Comparison of Selected Modeling Notations for Process, Decision and System Modeling

Krzysztof Kluza, Piotr Wiśniewski, Krystian Jobczyk, Antoni Ligęza
AGH University of Science and Technology
ul. A. Mickiewicza 30, 30-059 Krakow, Poland
E-mail: {kluza,wpiotr,jobczyk,ligieza}@agh.edu.pl

Anna Suchenia (Mroczek)
Cracow University of Technology
ul. Warszawska 24, 31-155 Krakow, Poland
Email: asuchenia@pk.edu.pl

Abstract—System specifications can be modeled using various types of notations and diagrams regarding applications of the particular model. In this paper, we present an overview of the existing solutions, focusing on UML, BPMN and DMN models and the diagrams provided by these notations. We perform a comparison of these approaches and provide examples of representing system requirements in these notations.

Index Terms—Software Engineering, UML, BPMN, DMN, Unified Modeling Language, Business Process Model and Notation, Decision Model and Notation

I. INTRODUCTION

SOFTWARE engineering aims to produce effectively good quality software. Various methods and processes are at the heart of software engineering [1]. In practical software design, parts of systems are specified using visual models. The standard for modeling software applications is Unified Modeling Language (UML). It provides diagrams to capture requirements, collaboration between parts of the software that realize them, the realization itself and models which show how everything fits together and is executed [2].

Business Process Management [3], in turn, is a modern approach to improving organization’s workflow, focused on reengineering of processes to obtain optimization of procedures, increase efficiency and effectiveness by constant process improvement. Business Process Model and Notation (BPMN) is the standard for designing business process models. BPMN can get along with with UML [4], but it does not support modeling of some concepts such as rules. Decision Model and Notation (DMN) provides a standard for modeling decisions and supports decision management and business rules.

The rest of this paper is organized as follows: In Sections II-IV, UML, BPMN, and DMN are introduced. Section V presents the comparison of the notations with the focus on the comparisons from the 4+1 view model architecture perspective. Contributions of the paper are summarized in Section VI.

II. UNIFIED MODELING LANGUAGE (UML)

Unified Modeling Language (UML) is a general-purpose modeling language in the field of software engineering. Modeling is about capturing a system as models [2], which can be depicted as sets of diagrams. Such diagrams describe the system (or a part of it). UML 2 defines a variety of diagrams divided into two main categories: structure diagrams – containing diagrams representing the structure of a modeled application, and behavior diagrams – contain diagrams representing general types of behavior. A tree showing the classification of the UML diagram types [5] is presented in Fig. 1.

The complete system can be described by a number of models describing the system from different angles, often on various levels of abstraction. By design, each UML diagram should be consistent with any other diagram representing the same model. But inconsistency is highly likely to occur in models. Some issues can be resolved using formal methods [6]–[8] or ontologies [9]–[11], but there are also other modeling problems such as exceptions [12] or using reverse engineered models [13].

UML itself is not a design method or a software process. It is only a notation which can be useful within a software process or designing. Another issue is a methodology which indicates how to apply a design. UML itself does not require any specific method, but mostly it is used with an object-oriented design method.

III. BUSINESS PROCESS MODEL AND NOTATION (BPMN)

Business Process Model and Notation (BPMN) [14] is the most widely used notation for modeling business processes. As the notation is quite complex, it has many application areas that may be found in [15]–[20].

The current BPMN 2.0 specification [21] provides four different types of diagrams:
1) Process diagram (describing the ways in which operations are carried out to accomplish the intended objectives of an organization),
2) Collaboration diagram (presenting the collaborative public Business 2 Business process),
3) Conversation diagram (which specifies the logical relation of message exchanges),
4) Choreography diagram (defining the expected behavior between two or more interacting business participants in the process).

In most cases, using only the process model is sufficient. The process model uses four basic categories of elements to model BPs: flow objects (activities, gateways, and events), connecting objects (sequence flows, message flows, and associations), swimlanes, and artifacts as shown in Figure 2.

**Figure 2: BPMN core elements of Process diagram**

In the case of flow object elements, activities denote tasks that have to be performed, events indicate something that happens during the lifetime of the process, and gateways determine forking and merging of the sequence flow between tasks in a process, depending on some conditions. The sequence flow between flow objects is used to model the flow of control in a process. The message flow between selected elements is used to model the flow of messages between participants of a process (which are depicted as different pools).

BPMN 2.0 defines more than 100 elements, thus practitioners differentiate them based on the degree of model detail. Three levels of models can be distinguished: a descriptive level, which is the basic level that uses a very intuitive subset of BPMN to reflect a "happy path" scenario and all major activities in a process; an analytical level, dedicated to analysts, modelers and business architects that use complex structures and elements to design fully representative processes, and an executable level for technicians in which execution details can be captured in the model. Additionally, many different extensions of BPMN were proposed to capture other aspects of business processes [23]–[28].

**IV. DECISION MODEL AND NOTATION**

DMN [29] is a brand new OMG standard for decision modeling. Such a decision determines the result (or selects some option) based on some input data. Its goal is to provide the notation for decision modeling so the decision can be easily presented in diagrams and understandable by business users [30]. The main purposes of the notation are: modeling human decision-making, modeling the requirements for automated decision-making, and implementing automated decision-making [29]. Such decision models can be integrated with BPMN models or exist separately [31].

