Structured Functional Decomposition Approach to Knowledge-Based Business Process Modeling

Dmitri SOSHNIKOV, Sergey DUBOVIK
Moscow Aviation Technical University, Volokolamskoye shosse 4, 125993 Moscow, Russia

Abstract. This paper presents an approach for structured decomposition of knowledge in certain problem domains, which can be classified as knowledge-based business processes. Knowledge in such problem domains is already partly formalized, and allows natural functional decomposition; however, it is complex enough to justify knowledge-based implementation of process execution. Examples include, in addition to many complex business processes, such areas as law consulting, notary services, etc.

The approach is based on structured functional decomposition towards primitive processes that can be described using traditional knowledge representation techniques, such as AND/OR trees or production rules. In this decomposition, 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 demonstrated that forward-chaining logical inference in resulting knowledgebase is modeling business process execution.

Some details of implementation are also discussed, and the modeling tool supporting the proposed approach on Microsoft Visio platform is presented. The role of the proposed modeling technique in the overall methodology for knowledge-based systems development is outlined.

1 Introduction

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 [1] and IDEF-0 [2].

This paper describes how structured functional decomposition can be applied to the process of knowledge extraction and modeling of complex knowledge-based problem domains. While this technique can be useful in quite different areas of expertise, we specifically focus on the subset of problem domains that can be classified as **knowledge-based business processes**, i.e. workflows, where many decisions are taken non-deterministically based on expert knowledge. In such problem domains, knowledge may already be partly formalized in the form of some free-text instructions, or even better modeled as a set of structural diagrams, but still the execution of business process for a particular case is complex enough to be described in a knowledge-base, and not inherently algorithmic. Such processes appear in the areas of law consulting, notary actions, workflow modeling, selection of products and services on e-business portals, etc.

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 [3]), 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 simulations.

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.

2 Knowledge-Based Business Process Modeling

2.1 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 process.

[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.

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 $\mathfrak{S}$. 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 $\mathfrak{S}'$ and $\mathfrak{S}''$ – 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. $\exists D_j \in \mathrm{output}(B_i) \quad B_i(\mathcal{S})|_{D_j} \neq \perp$. Each function B_i can also be extended to the whole set $\mathfrak{S}$ in a natural way. 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 $B_{i_1} \circ B_{i_2} \circ \cdots \circ B_{i_n}$, 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. Theoretically, from each initial state $\mathcal{S}_0$ there is a whole set of compositions that lead to final state.

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 one discreet output and several 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 [4].

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 $\{D_{jl}^{(k+1)}\}_{l=1}^{n}$ with values from domains $\mathbb{T}_l$, then the value of $D_j^{(k)}$ would belong to $\mathbb{T}_1 \times \cdots \times \mathbb{T}_n$. 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.

2.2 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.

