. "Data and Knowledge Representation Models of Distributed Frame Systems". In: *Proc. Pre-Conference Workshop of VLDB-2003 "Emerging Database Research in Eastern Europe"*, Brandenburg University of Technology at Cottbus, 2003, pp. 123-127.

5. Bosilj V., Hlupic V. "Petri nets and IDEF diagrams: Applicability and efficiency for business process modelling." *Informatica*, 2001; 25(1):123-133.

6. Luger G.F., Stubblefield W.A. "Artificial Intelligence: Structures and Strategies for Complex Problem Solving", Addison Wesley, 2002.

7. Sowa J.F. "Conceptual Structures: Information Processing in Mind and Machine", Addison-Wesley, 1984.

8. Soshnikov D. "An Architecture of Distributed Frame Hierarchy for Knowledge Sharing and Reuse in Computer Networks". In: *Proc. 2002 IEEE Intl. Conf. on Artifical Intelligence Systems*, 2002, pp. 115-119. IEEE Computer Society Press.

9. Yandex Guru Buying Advisor Web Site: `http://guru.yandex.ru` (In Russian)

10. Marca D., McGowan C. "SADT: Structured Analysis and Design Technique", McGraw Hill, 1988.

11. IDEF0: Integrated Definition for Function Modeling, National Institute of Standards and Technology Publication No. 183, 1993.

12. Chen P.P. "The Entity-Relationship Model — Toward a Unified View of Data". *ACM Transactions on Database Systems*, 1976;1(1):9-36.

13. Aitken S., Curtis J. "Design of a Process Ontology: Vocabulary, Semantics and Usage". In: *Proc.* 13^{th} *International Conference on Knowledge Engineering and Knowledge Management*, 2002, pp. 108-113. Springer Verlag.