Towards Software Development for Maintainability Evaluation of Business Process Models Using Fuzzy Logic

Andrii Kopp and Dmytro Orlovskyi

National Technical University "Kharkiv Polytechnic Institute", Kyrpychova str. 2, Kharkiv, 61002, Ukraine

Abstract
Business process modeling is a key tool in Business Process Management. Business process models are graphical representations of organizational activities used to depict ongoing tasks, events, and decision-making points to improve enterprise performance by gathering software requirements for workflow automation. Business process models facilitate the design of high-quality enterprise architectures, therefore, the quality of models impacts the success of proper solutions communication and implementation. Hence, this study focuses on the maintainability as one of the crucial quality attributes of business process models. Using the existing metrics of business process model complexity, this paper proposes to automatically detect complexity levels of business process models using the algorithm based on fuzzy logic and the respective software tool implementing the proposed algorithm. The software tool assumes treatment of business process models as the software engineering assets via the version-control system and its usage demonstrates processing of the large repository of business process models.

Keywords [1]
Business Process Modeling, Business Process Model Maintainability, Business Process Model Complexity, Fuzzy Logic, Software Tool

1. Introduction

Business process modeling is the main tool of the Business Process Management (BPM) approach, which is used to describe organizational activities in the form of graphical diagrams.

In general, BPM is seen as a discipline that combines the best practices of Information Technology (IT) and business management to help organizations identify, analyze, design, implement, and control business processes [1].

The capabilities of the BPM approach are used by organizations to deliver high quality products and services to customers and to find ways to improve organizational performance. Business processes are scenarios of activities driven by events and decisions. Business processes are also considered as "chains of events, actions, and decisions" according to [2].

Business process models are business process descriptions, most often graphical in the form of diagrams and charts, that help design and analyze information systems and serve as communication mechanisms between stakeholders [3].

Thus, business process model quality is, no doubt, vital for the successful implementation of BPM projects, or at least for collecting requirements to develop or improve enterprise information systems. Maintainability is one of the quality attributes particularly interesting for facilitating the evolution and execution of operations to support business process models in a highly dynamic environment [4].

The research object of this study is the procedure of maintainability evaluation of business process models. The research subject of this study is the software tool for maintainability evolution of business process models. This study aims to improve the maintainability of business process models by using complexity metrics similar to software metrics.

Information Technology and Implementation (IT&I-2023), November 20-21, 2023, Kyiv, Ukraine
EMAIL: kopp93@gmail.com (A. Kopp); orlovskyi.dm@gmail.com (D. Orlovskyi)
ORCID: 0000-0002-3189-5623 (A. Kopp); 0000-0002-8261-2988 (D. Orlovskyi)

Therefore, this study should answer the following research questions:

- How to assess the maintainability of a business process model using software-like metrics?
- How to improve and automate the evaluation of business process model maintainability?

2. Business Process Modeling

Business process modeling aims to represent organizational activities so that current operations can be analyzed and improved. Business process modeling is not only a requirement for ISO 9001 quality standards or BPM maturity models, but also plays an important role in the implementation of business process management [5].

Business process modeling helps to visualize important stages of a business process, how they relate to each other, which entities and information systems are involved in performing various tasks, and where communication with external parties takes place. Business process models are typically described visually using graphical diagrams that are linked together and supported by text annotations [5].

There are numerous studies in the field of business process model quality [6]:

- SEQUAL Framework;
- The Guidelines of Modeling (GoM);
- Quality Framework for conceptual modeling;
- Seven Process Modeling Guidelines (7PMG);
- Business process model quality metrics etc.

Thus, the quality of business process models can be considered as the degree to which the process model meets the requirements of the process modeling guidelines related to ISO 9001 standards, which declare quality as “the degree to which a set of inherent characteristics meets the requirements” [7].

Several business process modeling notations are used today, among which the most common are EPC (Event-driven Process Chain) and BPMN (Business Process Model and Notation), both graphical modeling languages used to describe business processes. However, the BPMN notation has almost replaced EPC as the most widely used standard for business process modeling due to its growing popularity in recent years [8].

Basically, BPMN business process diagrams describe workflows in terms of events and activities linked through control flows, which indicate the existing sequences in the process. Gateways are special elements connected through control flows that indicate whether branched steps of a business process are performed in parallel (AND), alternatively (XOR), or optionally (OR). The beginning of a business process is marked by a start event, and its completion is marked by one or more end events. Each pool represents a business process, while each lane represents a person involved in the activity [9].

The key elements of the BPMN business process modeling notation are shown in Fig. 1.

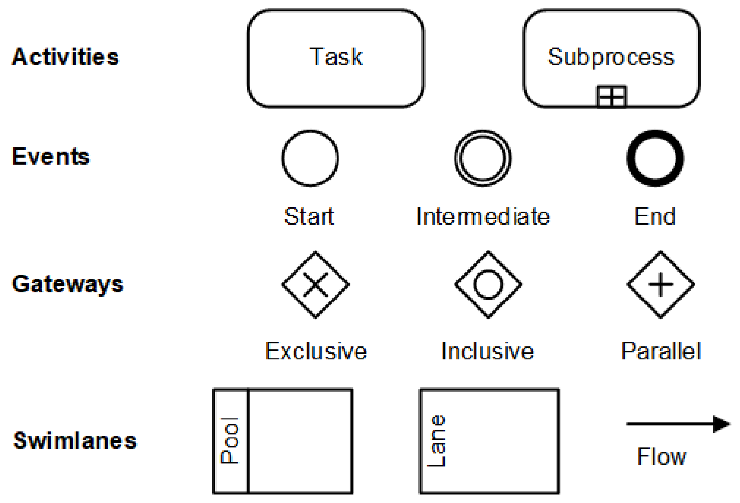


Figure 1: Essential elements of the BPMN graphical notation for business process modeling

3. Business Process Model Maintainability Evaluation

According to [10], the complexity of business process models has an undesirable impact on their maintainability. Therefore, the simpler the model, the easier it is to modify and the less money and time it will take to make changes to the business process [4]. In this paper, we propose to use complexity metrics based on software metrics to evaluate the suitability of BPMN business process models for use.

The business process model in BPMN notation is proposed to be represented as a directed graph [12], where:

$$BPModel = (N, l, A), \tag{1}$$

where:

- $N = T \cup E \cup G$ is the set of nodes representing different business process elements: tasks and sub-processes T, events E, and gateways G;
- $G = S \cup J$ is the subset of gateways, which in turn also consists of splits S and joins J;
- $l: G \to \{and, or, xor\}$ is the mapping that defines the types of AND, OR, and XOR gateways;
- $A \subseteq N \times N$ is the binary relationship that represents the arcs (control flows) of the business process model.

