

MAPPING BUSINESS PROCESS MODELS INTO SOFTWARE DESIGN DIAGRAMS

DOI: 10.5937/EMC26291R

Branislava Radu

University of Novi Sad, Technical Faculty “Mihajlo Pupin“, Zrenjanin, Republic of Serbia

E-mail: branislava.radu@tfzr.rs

Ljubica Kazi

University of Novi Sad, Technical Faculty “Mihajlo Pupin“, Zrenjanin, Republic of Serbia

ABSTRACT

In contemporary information system development, mapping business processes into software architecture represents a key step in ensuring alignment between business requirements and technical implementation. Differences in notations and abstraction levels between business models, such as BPMN and technical models, such as UML and service-oriented architecture, often lead to communication issues, loss of semantics and difficulties in system realization. This paper analyzes various approaches to transforming business processes into architectural and software models with a particular focus on model-driven methods, service-oriented architecture and agile approaches.

Key words: Business processes, Mapping, Software architecture, UML

INTRODUCTION

In contemporary information systems, one of the key challenges is aligning business processes with software architecture. Organizations are increasingly striving for digitalization and automation of their operations, which requires the precise translation of business requirements into technical solutions. However, differences between business models and IT models, both in terms of notation and level of abstraction, often lead to communication issues between business analysts and software engineers, as well as to loss of information during system implementation. (Elvesæter et al., 2010).

Business processes represent a set of interconnected activities with a clearly defined flow, inputs and outputs and their modeling enables a better understanding of organizational functioning and improvement of operational efficiency (Geambaşu, 2012). Effective management of these models helps organizations maintain competitiveness and continuously enhance their performance. Regular review and updating of processes ensure that the products and services delivered meet quality standards and user requirements (Aldin & De Cesare, 2009). Business Process Model and Notation (BPMN) is primarily intended for modeling business processes for business users, while UML diagrams are more technically oriented and intended for software developers. (Peixoto et al., 2008)

The aim of this paper is to demonstrate how business processes can be mapped into UML diagrams and software architectures. Special attention is given to software Service-Oriented Architecture (SOA), which divides functionalities into independent services connected within business process flows. This approach facilitates flexible implementation and service reuse, as well as better planning and management of software development. (Valipour et al., 2009) The main objective of these methods is to ensure continuity between the business and technical levels, reduce the possibility of errors and enable more efficient realization of information systems. A review of relevant literature provides a systematic insight into the mapping problem, highlights key practical challenges and points to possible improvements.

RESEARCH METHODOLOGY

For this research, a literature review was conducted with the aim of identifying and analyzing approaches to mapping business processes into software architecture and UML diagrams. The search was performed on Google Scholar using keywords such as: “business process model mapping to UML” and “business process model mapping to software architecture.”

Through the literature analysis, a large number of studies were identified, from which 10 studies were selected as relevant for examining methods, rules and tools for transforming business processes based on their titles and abstracts. The selected studies were thoroughly reviewed and analyzed and their approaches were grouped into two main categories: the first relates to mapping business processes to software architecture, while the second covers mapping business processes into UML. This analysis provides insight into the different approaches and solutions explored in the literature.

MAPPING BUSINESS PROCESSES TO ARCHITECTURE

This section analyzes different approaches to mapping business process models to software architectures.

One of the key challenges in Service-Oriented Architectures (SOA) is aligning business and IT models, as business analysts often use Business Process Model and Notation (BPMN) to model business processes, while IT architects apply the Service-Oriented Architecture Modeling Language (SoaML) to describe service architectures. Different notations can hinder communication and system implementation. To overcome this issue, Elvesæter et al. (2010) propose a Model Driven Engineering (MDE) approach with multiple levels of abstraction. In this approach, a BPMN business process model is transformed into a logical SoaML service architecture and then into a concrete implementation. Tasks are translated into UML actions, subprocesses into the service architecture, participants and roles from pools and lanes and message exchanges into service ports, while service contracts are established through interaction analysis. Mapping is carried out using the CIMFlex Editor and Atlas Transformation Language (ATL), which allow automatic conversion from BPMN XML models to SoaML models, preserving continuity between business and technical levels. A practical example of a purchase process shows that activities become service logic and participants become service actors, improving alignment between the business and IT layers. Future work includes enhancing transformation rules and supporting BPMN 2.0.

