Modeling of Microsystems Production Processes for the *MinaBASE* Process Knowledge Database Using Semantic Technologies

Daniel Kimmig*, Andreas Schmidt†*, Klaus Bittner*, and Markus Dickerhof*
*Institute for Applied Computer Science
Karlsruhe Institute of Technology
Karlsruhe, Germany
Email: {daniel.kimmig, andreas.schmidt, klaus.bittner, markus.dickerhof}@kit.edu
†Department of Informatics and Business Information Systems,
University of Applied Sciences, Karlsruhe
Karlsruhe, Germany
Email: andreas.schmidt@hs-karlsruhe.de

***Abstract*—In this paper, we present the consolidation of a process knowledge database for knowledge-intensive production processes in the field of microsystems technology with a workflow component. Among the requirements to be met by the workflow component are the hierarchical presentation of process chains, a close integration of the product structure in the form of assemblies, modules, and components, storage of previous (unsuccessful) attempts together with the information arising, the derivation of process patterns from concrete workflow instances, and the explicit modeling of dependencies among various steps in the workflow. In contrast to existing workflows, the complexity in the concrete microsystems technology application does not lie in potential branchings of activities, but in the inherent information and concepts of the process knowledge database and their relations and constraints. Starting from the metamodel developed for process modeling by List and Korherr, a multi-perspective model with four overlapping and integrated perspectives (system, process, project, and development perspectives) was developed to better manage the complexity of and reuse individual knowledge entities. As a proof of concept, the model is implemented by means of formal knowledge representation languages from the semantic web, which will be illustrated using the previously analyzed development perspective as an example.**

***Keywords*-process knowledge management, microsystems technology, semantic web**

I. Introduction

Knowledge, experience, and capacities of the employees make up the core competencies of an enterprise and have a crucial influence on its competitiveness. The knowledge required for creating values added is no public good, but a business resource that has to be administrated efficiently in order to ensure economic success. For software-technical support, knowledge management systems [1] have been established. In process-oriented knowledge management [2] highly knowledge-intensive production processes in microsystems technology, for example, are managed. Production processes in this field are characterized by a high interdisciplinarity, many process steps, and a low standardization. Frequently, a product is produced by an individually tailored production process [3]. Moreover, design decisions during the development of microsystems are strongly dependent on the characteristics of the different fabrication technologies applied. They may require knowledge of various disciplines like microoptics, biotechnology, or sensor technology. In practice, technical experts have to deal with heterogeneous, distributed, and partly incomplete data, such that new or already solved product development problems are difficult to handle. Knowledge management methods represent an promising approach to overcoming this barrier. The Institute for Applied Computer Science has developed the *MinaBASE* process knowledge database which structurally acquires the expert knowledge of microsystems technology and makes this knowledge available centrally, homogeneously, and collaboratively [4]. However, the *MinaBASE* approach does not provide for any process-oriented linking of these structured knowledge entities. Linking would allow not only for the representation of the knowledge required for microsystems production processes, but also for the modeling of these processes on an abstract level. So, an approach to process-oriented linking of existing process data will be presented in the following sections.

The paper will be structured as follows: The next section will present the underlying process knowledge database *MinaBASE*. Then, requirements made on process modeling in this context will be highlighted and existing process modeling standards will be analyzed for suitability. A solution approach will be described and implemented using semantic technologies to derive implicit knowledge from the modeled facts. A part of modeling as ontology will be described in detail in Section VI.