Vanderfeesten et al. introduced several complexity metrics adapted from software engineering [11]. Based on the simple LOC (Lines of Code) metric, which involves counting the number of lines of program code, the authors of [4] proposed the NOA (Number of Activities) metric to count the number of activities in a business process and NOAJS (Number of Activities, Joins, and Splits) to count the number of activities, split gateways, and join gateways in a business process.

Thus, formally, the NOA and NOAJS metrics can be defined based on the following expressions:

$$NOA = |T|, \tag{2}$$

$$NOAJS = |A| + |J| + |S|. \tag{3}$$

The CFC (Control Flow Complexity) metric, proposed by Cardoso [10] to measure the complexity of the control flow within a business process model, allows to assess the complexity of a business process based on the analysis of such gateways as XOR-Split, OR-Split, and AND-Split [4]:

$$CFC = \sum_{s \in S} CFC_{xor}(s) + \sum_{s \in S} CFC_{or}(s) + \sum_{s \in S} CFC_{and}(s), \tag{4}$$

where:

- $CFC_{xor}(s) = fan - out(s)$ is the gateway complexity metric of the XOR-Split type;
- $CFC_{or}(s) = 2^{fan-out(s)} - 1$ is the gateway complexity metric of the OR-Split type;
- $CFC_{and}(s) = 1$ is the gateway complexity metric of the AND-Split type;
- $s \in S$ is the split gateway;
- $fan - out(s)$ is the number of outgoing control flows from the gateway.

According to [10], the CFC metric can be used as a maintainability and quality metric, as it allows to assess the relative complexity of business process models and is quite easy to apply [4].

To assess the duration, scope, and complexity of a process, Vanderfeesten et al. proposed the HPC (Halsted-based Process Complexity) metric [11]. This metric allows predicting the number of errors and maintenance efforts of a business process. It can be used for most process modeling languages (BPMN, EPC, etc.):

$$HPC_N = n_1 \log_2(n_1) + n_2 \log_2(n_2), \tag{5}$$

$$HPC_V = (N_1 + N_2) \log_2(n_1 + n_2), \tag{6}$$

$$HPC_D = \frac{n_1}{2} \frac{N_2}{n_2}, \tag{7}$$

where:

- HPC_N is the process length;
- HPC_V is the process volume;
- HPC_D is the process complexity;
- n_1 is the number of unique elements of the control flow (tasks, gateways, events);

- n_2 is the number of unique variables manipulated by the process and its activities (tasks, gateways, events);
- N_1 is the number of all elements of the control flow (tasks, gateways, events);
- N_2 is the number of all variables manipulated by the process and its activities (tasks, gateways, events).

The IC (Interface Complexity) is another metric proposed by Vanderfeesten et al. [11], which is used to assess the complexity of business processes. The IC value is obtained by calculating the IC value for each activity and then summing it for all activities of the business process:

$$IC = length\left[\sum_{n\in N} fan-in(n) + \sum_{n\in N} fan-out(n)\right]^2, \tag{8}$$

where:

- $length$ is the coefficient equal to 1 for tasks and 3 for sub-processes;
- $n \in N$ is the business process activity;
- $fan-in(n)$ is the number of incoming control flows of the activity;
- $fan-out(n)$ is the number of outgoing control flows of the activity.

Therefore, it is proposed to evaluate business process models based on the CFC metric, which can be used as a metric of maintainability and quality [10].

For this purpose, the following thresholds of the CFC metric and the following linguistic estimates were determined in [13]:

- $CFC \leq 3$ for the low complexity and high maintainability;
- $3 < CFC \leq 9$ for the moderate complexity and moderate maintainability;
- $CFC > 9$ for the high complexity and low maintainability.

The following set describes the levels of maintainability of business process models for the linguistic variable "CFC Level":

$$Term(CFCLevel) = \{low, moderate, high\}. \tag{9}$$

The following trapezoidal membership functions, built using the thresholds of the CFC metric ± 1 to smooth the boundaries of the complexity and maintainability levels [13], describe the corresponding linguistic values of the "CFC Level" linguistic variable:

- low maintainability,

$$\mu_{low}(CFC) = \begin{cases} 1, & CFC \leq 2 \\ \frac{4-CFC}{2}, & 2 < CFC < 4; \\ 0, & CFC \geq 4 \end{cases} \tag{10}$$

- moderate maintainability,

$$\mu_{moderate}(CFC) = \begin{cases} 0, & CFC \leq 2 \\ \dfrac{CFC-2}{2}, & 2 < CFC < 4 \\ 1, & 4 \leq CFC \leq 8 \ ; \\ \dfrac{10-CFC}{2}, & 8 < CFC < 10 \\ 1, & CFC \geq 10 \end{cases} \tag{11}$$

- high maintainability,

$$\mu_{high}(CFC) = \begin{cases} 0, & CFC \leq 8 \\ \dfrac{CFC-8}{2}, & 8 < CFC < 10. \\ 1, & CFC \geq 10 \end{cases} \tag{12}$$

The parameters of the proposed membership functions were determined based on the thresholds of the CFC metric [13].

The universal set of the CFC variable is $[0, +\infty]$.

Syntactic and semantic rules are not defined for the proposed linguistic variable "CFC Level".

Fig. 2 below demonstrates the appearance of the proposed membership functions.

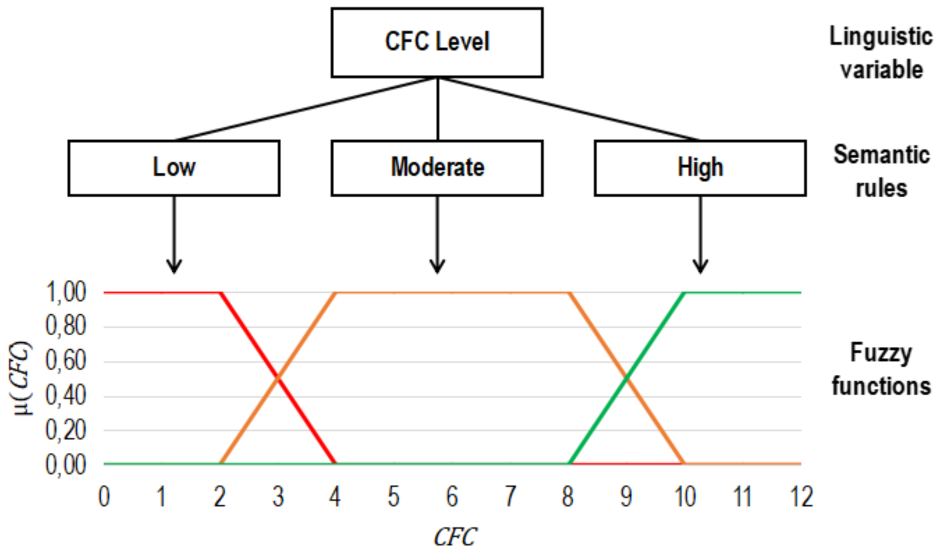


Figure 2: Membership functions of the CFC fuzzy values

Finally, the maximum value of the membership functions allows to evaluate the maintainability level of the business process model:

$$CFCLevel(CFC) = \underset{term \in Term}{\arg\max}\ \mu_{term}(CFC), \tag{13}$$

where $term \in Term$ is the level of maintainability of the business process model.