Similarly, another study associates IT project failures with starting development from technology, without prior analysis of the business context. Instead, a reverse approach is proposed, where the design begins with the business model, which is then translated into system architecture. Meertens et al. (2012) use the Business Model Canvas (BMC) in combination with ArchiMate, supported by Osterwalder’s Business Model Ontology (BMO) consisting of nine elements: value proposition, customers, channels, relationships, revenue, resources, activities, partnerships and costs, organized into four domains—product, customers, infrastructure and finances. Mapping connects BMC elements with ArchiMate elements (customer segments → actor/role, channels → interface, relationships → interaction, activities → process/function, resources → resource, partnerships → collaboration and revenue and costs → value) and their relationships, describing cooperation, dependencies and contributions to value creation. The value proposition, as the most complex element, is modeled as a combination of product, goal and customer value. The ArchiSurance case study demonstrates how the BMC model of an insurance company can be transformed into ArchiMate models, turning abstract concepts into processes, actors and resources, while addressing challenges such as overload and redundancy. This approach enables better system understanding, requirements tracking and change analysis, with potential for future automation and quantitative cost–benefit analysis.

Weiss (2010) focuses on the gap between business processes and IT implementation, emphasizing that different models complicate process realization in software. He proposes a method connecting Business Process Modeling (BPM) and services to facilitate the transition from business to technical

levels within SOA. The main contribution is a transformation method that semi-automatically converts BPMN diagrams into UML service models, including component, sequence and activity diagrams, enabling processes to be directly realized as service orchestrations. The BPMN diagram is formalized into a structure containing activities, events, gateway elements, control flows, messages and data with processes broken down into basic patterns—sequential, parallel, branching and loops—with transformation rules defined through an iterative Fold function. In the service model, primitive services represent core functionalities, while composite services orchestrate multiple services and processes, each functioning as a UML component with well-defined interfaces, behaviors and communication protocols. Transformation rules determine how BPMN elements become primitive or composite services across three aspects: structure, behavior and communication.

In the context of collaborative environments, Touzi et al. (2008) explore how business requirements from multiple organizations can be aligned with technical architectures using a Model-Driven Approach (MDA). Business requirements are first modeled at the Computation Independent Model (CIM) level and then transformed into a logical SOA model at the Platform Independent Model (PIM) level. Transformation occurs in two phases: first, the basic SOA elements are created through direct mapping from the BPMN model with pools and lanes represented as services and tasks linked to services and their activities. Data is transformed into business objects in the information view and the process structure organizes elements into three packages: services, information and process. The second phase involves linking rules, transforming the sequence flow into activity order and establishing connections between services, activities and data to preserve process logic. The implemented prototype uses Intalio Designer, Atlas Transformation Language (ATL) and TOPCASED to automate the transformation and visualize UML models.

Finally, applying business process modeling in agile software development shows that the traditional approach using user stories often loses connection with actual business activities. Kazi and Kazi (2025) propose the BPriS method, which converts a BPM model into a prioritized list of work items for sprint planning. The transformation goes through several stages: process decomposition, mapping to software functions, adding business objects, applying functional patterns, semantic grouping and prioritization based on value and dependencies. The result is a backlog that better connects business activities with IT functionalities, facilitates iteration planning and semantically organizes functions, although the method requires additional analysis as it is not fully automated.

MAPPING BUSINESS PROCESSES TO DIFFERENT SOFTWARE DESIGN - RELATED DIAGRAMS

This section analyzes different approaches to mapping business process models into software design diagrams, including use case diagram, class diagram, sequence diagram and activity diagram.

The development of methodologies for automatically translating business processes from BPMN notation into UML activity models represents a significant step toward better integration of business requirements and software design. Cibrán (2008) highlights the problem of BPMN models being poorly understood by IT architects, which hinders the direct connection between business processes and software models. As a solution, a Model-Driven Engineering (MDE) approach is proposed, where the BPMN diagram is used as input and the UML activity diagram as output with defined mapping rules implemented in the Atlas Transformation Language (ATL). Transformations include different types of mappings: one-to-one (Task → Action, Sub-process → StructuredActivityNode, Sequence Flow → Control Flow, Pool/Lane → ActivityPartition), one-to-many (message start event → InitialNode + AcceptCallAction), and many-to-one (multiple events → FlowFinalNode). Complex cases such as XOR and AND gateways are transformed into UML nodes—DecisionNode, MergeNode, Fork and Join—according to context. Automation ATL reduces manual translation and the risk of errors, enabling precise and consistent mapping of BPMN elements to UML, although some semantics may be lost due to UML's limited expressiveness compared to BPMN.