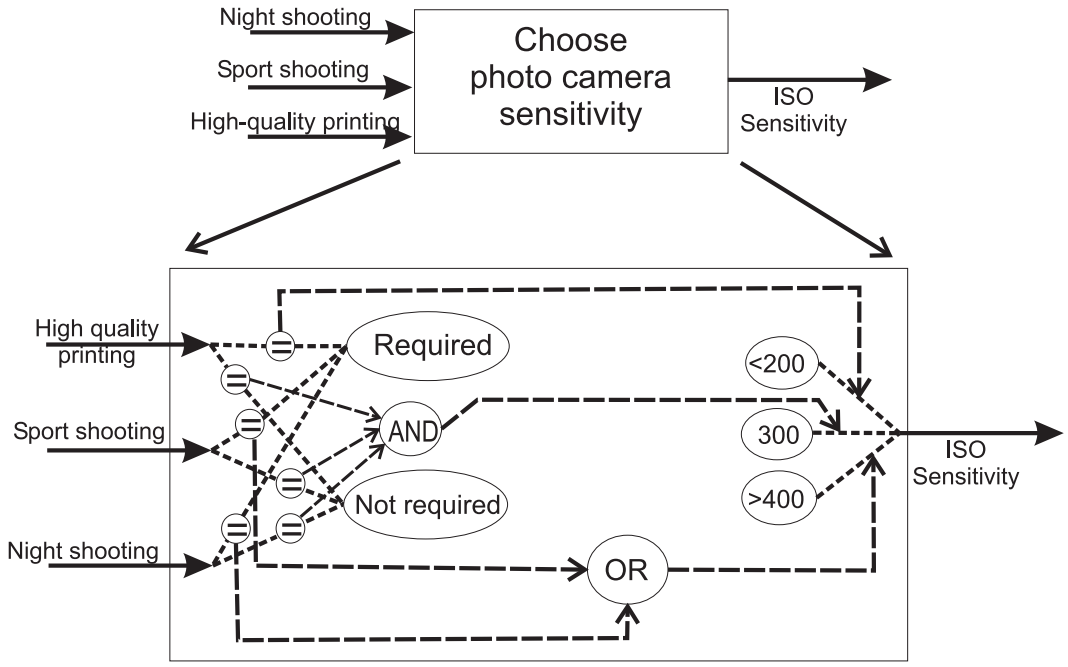

Figure 1: Atomic business process decomposition using AND/OR trees

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 [5, 6], 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).
- **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

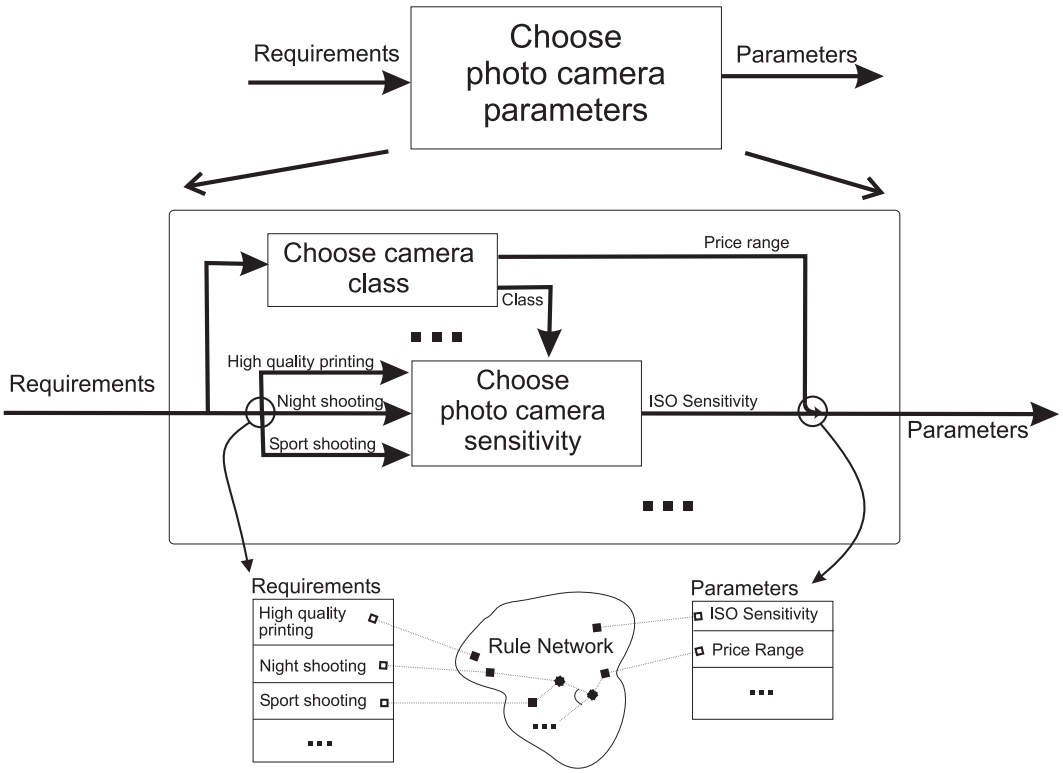

Figure 2: Frame induction from data-flow decomposition and multi-level business process analysis

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.