4. Software Requirements Elicitation

Functional requirements for software define what functions a system (or its individual components) should have. Thus, functional requirements define what the software should do. If these requirements are not met, the software cannot be considered to be designed to meet the needs of the users [14].

The most common examples of functional requirements are [14]:

- use cases;
- specification documents.

Thus, the requirements for a software solution for assessing the maintainability of business process models are presented as the UML (Unified Modeling Language) use case diagram (Fig. 3).

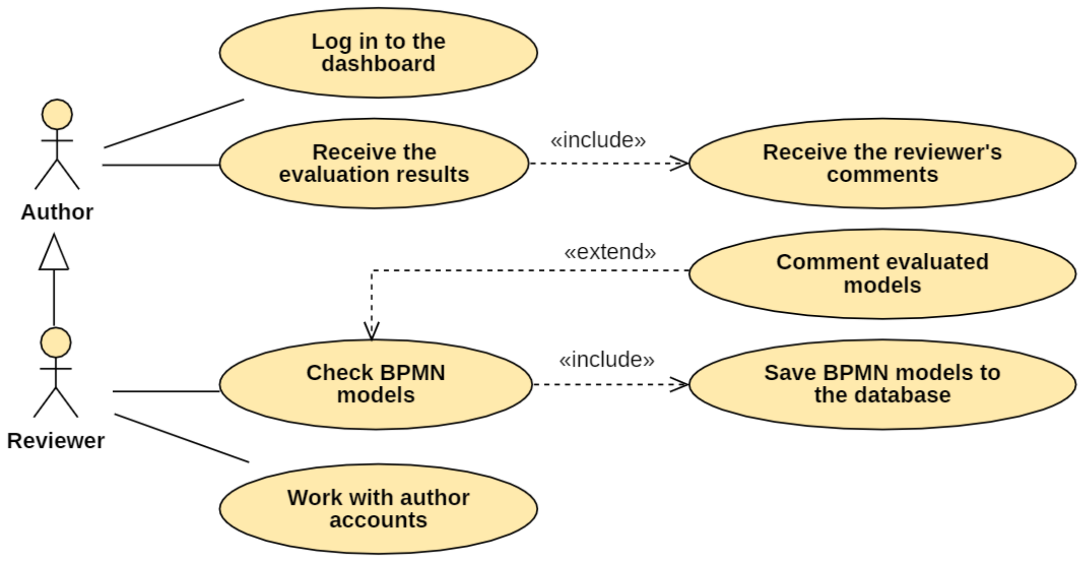

Log in to the dashboard
Author
Receive the evaluation results
«include»
Receive the reviewer's comments
«extend»
Comment evaluated models
Check BPMN models
«include»
Save BPMN models to the database
Reviewer
Work with author accounts

Figure 3: Use case diagram

The formulated Functional Requirements (FR) are presented in Table 1.

Table 1
Functional requirements for the software tool

FR	Role	Requirement
FR1	Reviewer	Work with author accounts so that they can enter the dashboard with the results
FR2	Reviewer	Check BPMN models to get a maintainability assessment and save them to the database
FR3	Reviewer	Comment evaluated models to recommend possible actions to improve their quality
FR4	Author	Log in to the dashboard to access the results of the BPMN model check
FR5	Author	Receive the evaluation results and the reviewer’s comments to make corrections and improve the quality of the BPMN model

Non-functional requirements define how the software fulfills its functional goals (not what the system will do, but how the system will do it). These requirements define the criteria used to evaluate the performance, reliability, and maintainability of the system. They define and impose external constraints on the software to be developed and on the development process [14].
The formulated Non-Functional Requirements (NFR) are presented in Table 2.

Table 2
Non-functional requirements for the software tool

NFR	Characteristics	Requirement
NFR1	Portability	The software must be a cross-platform application
NFR2	Compatibility	The software should work and look the same in different web browsers
NFR3	Compatibility	The software should work with business process models presented in the BPMN 2.0 format
NFR4	Internationalization	The language of the software user interface must be English
NFR5	Security	Passwords to software user accounts must be stored in encrypted form
NFR6	Usability	The software interface should not be overloaded, providing the user with up to 7 controls at the same time

In addition, certain system requirements for the software tool are outlined:

- architecture – the software should be created as a web application based on a three-tier client-server architecture [15];
- data storage – the software should use a relational database management system (DBMS) to implement the database of business process models and a distributed version control system (VCS) Git to implement the workflows for BPMN models similar to other software engineering artifacts;
- technologies – the system should use PostgreSQL DBMS [16], Git VCS [17], Python language [18], Flask [19], React [20] and Bootstrap [21] frameworks, HTML, CSS, and JavaScript.

5. Software Design and Development

Let us demonstrate the key components of the software tool, their purpose and allocation using the UML deployment diagram [22] (Fig. 4).