A similar problem is explored by Khelif et al. (2018), who propose the DESTINY methodology based on an MDA approach, allowing automatic generation of UML analysis models directly from BPMN diagrams at the CIM level. The method includes creating use case, sequence and class diagrams with transformation occurring in three phases: preparing the BPMN model by adding semantic context, defining mapping rules for elements to UML and automatically applying these rules. In this process, pools and lanes are translated into actors or classes, tasks become use cases or operations, data objects are transformed into UML classes and gateway elements become control structures in sequence diagrams. The process includes 1:n mapping, where a single BPMN element can generate multiple UML elements, while a pattern-based approach recognizes interaction patterns for automatic creation of use cases. Activity fragments are transformed into use cases and associations are established according to lanes and actors with defined rules for activities and gateways. This results in a complete, semantically rich UML analysis model, directly linked to business processes, allowing a more precise and realistic representation of the information system compared to methods focusing solely on structure.

Macek and Richta (2009) extend the transformation perspective by applying XML and XSLT standards to convert BPMN diagrams into UML activity diagrams (UML-AD), achieving platform independence and improving readability for users and analysts. The goal of the transformation is to preserve information about activity flows, actors and task names, enable reuse of existing BPMN diagrams in software development and improve communication with end users, as BPMN symbols often combine multiple pieces of information into a single element, making understanding difficult. UML-AD, on the other hand, provides clearer presentation and better tool support. The method is based on XML Metadata Interchange (XMI) representation of BPMN diagrams and XSLT transformation into UML-AD XMI with the input XMI describing the BPMN diagram and the output XMI generating the corresponding UML-AD diagram. Start events are translated into InitialNode, while loops and complex symbols are decomposed into multiple UML nodes—task, decision, edges. Due to the complexity of BPMN elements, the transformation often uses one-to-many mapping, which allows precise preservation of activity flows, control and conditions with improved readability, consistency and support for further use of diagrams in software development.

Rhazali et al. (2014) further develop the business process transformation approach using the MDA methodology, enabling the transition from the Computation Independent Model (CIM) to the Platform Independent Model (PIM) level, including the generation of UML use case and class diagrams. The MDA approach separates business description, software model and technical implementation, aiming for BPMN models to serve not only for process understanding but also as a foundation for automatic software development through clearly defined transformation rules. The process occurs in two steps: first, a simple BPMN model is created with clearly defined actors, subprocesses, tasks, activity flows and data; then, transformation rules are applied. Tasks are translated into use cases, actors remain actors, subprocesses are grouped into packages, execution order becomes include, conditions are transformed into extend and data is converted into UML classes. This approach enables a systematic and automated transition from CIM to PIM and precise generation of UML software models.

Previous studies mostly focused on static transformations of BPMN diagrams into UML models. Rabhi et al. (2022) build on these approaches but introduce agile principles and an iterative transformation method. Their work focuses on automatic and iterative transformation of business processes, where BPMN collaboration and process diagrams, identifying participants, subprocesses, tasks, decisions and output data, become UML use case diagrams through sprints. Activity flows are transformed into include relationships, decisions into extend relationships and participants into actors, enabling a flexible and controlled approach to complex processes. An e-commerce process case study demonstrates practical application, where subprocesses and tasks are gradually transferred to the PIM level through iterations. The iterative approach and defined transformation rules enable efficient automation and precise mapping of the business model into software design with future work envisaging transformation of PIM models into PSM and code generation for specific platforms.

CONCLUSION

The aim of this study is to explore methodologies for mapping business processes from BPMN models into UML diagrams and software architecture models. Special emphasis is placed on identifying challenges arising from differences in notations and levels of abstraction between business and IT models.

The analysis of the reviewed studies clearly shows that the problem of aligning business processes with IT architecture is multi-layered and requires a systematic approach. The main challenge lies in the differences between the notations used by business analysts and IT professionals: BPMN enables intuitive modeling of activity flows and interactions, while SoaML and UML are used for technical implementation of services and software models. Differences in complexity and semantics often lead to information loss or inaccurate transfer of business rules, while one-to-one, one-to-many, or many-to-one mappings can result in semantic loss and require additional decomposition of complex elements. The formalization of transformation rules and the application of MDE/MDA approaches, together with pattern-based methods, enable semi-automated transformation and pattern recognition, contributing to better consistency between the business and technical layers, although full automation still requires additional analysis and evaluation. Mapping business elements to technical models allows abstract business concepts to become concrete technical models with clearly defined relationships, but also highlights the problem of redundancy and model overload, suggesting the need for optimization and automated evaluation of transformations. At the same time, iterative approaches through sprints and prioritized backlogs enable flexible and controlled implementation, where the transformation of business activities directly influences the development of software functionalities, reducing the gap between the business and IT layers and allowing efficient iteration planning.