II. *MinaBASE* Process Knowledge Database

The *MinaBASE* process knowledge database was developed within the framework of the MikroWebFab joint project funded by the BMBF [2]. Technology partners

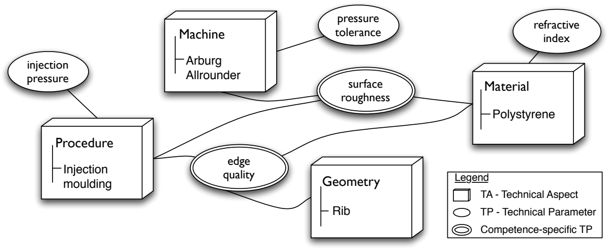

Figure 1. Schematic representation of a *MinaBASE* competence

of a virtual enterprise used it for the structured storage of technical production parameters of the processes and materials used in microsystems technology and of partner-specific technical competencies. The so-called technical aspects (TA) that serve to model materials, machines, and fabrication technologies are the smallest information entity in MinaBASE [5]. TA are arranged in taxonomies using generalization hierarchies. The number and contents of taxonomy trees can be specified and modified during runtime, such that a flexible structure tailored to meeting the requirements of microsystems technology can be defined for the storage of production knowledge. TA can be assigned properties that are referred to as technical parameters (TP). A TP is specified as a character string, integer, or floating-point number and references an attribute, e.g., density. As in the object-oriented approach, the TP of a TA are passed on to partial hierarchies located below in the hierarchy tree. In addition, lower hierarchy levels can further refine the inherited TP by specifying general value ranges.

To model the capacities of a technology partner, competencies [4] are considered a set of various TA of disjunct hierarchy trees, which is illustrated in Figure 1.

Here, the competence of ”injection molding of a polystyrene web using the Arburg Allrounder machine”, with several TP is represented schematically. The respective TA are selected from the hierarchy trees of process, machine, material, and geometry element, with the TA having own TP, such as ”injection pressure” of the injection molding process. Combination of these TA yield the competence having other TP, such as the edge quality and surface roughness. Consequently, a competence is a type of view on a certain combination of TA with properties in the form of TP that are only valid for this combination and, hence, characterize the competence in more detail. Accordingly, TA can be used in several competences, which illustrates their role as reusable, encapsulated, smallest information entity.

III. Requirements

In this section, the boundary conditions of and requirements on process modeling for *MinaBASE* shall be analyzed. Five types of requirements can be distinguished and will be described in more detail below.

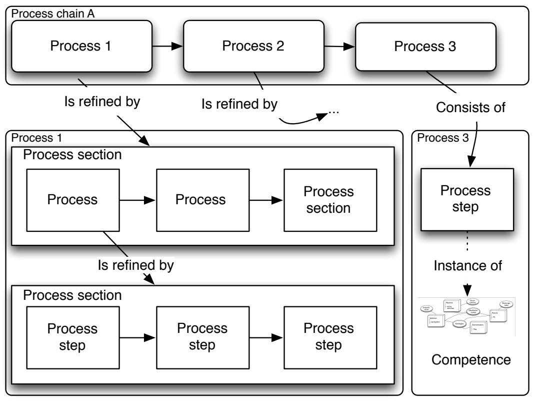

Figure 2. Hierarchical process chains for modularization

A. Hierarchical Process Chains and Variants

In process modeling for *MinaBASE* it is crucial to arrange process chains hierarchically for reducing complexity. In this way, basic information can generally be presented on higher levels, while details are hidden on lower levels. This is illustrated in Figure 2.

The subelements of process chains are single processes. Hierarchical process chains are characterized by the fact that the process elements involved can be refined in any way. In the example the first process is refined by a so-called process section. Process sections are a special process element, they represent well-defined workflows, such as the LIGA process in microsystems technology, which combines the techniques of lithography, electrodeposition, and molding [6]. The first element of the upper process section is refined by another process section which consists of atomic process steps only. They correspond to an instancing of competences. If this direct allocation of process step to competence does not yet exist, e.g., in the early development stage of the process chain, it must be possible to model technologies within an atomic process element as well as complete subprocesses as alternative variants for a part of the process chain.