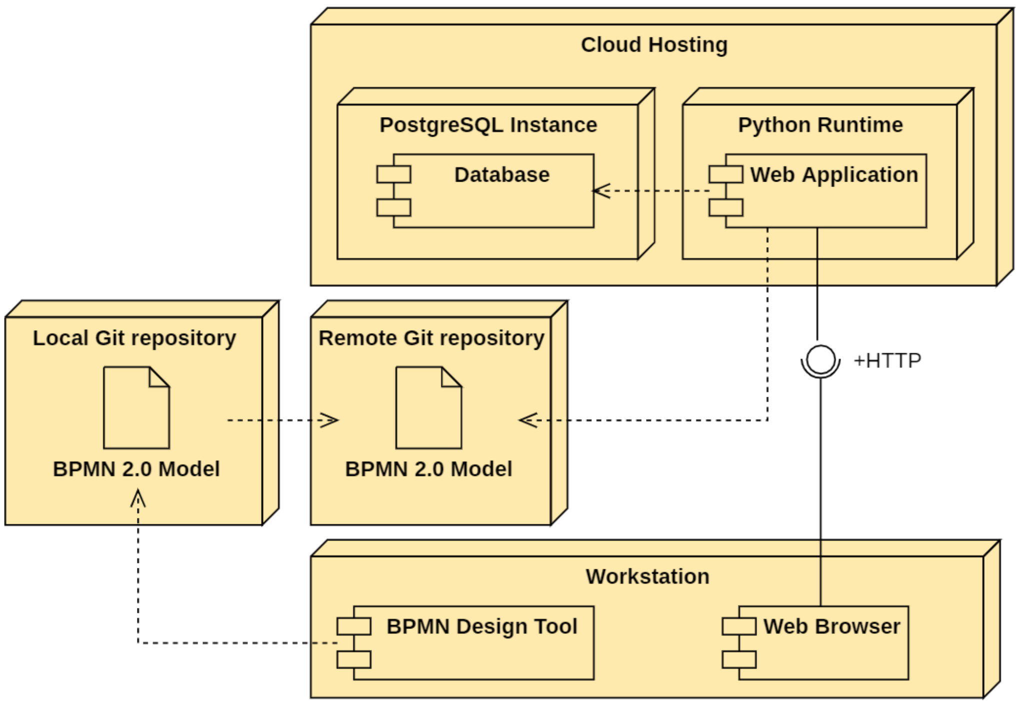


Figure 4: Software tool deployment diagram

According to the deployment diagram (Fig. 4), BPMN models are added to the local Git repository by users, then sent to a remote Git repository, processed by a software solution created in Python, the processing results are stored in a database based on the PostgreSQL database, and then displayed on a web page to the user.

Let us present the internal structure of the software tool for assessing the maintainability of business process models using the UML class diagram [22] (Fig. 5).

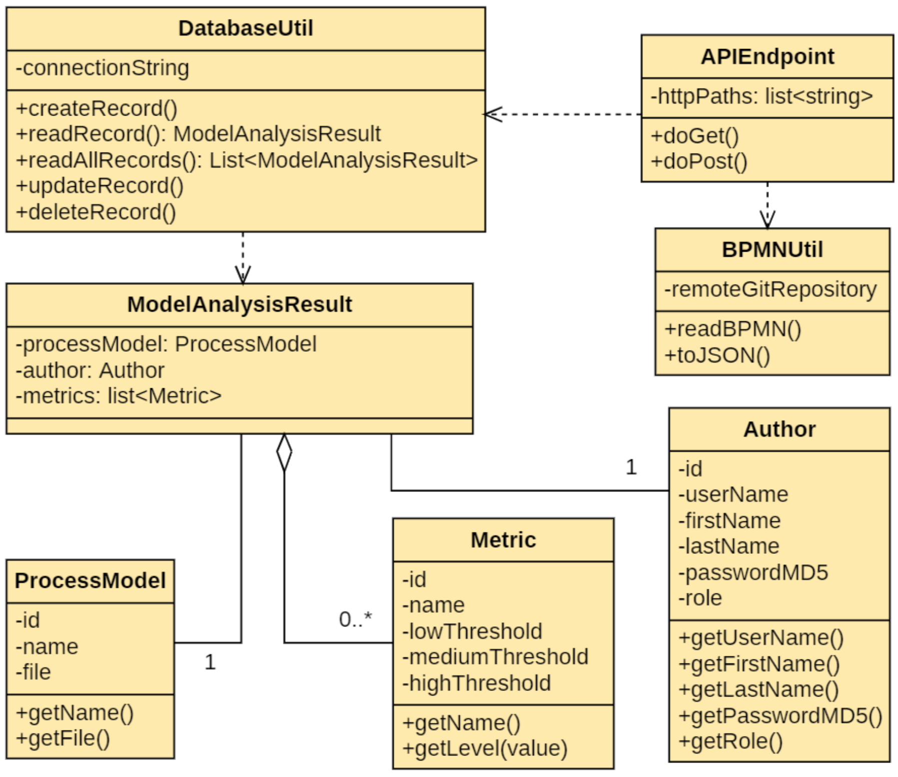


Figure 5: Software tool class diagram

The ProcessModel, Metric, and Author classes provide the object-relational mapping [23] between objects and database entities. In addition, these classes have getter methods for accessing fields.

The ModelAnalysisResult class, shown in the UML diagram (Fig. 5), is an object-oriented mapping of a fact table with information about the business process model, author, and calculated metrics.

The DatabaseUtil class acts as an access point to the database and implements the operations of creating, updating, deleting, and reading data (CRUD).

The BPMNUtil class provides capabilities for accessing a collection of BPMN 2.0 documents, reading files, and converting them to JSON objects.

The APIEndpoint class implements a web application and processes client requests.

The interaction between the software tool components is demonstrated using the UML sequence diagram [22] (Fig. 6).

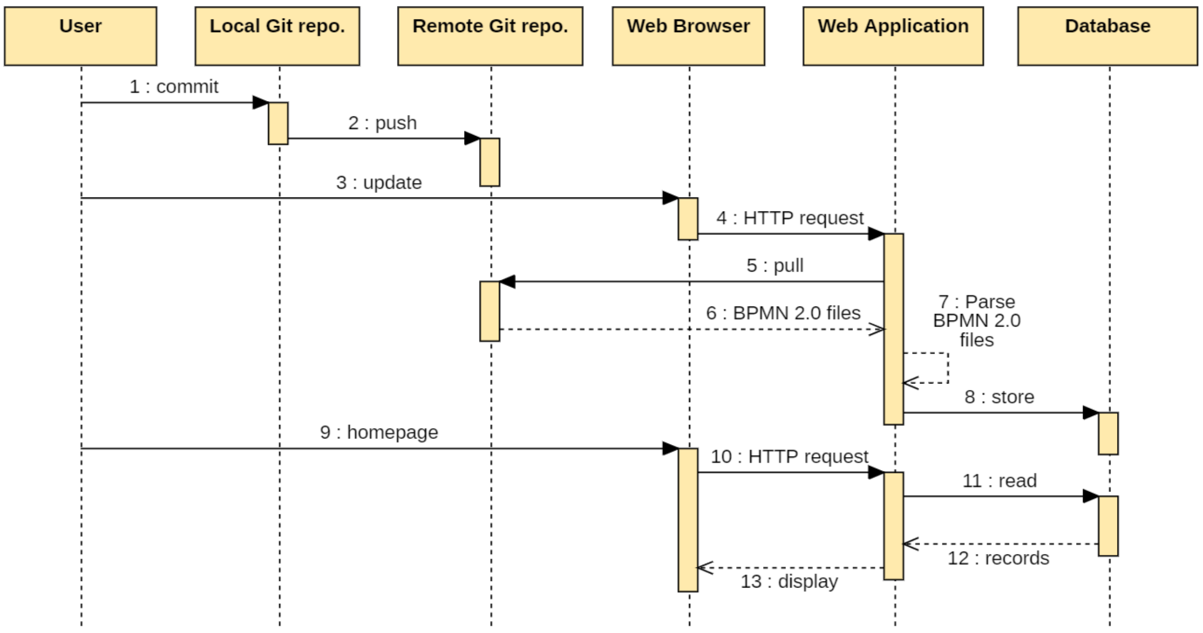


Figure 6: Software tool sequence diagram

Let us consider interactions between the software components depicted in more detail (Fig. 6):