Future directions include improving automated mapping tools to minimize redundancy and reduce the risk of semantic loss, as well as introducing formal verification and defining quality metrics for transformations to ensure the accuracy and reliability of the resulting models. Special attention should be given to semantic alignment of terminology between the business process and the software solution, as well as to the integration of different software components within the execution of a single business process. Additional challenges relate to the scalability of solutions, especially for larger and more complex systems, where increased complexity and model overload make understanding, maintaining and managing transformations more difficult. Another significant challenge is ensuring the flexibility of proposed software solutions so that systems can adapt to technological changes and meet new organizational requirements. These development directions contribute to more efficient implementation, reduced errors and better alignment between the business and IT layers in modern information systems.

REFERENCES

- Aldin, L., & De Cesare, S. (2009). A comparative analysis of business process modelling techniques. In *Proceedings of the 14th UK Academy for Information Systems (UKAIS) Annual Conference* (pp. 1–17), Oxford, United Kingdom.
- Cibrán, M. A. (2008). *Translating BPMN models into UML activities*. Thales Research & Technology, ThereSIS Security & Information Systems.
- Elvesæter, B., Panfilenko, D., Jacobi, S., & Hahn, C. (2010). Aligning business and IT models in service-oriented architectures using BPMN and SoaML. In *Proceedings of the First International Workshop on Model-Driven Interoperability* (pp. 61–68).
- Geambaşu, C. V. (2012). BPMN vs. UML activity diagram for business process modeling. *Journal of Accounting and Management Information Systems*, 11(4), 637–651.
- Kazi, L., & Kazi, Z. (2025). BPriS: Disciplined agile delivery planning method based on work items list pattern applied to prioritized semantically coupled software functions derived from business process model and software functional pattern. *Applied Sciences*, 15(9), 5091.

- Khelif, W., Ayed, N. E. B., & Ben-Abdallah, H. (2018). From a BPMN model to an aligned UML analysis model. In *Proceedings of the 13th International Conference on Software Technologies (ICSOFT 2018)* (pp. 657–665).
- Macek, O., & Richta, K. (2009). The BPM to UML activity diagram transformation using XSLT. In *Dateso 2009 Proceedings* (pp. 119–129).
- Meertens, L. O., Iacob, M. E., Nieuwenhuis, L. J., van Sinderen, M. J., Jonkers, H., & Quartel, D. (2012). Mapping the business model canvas to ArchiMate. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing* (pp. 1694–1701).
- Peixoto, D. C., Batista, V. A., Atayde, A. P., Borges, E. P., Resende, R. F., & Pádua, C. I. P. (2008). A comparison of BPMN and UML 2.0 activity diagrams. In *Proceedings of the Simpósio Brasileiro de Qualidade de Software (SBQS 2008)* (pp. 1–12).
- Rabhi, O., Arrassen, I., & Erramdani, M. (2022). Automatic agile transformation with approach by modeling from BPMN collaboration diagram model to UML use case. In *Proceedings of the International Conference on Industrial Engineering and Operations Management (IEOM 2022), Istanbul, Turkey*.
- Rhazali, Y., Hadi, Y., & Mouloudi, A. (2014). Transformation method CIM to PIM: From business processes models defined in BPMN to use case and class models defined in UML. In *International Conference on Multimedia Computing and Systems (ICMCS)*.
- Touzi, J., Bénaben, F., & Pingaud, H. (2008). Prototype to support morphism between BPMN collaborative process model and collaborative SOA architecture model. In B. Vallespir, T. Alix, & D. Romero (Eds.), *Enterprise Interoperability III: New Challenges and Industrial Approaches* (pp. 145–157). Springer.
- Valipour, M. H., AmirZafari, B., Maleki, K. N., & Daneshpour, N. (2009). A brief survey of software architecture concepts and service-oriented architecture. In *2009 2nd IEEE International Conference on Computer Science and Information Technology* (pp. 34–38). IEEE.
- von Rosing, M., White, S., Cummins, F., & de Man, H. (2015). *Business process model and notation (BPMN)*. Morgan Kaufmann.
- Weiss, P. (2010). Modeling of service-oriented architecture: Integration of business process and service modeling. *Information Sciences and Technologies Bulletin of the ACM Slovakia*, 2(2), 79–92.