B. Integration of Product Structure and Process Chains

To describe the setup of a microsystem, it is recommended to store its components in a product structure (cf. Figure 3), by means of which a microsystem can be set up and structured according to the construction kit principle. The first structuring means are modules that encapsulate a logical functional area and hide the details from other modules. Individual components having certain functions can be joined to a logical entity by assemblies. Theoretically, a module is composed of a few single components or of a large number of assemblies. To produce a complete microsystem, the process chains resulting in components and the integration of these components are relevant. Joining of the individual components and assemblies requires or represents an own type of processes integrating the components as results of the process chains of the microsystem.

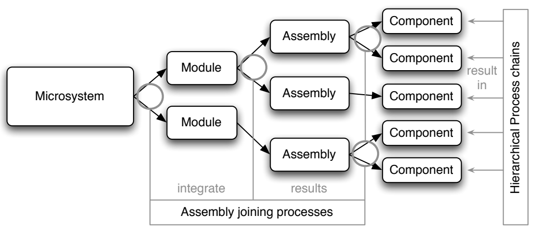

Figure 3. Allocation of process types to the product structure

C. Versioned Documentation of Solution Approaches

On the way towards finding a solution for a technical problem, many data are generated, which will be very valuable when solving future problems [3]. If acquired at all, they exist, e.g., in the form of a text or table document often decentrally without any relationship to the solution process. Therefore, these data have to be combined manually. This is aggravated by the fact that the data are often administrated by different persons and exist in a heterogeneous structure. Another obstacle lies in the fact that success only is documented in many cases, but not the errors made on the way towards it. But it is the information whether and why a certain approach did not work in the past, which needs to be available when implementing new ideas, such that errors already made will not be repeated. These circumstances frequently make the implementation of new ideas time-consuming and expensive. Good ideas are rejected in the beginning already, only because an adequate and collaborative access to information is lacking [7]. To overcome this problem, process chains must be stored in various versions for the same technical problem or components of the product structure and easy to reproduce. A new version may result from the fact that experiments for the process model applied so far have shown that parameters of certain materials affect the quality required in an unexpected adverse way.

D. Derivation of Templates from Ad hoc Workflows

A decisive criterion of efficient knowledge management is access and an efficient reusability of knowledge entities. In microsytems technology some basic processes are distinguished, such as ”disposable” or ”AVT”. They require similar workflows in each case. It must therefore be possible to derive project-independent patterns from project-specific process chains, which may then serve as documentation or templates for new projects by individual instancing, as shown in Figure 4. Patterns are supposed to accelerate the development of new products by reusing existing templates.

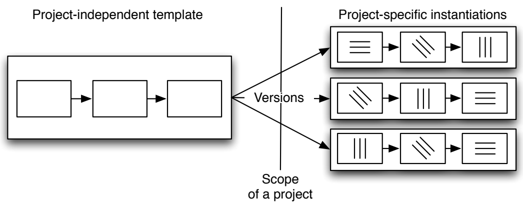

Figure 4. Abstraction of templates by project-specific instancing

E. Modeling of Dependences

Apart from defining the structure and order of process elements, it is important to express dependences of process elements. This significantly extends the capability of modeling process chains, as not only the structure alone is of interest, but also the fact why exactly this structure is required. In addition, process elements may have certain TP as a prerequisite. The possibility of expressing the following types of dependences explicitly facilitates the finding of errors in an early phase, in which these errors can be eliminated at low costs.

- *PreConditions* - A process element is executed under a condition, for example, the existence of an applied layer.
- *DuringConditions* - Circumstances prevailing while executing a process element, for example, the temperature of a production process. When using a process, TP may occur, which can now be documented.
- *PostConditions* - Express that consistency conditions, such as the observation of a certain TP, shall remain valid even after the execution of the process element for all following process elements.
- *Effects* - Are the result of a process element, for example, reaching of property required by a specification. Previous post-conditions are overwritten, if, e.g., a lacking thermal resistance due to the application of an insulation layer is defined as an effect of a later process element.

Having formulated own conditions and effects, other technical experts can compare them with own dependences so that cooperation is supported. Modeling of the dependences provides for an explicitly formulated representation that can be communicated and evaluated automatically.