1. The user (the author of BPMN model) performs a commit operation and adds the BPMN 2.0 file to the local Git repository.
2. The user performs a push operation and sends the BPMN 2.0 file to the remote Git repository.
3. The user opens the software in a web browser and starts updating the models.
4. The web browser sends an HTTP (Hyper Text Transfer Protocol) request to the web application to update the business process models.
5. The web application requests access to the BPMN models stored in the remote Git repository.
6. The web application retrieves BPMN 2.0 files from the remote Git repository.
7. The web application processes the BPMN 2.0 files to extract data about the business process structure and makes necessary calculations.
8. The web application saves the results of business process models maintainability analysis for future use in the repository.
9. The user opens the main page of the software solution in a web browser.
10. The web browser sends an HTTP request to the web application to get data from the repository.
11. The web application queries the repository to obtain the results of the business process model maintainability analysis.
12. The web application retrieves the data from the repository.
13. The web application displays the results of the business process model maintainability analysis retrieved from the repository, which the user will view in a web browser on the main page of the software tool.

Thus, the detailed structure of the software tool components designed to evaluate the maintainability of business process models is depicted using the UML component diagram [22] (Fig. 7).

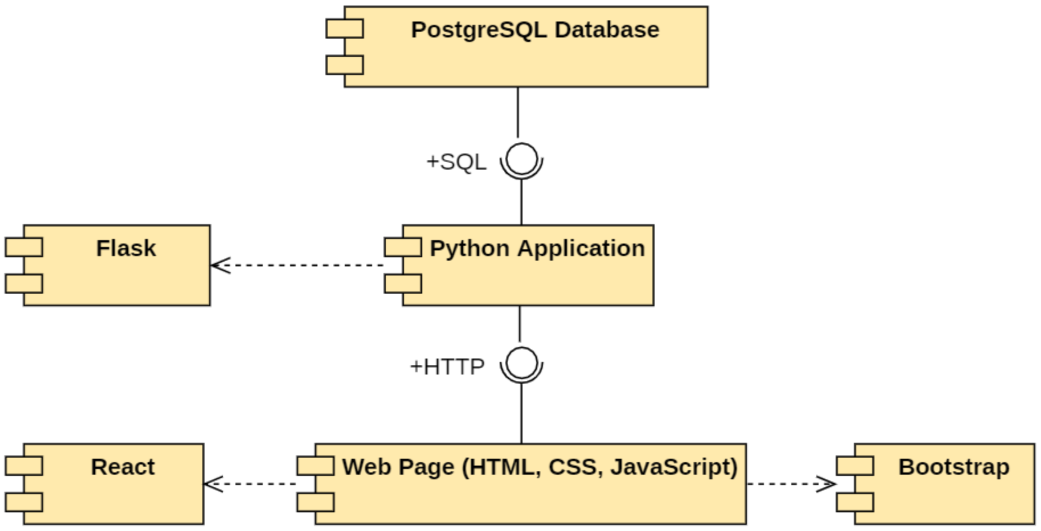


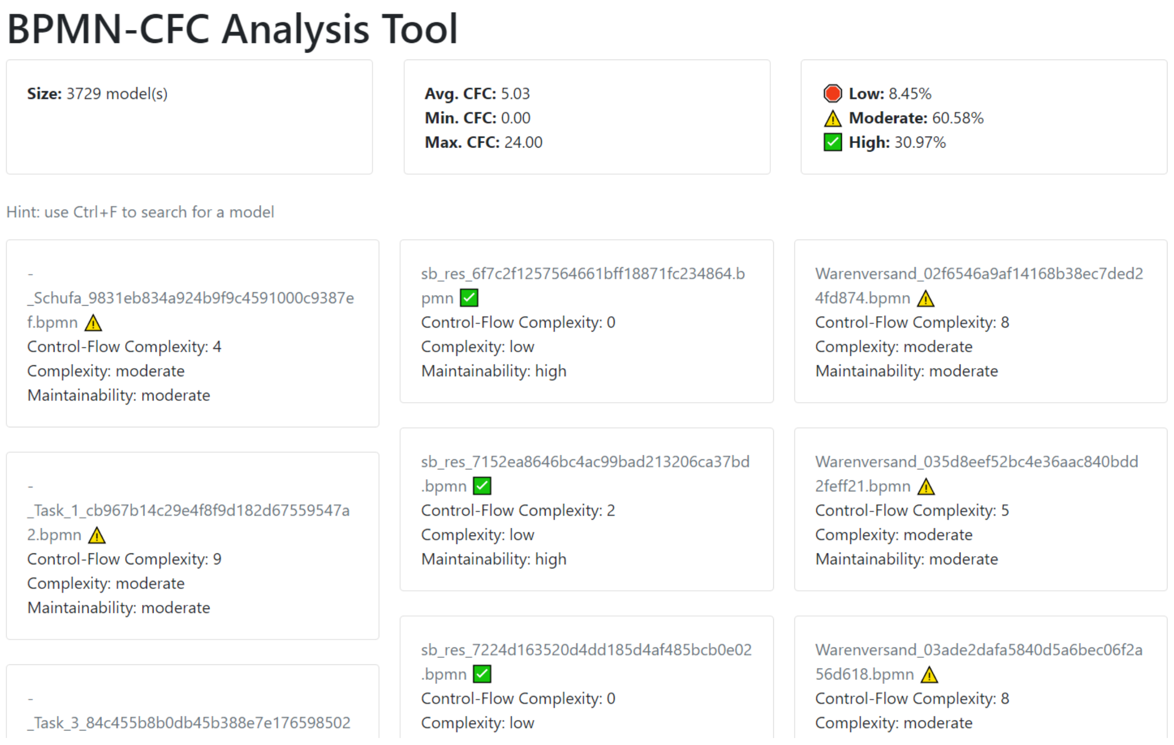
Figure 6: Software tool component diagram

Fig. 6 above demonstrates development tools and software solution components created with their help to assess the maintainability of business process models.

6. Experimental Results

To test the performance of the software tool for assessing the maintainability of business process models, which can be used on practice to refine described workflows, we used the GitHub repository "bpmn-for-research", which contains 3729 business process models in BPMN 2.0 notation [24].

The main page of the developed software tool is shown in Fig. 7.