2.3 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.2. 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.

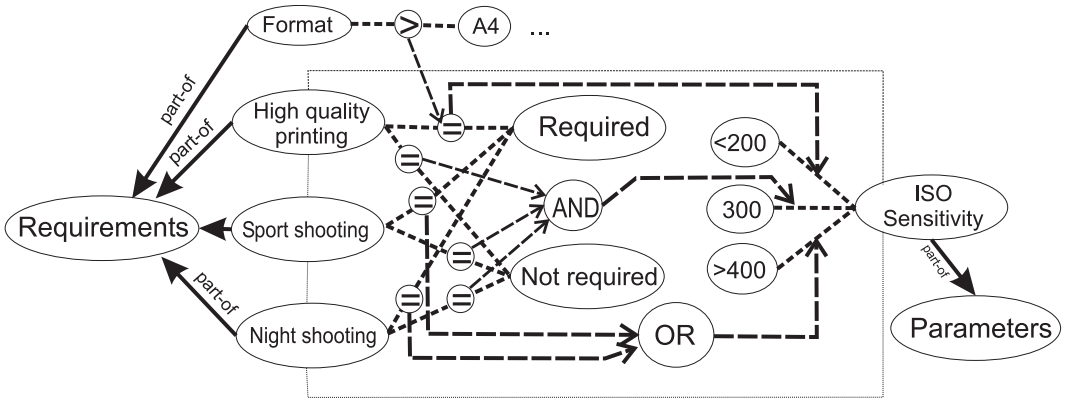

Figure 3: Transformation of business process diagram into extended conceptual graph

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 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

2.4 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 [7, 8] 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.3). 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 [5, 6]. 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 [6], 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.5.

2.5 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 combi-

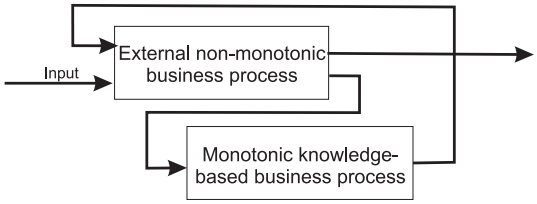

Figure 4: Using external business process to avoid monotonicity constraint

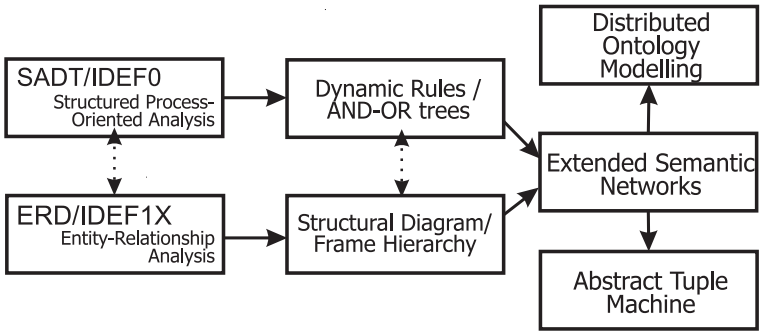

Figure 5: Set of related methodologies used for knowledge extraction and process modeling

nation of knowledge-based process and some external process that is not based on logical inference, eg. algorithmic (see Fig.4). 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.

3 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.5) 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.

3.1 Entity-Relationship Models

During the development of information systems SADT/IDEF0 modeling is typically paired with Entity-Relationship modeling using IDEF1X/ERD notations. Entity-Relationship dia-