IV. MinaBASE Process Modeling

This section will focus on standards of business process modeling. Then, their suitability for *MinaBASE* process modeling will be studied taking into account the requirements described above. Finally, the *MinaBASE* process model designed will be presented.

A. Standards of Process Modeling

1) Event-driven Process Chains: Event-driven process chains [8] are part of the ARIS concept (architecture of integrated information systems) [9], in which they act as graphical, semi-formal modeling language for business processes. An event-driven process chain is a directed graph, whose nodes consist of alternating events and functions as well as logical connectors. An event is a state initiating a function or initiated by the latter. Logical connectors allow for the splitting and combination of parallel or alternative workflows by the interconnection symbols `AND`, `OR`, and `XOR`. The ”process path” is used to reference partial processes. By ”extended event-driven process chains”, event-driven process chains are extended by notations for organizational units, data objects, and services.

2) Petri-Nets: Petri nets [10] have a formal mathematical basis [11], [12] and are used for modeling and simulating business processes as well as for conceiving concurrent and parallel algorithms. A Petri net is a directed, bipartite graph containing two types of nodes, the transition for events and the place for conditions. Places may contain marks. During the so-called ”firing”, these marks are removed from the input places of a transition and newly generated in the output places, as a result of which the net can be run as a simulation. Hierarchical Petri nets [13] allow for the storage of partial processes in own nets. The predicate transition nets contain structured marks representing objects [14], whose state can be modified by calculations in transitions.

3) Business Process Modeling Notation: The Business Process Modeling Notation (BPMN) [15] is a semi-formal modeling language for business processes with a small formal basis. BPMN is used to define a workflow that is translated into languages like BPEL (Business Process Execution Language) in order to integrate web services in the processes by a ”service-oriented architecture”, for instance. A central element is the ”business process diagram” that consists of ”flow objects” (atomic and composed activities for subprocesses, gateways as connectors, notations for events), ”connecting objects” (edges for the workflow and information flows), ”swimlanes” (allocation of roles), and ”artifacts” (information objects and metadata) [16].

4) Activity Diagrams: The Unified Modeling Language (UML) is a formal and visual modeling language for the design and documentation of artifacts of software systems. The UML defines various types of diagrams [17] to model, e.g., the structure (class and component diagrams) and behavior (sequence and activity diagrams). Activity diagrams were used for the process specification language [18] to model production processes. They contain actions as elementary elements that model complex behavior by chaining with control and object flows and logical connectors. Control flows are directed edges specifying the sequence of actions. Object flows extend this semantics to represent data flows of objects along an edge. An action of an activity can be structured hierarchically to represent the exact workflow in another diagram.

B. Applicability of the Process Modeling Standards

The first requirement in Section III-A deals with the hierarchization of process chains and the possibility of representing variants. All standards analyzed in IV-A support own forms of subprocesses that can be referenced. However, they do not directly provide for the modeling of technical variants. It is possible to note several variants in a running text indexing symbols for functions, but this is a rather informal approach that is difficult to evaluate automatically when verifying the process model in terms of the dependencies modeled. For this reason, the requirement is met partly only. Requirement III-B asks for an integration of process chains with the components of a microsystem to make it clear which component is produced by which process chain. *ARIS* defines a performance view for the representation of results of process chains, a general product model, a hierarchical product tree for the event-driven process chains meeting the requirement. Petri nets do not support any form of product models and do not meet the requirement. BPMN and activity diagrams allow for the modeling of unstructured objects. Activity diagrams support typed object flows. BMPN uses ”data objects” that stand for used documents. As both standards do not allow for hierarchical objects, they meet the requirement partly only. Requirement III-C focuses on the support of an iterative, collaborative, and centrally available project documentation of solution approaches for them to be used for the development of new microsystems and for the later reproduction of errors and experience gained during previous developments. As all standards support a serialization by, e.g., XML data formats, this requirement can be met in principle by all standards. Requirement III-D covers the storage of process chains as templates for new processes. In principle, all standards are capable of using so-called reference processes given in a top-down manner. They have to be adapted to the existing conditions. As a derivation of process templates from existing processes is much more important for *MinaBASE*, however, the standards meet this requirement partly only. Requirement III-E deals with the modeling of dependences of process elements. Due to the informal character of event-driven process chains, dependences can be expressed as running text only, such that the standards do not meet the requirement. In Petri nets events are modeled, which are executed only after their preconditions are met and result in post-conditions. These are expressed by the structure of the process chain. This aggravates the allocation of knowledge entities of *MinaBASE*, such as TA and TP or effects for product properties as functions of individual process elements. Consequently, Petri nets meet this requirement partly only. The UML contains the Object Constraint Language (OCR), such that constraints and conditions can be modeled. However, programming knowledge in OCR is required. BPMN has explicit language constructs for the waiting for the receipt of messages or other events, such that a simple type of dependences can be modeled. Consequently, BPMN and activity diagrams meet the requirement partly.