BPMN-CFC Analysis Tool

Size: 3729 model(s)

Avg. CFC: 5.03
Min. CFC: 0.00
Max. CFC: 24.00

🔴 **Low:** 8.45%
⚠ **Moderate:** 60.58%
✓ **High:** 30.97%

Hint: use Ctrl+F to search for a model

-_Schufa_9831eb834a924b9f9c4591000c9387ef.bpmn ⚠
Control-Flow Complexity: 4
Complexity: moderate
Maintainability: moderate

sb_res_6f7c2f1257564661bff18871fc234864.bpmn ✓
Control-Flow Complexity: 0
Complexity: low
Maintainability: high

Warenversand_02f6546a9af14168b38ec7ded24fd874.bpmn ⚠
Control-Flow Complexity: 8
Complexity: moderate
Maintainability: moderate

-_Task_1_cb967b14c29e4f8f9d182d67559547a2.bpmn ⚠
Control-Flow Complexity: 9
Complexity: moderate
Maintainability: moderate

sb_res_7152ea8646bc4ac99bad213206ca37bd.bpmn ✓
Control-Flow Complexity: 2
Complexity: low
Maintainability: high

Warenversand_035d8eef52bc4e36aac840bdd2feff21.bpmn ⚠
Control-Flow Complexity: 5
Complexity: moderate
Maintainability: moderate

-_Task_3_84c455b8b0db45b388e7e176598502

sb_res_7224d163520d4dd185d4af485bcb0e02.bpmn ✓
Control-Flow Complexity: 0
Complexity: low

Warenversand_03ade2dafa5840d5a6bec06f2a56d618.bpmn ⚠
Control-Flow Complexity: 8
Complexity: moderate

Figure 7: Main page of the developed software tool

The main page contains the name of the software tool "BPMN-CFC Analysis Tool" and aggregated indicators that show the number of BPMN 2.0 models in the working directory (Git repository), the average, minimum, and maximum values of the CFC metric, as well as the number (in percent) of models with low, medium, and high usability.

As can be seen from Fig. 7, the analyzed set of 3729 business process models [24] has the following CFC metric indicators:

- average value – 5.03;
- minimum value – 0.00;
- maximum value – 24.

Therefore, it is possible to make assumptions about different levels of BPMN models maintainability from the analyzed data set [24]:

- 8.45% of BPMN models have low maintainability (i.e., complex to work with);
- 60.58% of BPMN models have moderate maintainability (i.e., comfortable to work with);
- 30.97% of BPMN models have high maintainability (i.e., easy to work with).

The specific results of the BPMN 2.0 business process models maintainability analysis are displayed in the form of cards with the file name, a visual mark on the model's maintainability (Fig. 7), the value of the CFC metric, the complexity and level of maintainability of the model.

Two BPMN models representing the business process of goods dispatch were used to perform control calculations. The first model is shown in Fig. 8.

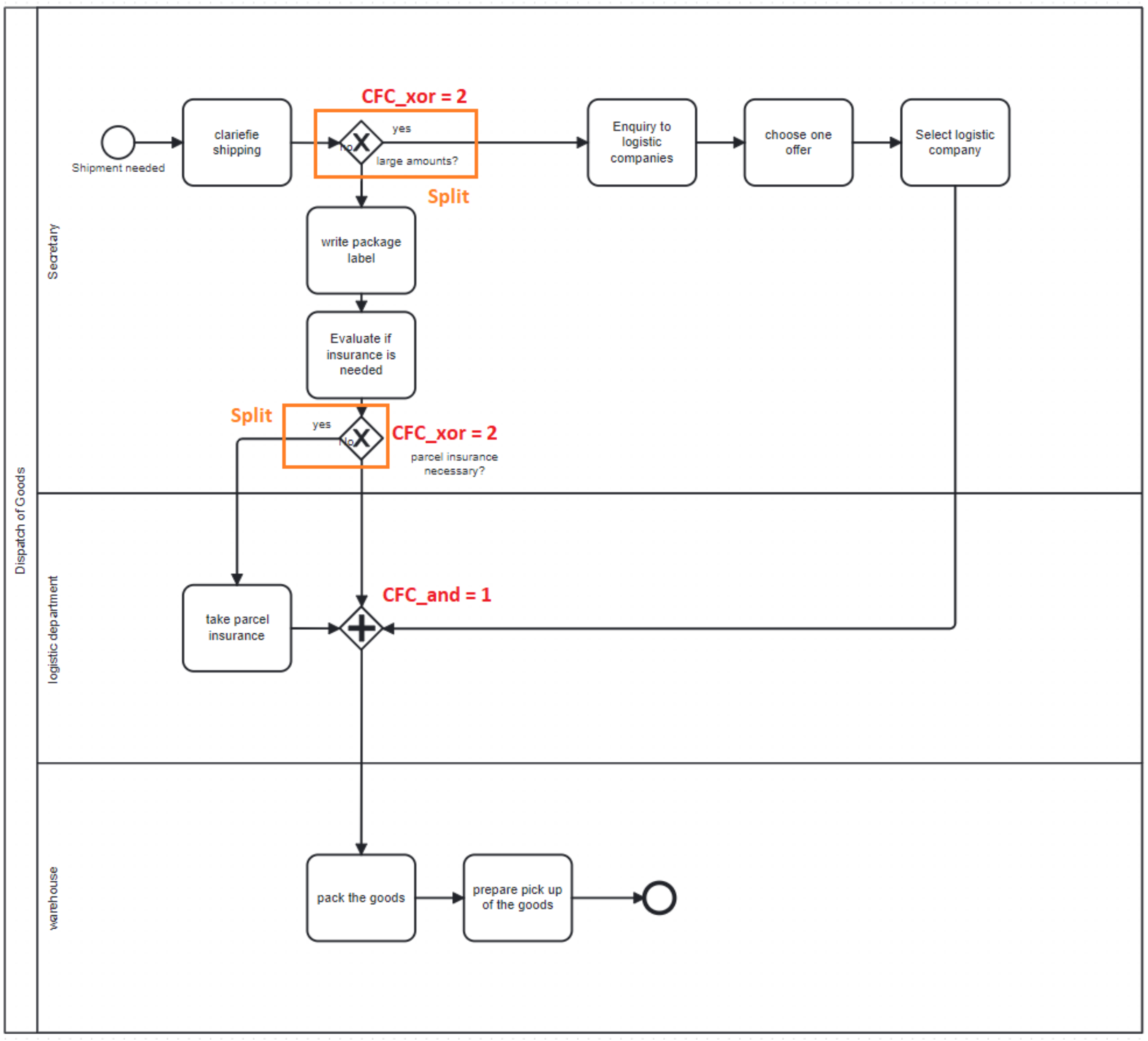


Figure 8: The first analyzed BPMN model of goods dispatch

The results of the analysis of this model (Fig. 8) indicate an average complexity ($CFC = 5$) and, accordingly, an average level of modifiability of this model (Fig. 10a).