There are four types of core elements in DMN: Decision, Business Knowledge Model, Input Data, and Knowledge Source (see Fig. 3). Decision elements are used to determine an output from a number of inputs using some decision logic. Business Knowledge Model elements denote functions encapsulating business knowledge (like decision table, business rules or analytic models). Input Data elements are used for modeling the input of a Decision or Business Knowledge Model when values are defined outside of the decision model. Knowledge Source elements model authoritative knowledge sources in a decision model. These elements can be connected using different requirement connectors. There are three different types of them: Information, Knowledge, and Authority.

**Figure 3: The types of DMN elements**

The decision model is usually represented as Decision Requirements Graph (DRG). DRG can be split into one or more Decision Requirements Diagrams (DRD) presenting a particular view of the model [29].

DMN provides a wide range of tools (various types of decision logic representation, Elements, and Requirements) to implement decision-making, automated or not. It can be easily adjusted and understood. It fills the gap in the market of decision modeling and is often used with BPMN.

**V. COMPARISON OF UML, BPMN AND DMN**

We compare the diagrams of the UML, BPMN and DMN notations using the evaluation Framework for BPM/ISM technique [33] and comparing the diagrams in terms of system specification views, especially focusing on the “4+1” view model architecture [32].

Software architecture deals with abstraction, with composition and decomposition. To describe such architecture, a "4+1" model is often used. The model was designed by Philippe Kruchten and used for "describing the architecture of software-intensive systems, based on the use of multiple, concurrent views" [32]. The views are used to describe the system from the viewpoint of different users (end-users, developers and project managers) [32], [34]. The "4+1" view model supports five main views, as shown in Figure 4 and in Table:

1. **Logical View** – an object model of the design.
2. **Process View** – concurrency and synchronization aspects.
3. **Development View** – static organization of the software.
4. **Physical View** – mapping of the software to the hardware.
+1 **Use-cases view** – various usage scenarios.
The Logical View (Object-oriented Decomposition) and the Process View are at a conceptual level and are used from analysis to design [35]. This view focuses on realizing an application’s functionality in terms of structural element, key abstractions and mechanisms, distribution of responsibilities and separation of concerns. Users-architects use this view for functional analysis [35]. The Process View (process decomposition) [36] captures the concurrency and synchronization aspects of the design. Development View describes the static organization of the software in its development environment [35]. The Physical view (mapping software to hardware) describes the mapping(s) of the software onto the hardware and reflects its distributed aspect [36]. Use case view presents functionality of the system, its external interfaces, and principal users of the system.

![Diagram](image.png)

Figure 4: The "4+1" view model architecture

Similarly, the evaluation framework for BPM/ISM [33] is not intended to be rigid, as the lines between depth and breadth of modeling are blurred and hard to be separated. Our evaluation of the UML, BPMN, and DMN notations in terms of the Giaglis evaluation framework is presented in Table II. Table III presents the comparison of the diagrams in these notations, especially focusing on their application in "4+1" view model architecture [32].

VI. SUMMARY

This paper has given an overview and provided a comparison of the most popular notations for modeling systems, processes, and decisions, i.e. the UML, BPMN and DMN notations. These results have been presented in terms of the "4+1" view model architecture. Further specification of the contribution is going to be a subject of future research to find the appropriate modeling methods for particular systems.

<table>
<thead>
<tr>
<th>View</th>
<th>Detail</th>
<th>Stakeholders</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logical</td>
<td>Subsystems Classes</td>
<td>End Users</td>
<td>Functionality</td>
</tr>
<tr>
<td>Implementation</td>
<td>Components, Packaging</td>
<td>Developer, Project, Manager</td>
<td>Used to be called Development View</td>
</tr>
<tr>
<td>Deployment</td>
<td>Topology, Mapping to Platforms</td>
<td>System Engineer</td>
<td>Used to be called, Physical View</td>
</tr>
<tr>
<td>Process</td>
<td>Performance, Throughput, Concurrency</td>
<td>System Integrator</td>
<td>It is a Computer Engineering term</td>
</tr>
<tr>
<td>Use case</td>
<td>Architecture, Discovery, View Validation</td>
<td>Analyst, Tester</td>
<td>Sometimes called Scenarios</td>
</tr>
</tbody>
</table>

REFERENCES

Table II: Comparison of selected modeling approaches

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Functional</td>
<td>(UML) DMN</td>
<td>BPMN DMN</td>
<td>UML DMN</td>
<td>UML DMN</td>
<td></td>
</tr>
<tr>
<td>Behavioural</td>
<td>BPMN (DMN)</td>
<td>BPMN (DMN)</td>
<td>BPMN (DMN)</td>
<td>BPMN DMN</td>
<td></td>
</tr>
<tr>
<td>Organizational</td>
<td>(DMN)</td>
<td>(BPMN)</td>
<td>(BPMN)</td>
<td>(UML)</td>
<td>—</td>
</tr>
<tr>
<td>Informational</td>
<td>UML DMN</td>
<td>BPMN DMN</td>
<td>BPMN DMN</td>
<td>BPMN DMN</td>
<td>BPMN DMN</td>
</tr>
</tbody>
</table>

Table III: Comparison of diagrams

<table>
<thead>
<tr>
<th>System specification</th>
<th>Class diagram</th>
<th>UML</th>
<th>BPMN</th>
<th>DMN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Structure</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Behaviour</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Requirements</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Implementation</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>4+1 Logical view</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>4+1 Process view</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>4+1 Development view</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>4+1 Physical view</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>4+1 Scenarios</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

✓ – the diagram supports or is used in the particular view
✓ ✓ – the diagram partially supports or can be used as an additional element in the view