To sum up, none of the standards described fulfils all requirements to the complete extent. As a result of the ARIS concept, event-driven process chains are the standard fulfilling most of the requirements, with informal modeling aggravating the storage of dependences that can be evaluated automatically. BPMN lacks the formal semantics of a metamodel for being extended such that the limits of graphical notation are overcome. Petri nets do not fulfill two requirements and in spite of their strongly formal basis, they can hardly be applied for *MinaBASE* process modeling. It is found that in spite of varying strengths and weaknesses, none of the standards fulfils the requirements to a sufficient extent and that an individually tailored knowledge representation is suited best.

V. Perspective Model for *MinaBASE*

A generic metamodel for standards of process modeling was developed in [19]. Based on this model, the standards described above (see Section IV-A) were evaluated. When comparing this metamodel with the requirements made on *MinaBASE* process modeling, it is found that even the metamodel that combines the modeling capabilities of many standards by various perspectives meets the requirements to a limited extent only. Hence, it is absolutely necessary to conceive an individually tailored solution, since an expressive methodology for business process modeling is not suited for *MinaBASE*, because the complexity does not lie in the branching of activities, but in the information, concepts, their relationships and constraints associated with the activities, and above all in the implicit knowledge resulting from combination. An integrated solution approach to knowledge representation has to define own perspectives in order to close the gap between the requirements and the generic metamodel and to delete the unused perspectives from the metamodel. The multi-perspective model developed here is shown in Figure 5. This model is divided into four overlapping and integrated perspectives and, hence, facilitates the management of complexity and reuse of individual knowledge entities. The system perspective contains the product structure from requirement III-B, i.e. the structured grouping of microsystems and the specifications to be fulfilled by the components. Consequently, this perspective covers everything relating to the setup and functions of a microsystem. The process perspective covers the requirements from III-A, i.e. modeling of workflows in the process chain and the hierarchical structure and variants of possible technologies and competences. The link between the process and system perspectives is the allocation of components of the product structure as results of process chains from requirement III-B. The development perspective expresses the requirement III-E, i.e. modeling of pre-, during-, and post-conditions and of effects of process elements. This separates the logical setup of process chains from their partly automatic verification and validation on the basis of the constraints and dependences expressed by conditions. The project perspective covers the requirements III-C and III-D. Provided that *MinaBASE* is used properly, it is possible to view versions of the process chains and product structures in a historically reproducible manner. This results in an iterative product development cycle and may reduce the consumption of resources of future developments in terms of time and costs.