The second model of the goods dispatch business process is shown in Fig. 9.

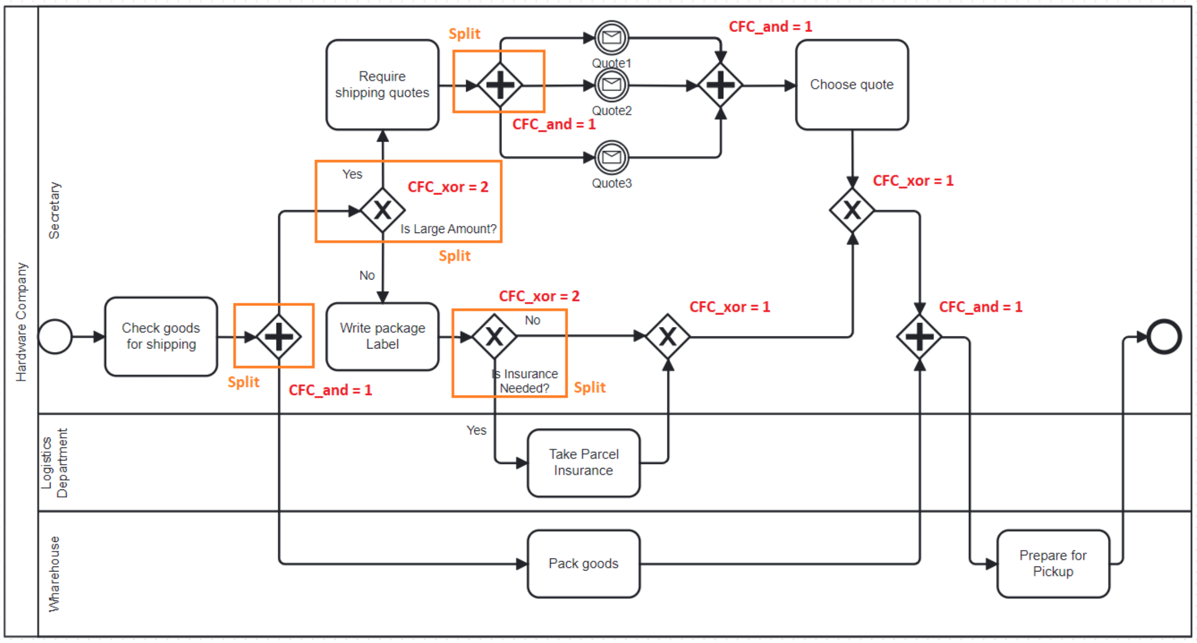
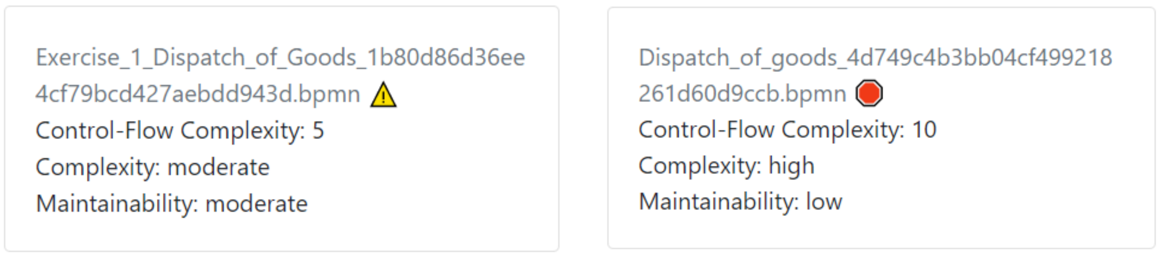


Figure 9: The second analyzed BPMN model of goods dispatch

This model (Fig. 9) is complex ($CFC = 10$) and has a low modifiability (Fig. 10b).

The maintainability evaluation results of the first and second BPMN models of the goods dispatch business process are shown in Fig. 10.



Exercise_1_Dispatch_of_Goods_1b80d86d36ee4cf79bcd427aebdd943d.bpmn ⚠
Control-Flow Complexity: 5
Complexity: moderate
Maintainability: moderate

Dispatch_of_goods_4d749c4b3bb04cf499218261d60d9ccb.bpmn 🛑
Control-Flow Complexity: 10
Complexity: high
Maintainability: low

a b

Figure 10: Evaluation results of the first and second BPMN models

It can be seen from Fig. 8 and Fig. 9 that the software tool assumes both split and joins gateways to calculate the maintainability level in compare to the original CFC metric (Fig. 11).

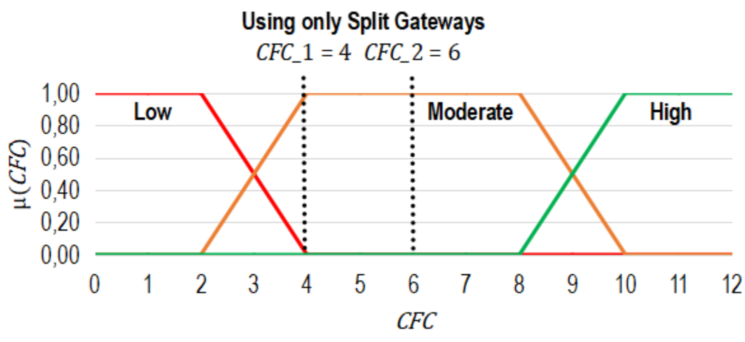


Figure 11: CFC levels when using only split gateways to assess model complexity

While the use of the original CFC metric results in lower complexity values (i.e., both considered BPMN models are assessed as “moderate” to maintain), the software tool assuming both split and join gateways results in more realistic estimates since join gateways provide additional complexity to the BPMN model and impact the model’s structuredness (i.e. when join gateways must correspond split gateways and vice versa).

Therefore, analyzed BPMN models are assessed as “moderate” and “high” to maintain (Fig. 12).

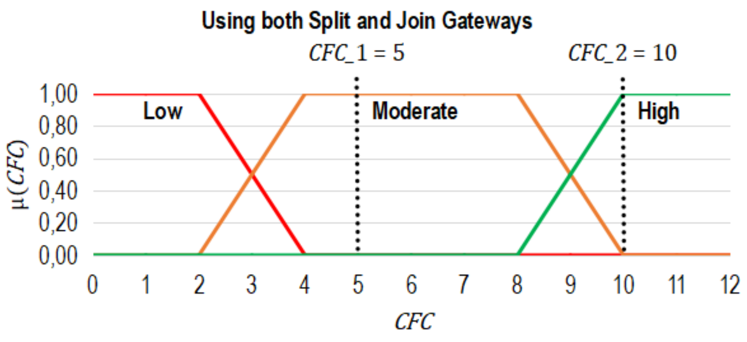


Figure 12: CFC levels when using both split and join gateways to assess model complexity

The use of fuzzy logic similar to human reasoning makes it easier for business process model authors to analyze the complexity and maintainability of BPMN models, and, consequently, design high-quality workflow diagrams to continuously improve enterprise solutions, and avoid errors and related costs.