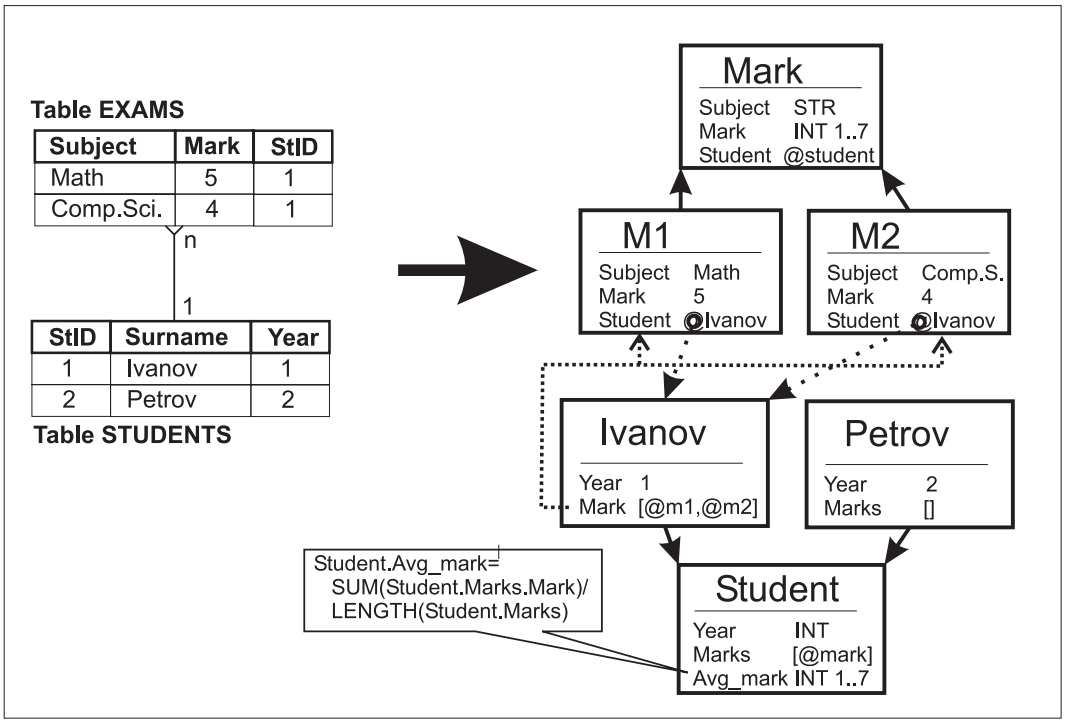

Figure 6: Relationship between relational database and frame structure

gram 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 [6], 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.6). 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.

3.2 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 [10]. 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 [6] 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.

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.

4 Implementation Details

Modeling technique described in this paper has been partially implemented in a prototype of 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. The tool supports generation of production-frame knowledgebases in JULIA and LIMA formats [11], that can be immediately used for the creation of Java- and .NET-based knowledge-based reasoning systems respectively.

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 named Dixi has been presented on the Microsoft Imagine Cup 2004 competition, and was awarded the second place in the CIS Regional Final.

5 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, and whether some other reference-based translation into families of frames may be used. 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 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.

6 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.

In addition, modeling based on functional decomposition fits well into the infrastructure of other data and knowledge representation models as presented on Fig.5. 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.

7 Acknowledgements

The material presented in this paper is based on research supported by the grant of President of Russian Federation No.MK-2569.2004.9 for the support of young scientists.

References

[1] Marca D.A, McGowan C.L. SADT: Structured Analysis and Design Technique. McGraw Hill, 1988.

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

[3] Giaglis, G.M. A Taxonomy of Business Process Modelling and Information Systems Modelling Techniques. *International Journal of Flexible Manufacturing Systems*, 2001.

[4] Soshnikov D. Interoperability Semantics in Distributed Frame Hierarchy. In *Proc. of* 4^{th} *Computer Science and Information Technology Workshop*, Patras, Greece, 2002.

[5] Dubovik S., Soshnikov D. Using Semantic Networks Extended with AND/OR Trees for Structural and Dynamic Knowledge Representation in Knowledge-Based Systems. In *Moscow Aviation Institute Electronic Journal "Trudy MAI"*, No.13, 2003. (In Russian)

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

[7] Luger G.F. Stubblefield W.A. Artificial Intelligence: Structures and Strategies for Complex Problem Solving. — Addison Wesley, 2002. (4th Ed.)

[8] Sowa, J.F. Conceptual Structures: Information Processing in Mind and Machine, Addison-Wesley, 1984.

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

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

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