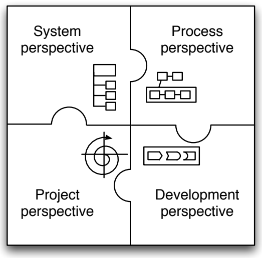

Figure 5. Model for *MinaBASE* process modeling

VI. Modeling in OWL

It is the objective of *MinaBASE* to acquire process knowledge in microsystems technology such that the finding, combination, and, ideally, verification of knowledge can be supported automatically. To ensure this for the model described, it is reasonable to model the perspectives and their concepts by formal knowledge representation languages from the semantic web, e.g., OWL (Web Ontology Language) [20], as this makes the semantic relations described explicit. After acquiring the process knowledge in this form, the next step may be a definition of rules, e.g., with SWRL (Semantic Web Rule Language) [21] and queries of increased content values using SPARQL (SPARQL Protocol and RDF Query Language) [22]. Previous approaches, such as the ARIS concept [23], define a semantic model, but the significance of the contents of the process elements is lost rapidly by marking with a purely free text. An adequate support of the modeler in semantic annotation, i.e. the filling of the process models with contents, usually does not take place [24], such that hardly any automation and machine support is possible, since the interpretation of the concepts, terms, and relations used is left to a human brain. For this reason, knowledge representation of the multi-perspective model is accomplished by formal languages from the semantic web. As it is a principle of *MinaBASE* to separate the so-called build-time from the runtime, i.e. to define the concepts used to structure the production knowledge during runtime, however, the build-time must have a flexible structure allowing for later instancing and adaptation. Hence, the

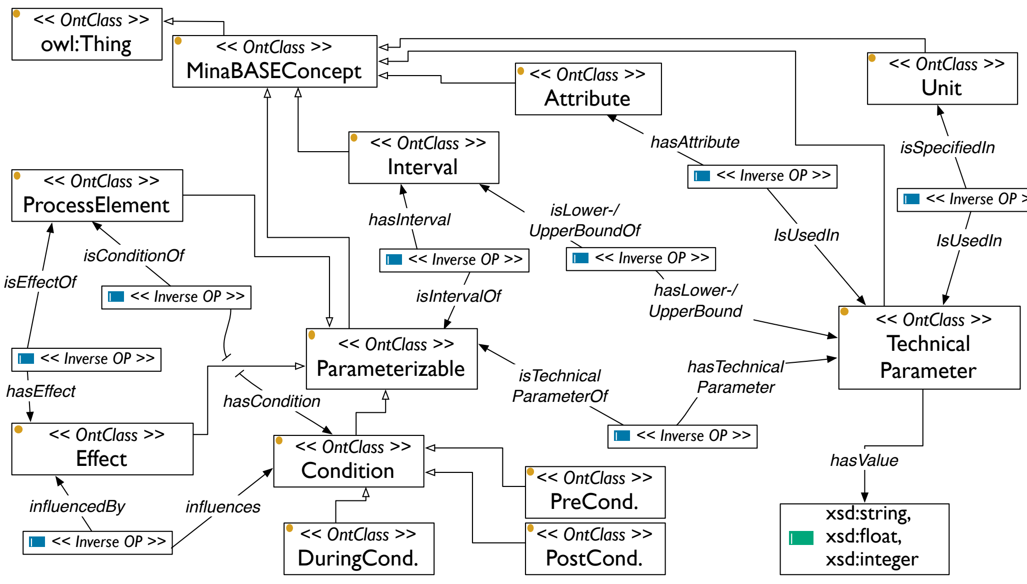

Figure 6. Schematic representation of classes and relationships of the development perspective in OWL