7. Conclusions and Future Work

The maintainability of business process models plays a key role in obtaining a model that can be easily improved, modified, and adapted in accordance with changes in the organization and in the external environment. Thus, this study addresses the relevant problem of increasing the maintainability of business process models, which will allow avoiding the cost of correcting errors by identifying problems at the stage of domain analysis.

Thus, the following results were obtained in this study:

- the business process modeling domain and the existing metrics of business process models are reviewed – the CFC metric is chosen to build the fuzzy logic-based approach for BPMN analysis;
- the novel approach to the control-flow complexity measurement of business process models based on fuzzy logic is proposed – it resembles human reasoning to make the analysis results more understandable by BPMN model designers;
- the architecture of the software tool for assessing the maintainability of business process models based on the proposed approach is proposed – it assumes the idea of treating BPMN 2.0 models in the same way as other software engineering assets (i.e., source code files, configuration files, etc.) using the version-control system, this study assumes the use of Git;
- the software tool is developed using the Python programming language, the Flask and React web frameworks, the Bootstrap framework for building web interfaces, as well as the HTML, CSS, and JavaScript web development tools – using the proposed technology stack allows to rapidly build an initial prototype and perform the experiments;
- the solution of the business process model maintainability evaluation problem is demonstrated – the large collection of BPMN models from the Camunda GitHub repository is evaluated;
- the results are analyzed and discussed based on the CFC metric values obtained for the test set of BPMN 2.0 models – considering both split and join gateways allows to obtain a clearer picture regarding the complexity and maintainability of business process models.

Future work in this field includes elaboration of the proposed software solution and its application to the relevant cases of BPMN models in different domains. Also, it is planned to improve the proposed approach by introducing the elements of automatic business process model correction depending on the desired level of its complexity and maintainability.

8. References

[1] N. Ahrend, Opportunities and limitations of BPM initiatives in public administrations across levels and institutions, Humboldt University of Berlin, Berlin, 2014.

[2] M. Dumas, M. La Rosa, J. Mendling, H. A. Reijers, Introduction to Business Process Management, in: Fundamentals of Business Process Management, Springer, Berlin, Heidelberg, 2018, pp. 1–33. doi:10.1007/978-3-662-56509-4_1.

[3] F. Kahloun, S. A. Ghannouchi, A Classification Algorithm for Assessing the Quality Criteria for Business Process Models, in: International Conference on Hybrid Intelligent Systems, vol. 734, Springer, Cham, 2017, pp. 71–81. doi:10.1007/978-3-319-76351-4_8.

[4] H. B. Haj Ayech, S. A. Ghannouchi, E. A. El Hadj Amor, Extension of the BPM lifecycle to promote the maintainability of BPMN models, Procedia Computer Science 181 (2021) 852–860. doi:10.1016/j.procs.2021.01.239.

[5] H. A. Reijers, J. Mendling, J. Recker, Business Process Quality Management, in: Handbook on Business Process Management 1, International Handbooks on Information Systems, Springer, Berlin, Heidelberg, 2015, pp. 167–185. doi:10.1007/978-3-642-45100-3_8.

[6] J. Pavlicek, R. Hronza, P. Pavlickova, K. Jelinkova, The Business Process Model Quality Metrics, in: Enterprise and Organizational Modeling and Simulation, vol. 298, Springer, Cham, 2017, pp. 134–148. doi:10.1007/978-3-319-68185-6_10.

[7] R. Tricker, ISO 9001:2015 for Small Business, Taylor & Francis, 2016.

[8] A. Khudori, T. A. Kurniawan, F. Ramdani, Quality Evaluation of EPC to BPMN Business Process Model Transformation, Journal of Information Technology and Computer Science 5(2) (2020) 207–220. doi:10.25126/jitecs.202052176.

[9] H. G. Ceballos, V. Flores-Solorio, J. P. Garcia, A Probabilistic BPMN Normal Form to Model and Advise Human Activities, in: Engineering Multi-Agent Systems, vol. 9318, Springer, Cham, 2015, pp. 51–69. doi:10.1007/978-3-319-26184-3_4.

[10] C. Zhou, D. Zhang, D. Chen, C. Liu, Business Process Complexity Measurement: A Systematic Literature Review, IEEE Access 11 (2023) 47940–47955. doi:10.1109/ACCESS.2023.3275764.

[11] F. Kahloun, S. A. Ghannouchi, Quality Criteria and Metrics for Business Process Models in Higher Education Domain: Case of a Tracking of Curriculum Offers Process, Procedia Computer Science 100 (2016) 1016–1023. doi:10.1016/j.procs.2016.09.274.

[12] F. Corradini, A. Ferrari, F. Fornari, S. Gnesi, A. Polini, B. Re, G. O. Spagnolo, A Guidelines framework for understandable BPMN models, Data & Knowledge Engineering 113 (2018) 129–154. doi:10.1016/j.datak.2017.11.003.

[13] W. Kbaier, S. A. Ghannouchi, Determining The Threshold Values Of Quality Metrics In BPMN Process Models Using Data Mining Techniques, Procedia Computer Science 164 (2019) 113–119. doi:10.1016/j.procs.2019.12.161.

[14] J.-L. Boulanger, Requirements Specification of a Software Application, in: Certifiable Software Applications, 2018, pp. 43–73. doi:10.1016/B978-1-78548-119-2.50004-2.

[15] M. E. Rana, O. S. Saleh, High assurance software architecture and design, in: System Assurances, 2022, pp. 271–285. doi:10.1016/B978-0-323-90240-3.00015-1.

[16] PostgreSQL. URL: https://www.postgresql.org/.

[17] Git. URL: https://git-scm.com/.

[18] Python. URL: https://www.python.org/.

[19] Flask. URL: https://flask.palletsprojects.com/.

[20] React. URL: https://react.dev/.

[21] Bootstrap. URL: https://getbootstrap.com/.

[22] J. Osis, U. Donins, Unified Modeling Language: A Standard for Designing a Software, in: Topological UML Modeling, 2017, pp. 3–51. doi:10.1016/B978-0-12-805476-5.00001-0.

[23] A. Torres, R. Galante, M. S. Pimenta, A. J. B. Martins, Twenty years of object-relational mapping: A survey on patterns, solutions, and their implications on application design, 2017, pp. 1–18. doi:10.1016/j.infsof.2016.09.009.

[24] BPMN for research. URL: https://github.com/camunda/bpmn-for-research/.