OWL ontology to be generated for the model perspectives should rather be an "upper-level ontology" allowing for the instancing and allocation of process knowledge during runtime. Figure 6 displays a schematic representation of the modeling of the development perspective using the OWL language, which will now be explained in more detail. As a graphical notation, the visual metaphors of the OWL editor *Protege* are used, with a circle representing a class and a rectangle an "object" or "data type property". The uppermost element of the class hierarchy is the class *MinaBASEConcept* that inherits from *owl:Thing* and serves as a central extension point for the classes of the respective perspectives. While the process perspective contains the hierarchical setup of the process elements, the development perspective models the dependences in the process chain using TP, conditions, and effects. The class *ProcessElement* possesses general relations for input and output edges to other process elements in the process perspective and acts as basic class for composed *CompositeProcessElements* like *ProcessSection* and atomic process elements like *ProcessStep*. Hence, the perspectives can be linked without having to combine the details of the perspectives. A central class of this perspective is *Parameterizable* that encapsulates the allocation of TP by the inverse ObjectProperty *hasTechnicalParameter*, such that subclasses of Parameterizable, namely, ProcessElement, *Effect*, and *Condition*, inherit this relation. The classes of *PreCondition*, *DuringCondition*, and *PostCondition* are derived from *Condition*. Their existence is required for referencing in rule languages and implementing their semantics. An effect is linked with conditions via the *influences* relation. In this way, preconditions of previous process elements can be relaxed. By *hasCondition* or *hasEffect*, instances of these classes are allocated to the process elements. Apart from the TP, also intervals can be allocated to instances of the class *Parameterizable* as value ranges by *hasInterval*. The class *Interval* may contain upper and lower limits of TP using the relations *hasLowerBound* and *hasUpperBound*. A TP is implemented by the class *TechnicalParameter* and possesses a data type property *hasValue* for typed values in the form of integers, floating-point numbers or character strings. These are specified in a certain unit via the class *Unit* and reference an *Attribute*, for example, the density. Using the inverse relation *isUsedIn* of the *hasAttribute*, TP comparable in rules can be determined in order to determine proposals for the allocation of new TP when reusing existing process elements for the modeling of new process chains.

VII. Conclusion and Future Work

This paper presented an approach to extending the *MinaBASE* process knowledge database, a system for managing the knowledge in the field of microsystems technology. By means of this approach, the knowledge entities of the system, basic data on processes, materials, and production competencies, can be combined in a process-oriented manner. First, the modeling requirements were presented, which result from the special characteristics of microsystems technology compared to conventional mechanical engineering. In particular, the interdisciplinarity of the expert knowledge required, the low standardization of production methods, and the necessity of an iterative solution of development problems characterize microsystems technology. Central requirements in *MinaBASE* are process chains that can be interlinked in many ways along an "is-Part-of" hierarchy, the modeling of product structures and their allocation to process chains, the definition of dependencies within the process chain, a versioned storage of project-specific documentation, and the possibility of deriving project-independent and reusable process templates for new process chains. Then, standards of process modeling (event-driven process chains, Petri nets, BPMN, and activity diagrams of the UML) were analyzed for their suitability for meeting the requirements described. As none of the existing standards meets all requirements, an individual, multi-perspective, and interlinked model was conceived, which is tailored to meeting these requirements. To not only store the production knowledge in *MinaBASE*, but derive new knowledge from implicit relationships within the knowledge base, technologies from the semantic web were selected to implement the model conceived. As a first step, the concepts and relations of the model were formulated as ontology in OWL and the development perspective was presented. Extension of the ontology by rules for the implementation of the semantics of conditions and effects of the process elements, integration in the existing application architecture of *MinaBASE*, testing of the ontology using data from practice, and a possibly resulting refinement of the concepts selected and relations have been identified as future research topics.

REFERENCES

[1] I. Nonaka and H. Takeuchi, "The knowledge-creating company," *Harvard Business Review*, vol. 6, pp. 96–104, 1991.

[2] M. Dickerhof, "Prozesswissensmanagement für die Mikrosystemtechnik." *Statusseminar MikroWebFab, Karlsruhe*, 2003.

[3] U. Hansen, C. Germer, S. Büttgenbach, and H. Franke, "Rule based validation of processing sequences," in *Techn. Proc. MSM*, 2002.

[4] M. Dickerhof, O. Kusche, D. Kimmig, and A. Schmidt, "An ontology-based approach to supporting development and production of microsystems," *Proc. of the 4th Internat. Conf. on Web Information Systems and Technologies*, 2008.

[5] M. Dickerhof and A. Parusel, "Bridging the Gap—from Process Related Documentation to an Integrated Process and Application Knowledge Management in Micro Systems Technology," *Micro-Assembly Technologies and Applications*, pp. 109–119, 2010.

[6] E. Becker and W. Ehrfeld, "Das LIGA-Verfahren–Mikrofertigung durch Röntgentiefenlithographie mit Synchrotronstrahlung, Galvanoformung und Kunststoffabformung," *Phys. Bl*, vol. 44, no. 6, pp. 166–170, 1988.

[7] D. Ortloff, J. Popp, K. Hahn, T. Schmidt, and R. Bruck, "Tool Support for Microelectronic Process Development," *Mixed Design of Integrated Circuits and Systems, MIXDES 2008*, pp. 467–472, 2008.

[8] A.-W. Scheer, "Semantische Prozeßmodellierung auf der Grundlage Ereignisgesteuerter Prozeßketten (EPK)," *Veröffentlichungen des Instituts für Wirtschaftsinformatik*, 1992.

[9] ——, "ARIS-House of Business Engineering," *Veröffentlichungen des Instituts für Wirtschaftsinformatik*, vol. 133, 1996.

[10] C. A. Petri, "Kommunikation mit Automaten," Ph.D. dissertation, Institut für instrumentelle Mathematik, Bonn, 1962.

[11] T. Murata, "Petri nets: Properties, analysis and applications," *Proceedings of the IEEE*, vol. 77, no. 4, pp. 541–580, 1989.

[12] J. E. Jorg Desel, *Free Choice Petri Nets*. Cambridge University Press, 2005.

[13] R. Fehling, *Hierarchische Petrinetze: Idee und grundlegende Struktur.* Universität Dortmund, Germany; Lehrstuhl Informatik 1, Forschungsbericht Nr. 344, 1990.

[14] P. Elgass, H. Krcmar, and A. Oberweis, "Von der informalen zur formalen Geschäftsprozeßmodellierung," *Geschäftsprozeßmodellierung und Workflow-Management. Modelle, Methoden und Werkzeuge. Bonn, Albany: Internat. Thomson Publ*, pp. 125–139, 1996.

[15] S. White, "Introduction to BPMN," *IBM Cooperation*, 2004.

[16] OMG, "BPMN 1.2 - Final Adopted Specification," 2009.

[17] ——, "Unified Modeling Language, Superstructure 2.2," 2009.

[18] C. Schlenoff, M. Gruninger, T. Creek, M. Ciocoiu, and J. Lee, "The Essence of the Process Specification Language," *Transactions of the Society for Computer Simulation*, vol. 16, pp. 204–16, 1999.

[19] B. List and B. Korherr, "An evaluation of conceptual business process modelling languages," in *SAC '06: Proceedings of the 2006 ACM symposium on Applied computing*. New York, NY, USA: ACM, 2006, pp. 1532–1539.

[20] M. Dean and G. Schreiber, "OWL Web Ontology Language Reference," W3C, W3C Recommendation, 2004.

[21] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean, "SWRL: A Semantic Web Rule Language Combining OWL and RuleML," World Wide Web Consortium, W3C Member Submission, 2004. [Online]. Available: http://www.w3.org/Submission/SWRL

[22] A. Seaborne and E. Prud'hommeaux, "SPARQL Query Language for RDF," *W3C Recommendation*, 2008.

[23] C. Fillies and F. Weichhardt, "On Ontology-based Event-driven Process Chains," in *GI-Workshop EPK*, 2005.

[24] B. Heinrich, M. Bewernik, M. Henneberger, A. Krammer, and F. Lautenbacher, "SEMPA–Ein Ansatz des Semantischen Prozessmanagements zur Planung von Prozessmodellen," *Wirtschaftsinformatik*, vol. 50, no. 6, pp. 445–460, 2008.