Communication papers of the Federated Conference on Computer Science and Information Systems, pp. 295–302 DOI: 10.15439/2017F308 ISSN 2300-5963 ACSIS, Vol. 13

Overview of Verification Tools for Business Process Models

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

Piotr Wiśniewski, Antoni Ligęza
AGH University of Science and Technology
al. A. Mickiewicza 30, 30-059 Krakow, Poland
E-mail: {wpiotr,ligeza}@agh.edu.pl

***Abstract*—Formal verification of process models is an important issue in Business Process Management. Such a verification provides the information about the correctness of a process model, can be also used for checking business compliance or as a preliminary step to simulation. In this paper, we provide an overview of the existing tools for such a verification.**

***Index Terms*—process models verification, process models anomalies, verification tools, business process verification**

I. Introduction

DESIGNING process models is currently a broad term. Considering designing processes, one can think about models designed manually by business analysts, as well as models generated from other representations such as: natural text description [1], structured text [2], spreadsheets [3], other representations like UML [4], or mining such models from event logs [5]. All such models can suffer from various anomalies [6]–[8]. Some of such anomalies can be avoided by validation models [9], [10] or verification [11] of models.

Business Process Model and Notation (BPMN) [12] is the most common notation for representing process models. BPMN supports process documentation, communication and visualization using clear graphical representation understandable even for non-technical business people. Although it provides a standardized understanding of BPMN elements and constructs, no formal semantics is provided within this specification. This can lead to misinterpretations or errors in models. Thus, there is a field for studies focusing on formal representation and verification of such process models.

Such a verification of process models can provide the information about the correctness of a process model in terms of syntactic and structural anomalies of models. The main focus of this paper is to present an overview of the verification tools for process models which can detect some anomalies in business processes. This paper provides an overview of several existing tools. Their capabilities and properties were analyzed.

The rest of the paper is organized as follows. Section II covers the presentation of the BPMN notation. Section III and IV give the literature overview on process anomalies, especially concerning anomalies in BPMN models. Section V provides the description and comparison of various verification tools used for verifying business process models. The final section concludes the paper.

The paper is supported by the AGH UST grant.

II. Business Process Model and Notation

Process models in BPMN are represented as diagrams modeled using a limited set of graphical elements. There are four main groups of elements, namely: flow objects, connecting objects, swimlanes and artifacts.

Flow objects are the key elements describing the process. The set of flow objects consist of three core element types: events, activities and gateways (see Fig. 1):

1) activities – represented by rounded-corner rectangles, describe the work that has to be completed within a process,
2) events – represented by circles, describe something that happen during the time of the process,
3) gateways – represented by diamond shapes, control the flow of the token between flow objects.

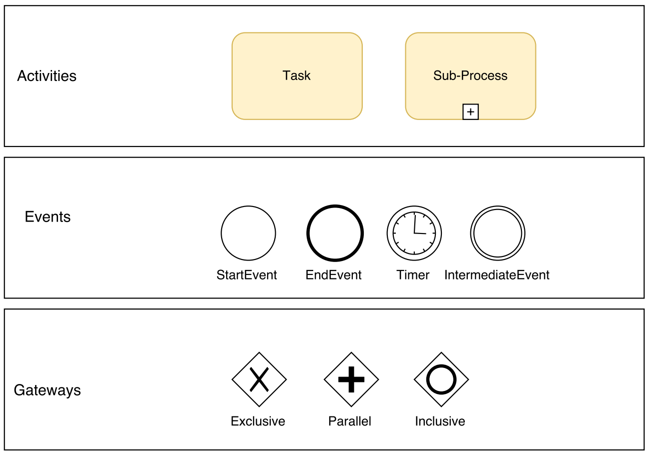

Fig. 1. BPMN flow objects of internal Business Process Model

In the case of activities, there are two kinds of them: tasks and sub-processes. A task represents a single unit of work should not be divided. A sub-process is used for complex work which can be divided into smaller units and specified in the lower level as a separated process.

There are three types of events: the start, intermediate and the end event. The start event works like a trigger to a process and shows its beginning (and under what conditions the process begins). The end event indicates where the business process ends. The intermediate event represents what happens between start and end events.

Gateways constitute the mechanism of controlling the way in which business process is executed. The most typical types of gateways are as follows:

- data-based exclusive gateway – used for controlling the process flow based on given process data (see Fig. 2),

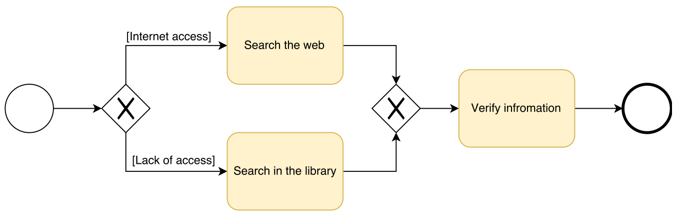

Fig. 2. Data-Based Exclusive Gateway

- inclusive gateway – used for creating potentially parallel paths based on the conditions of all outgoing flows (see Fig. 3),

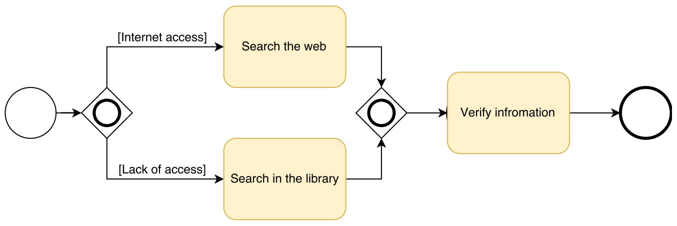

Fig. 3. Inclusive gateway

- parallel gateway – used for paralleling the flows without the need of checking any conditions (see Fig. 4).

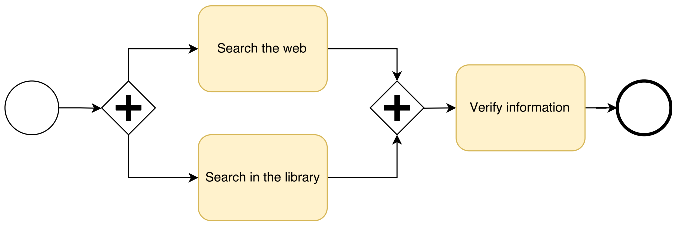

Fig. 4. Parallel gateway

- event-based gateway – used for modeling alternative paths that are based on events (Fig. 5).

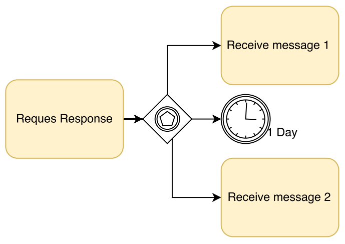

Fig. 5. Event-based gateway

Flow objects are connected using connecting objects, which are of three types: sequences, messages, and associations. Sequence flow shows the order in which particular activities are performed in a process. Message flow shows the flow of messages between two process participants. Association links some artifacts to activities, events, gateways or flows.

A simple auction process, presenting the basic BPMN flow objects, is depicted in Fig. 6.

For demonstrating what business function a particular flow object is connected with or by which part of system it is executed, BPMN usually uses the concept of swimlanes. Participants are represented as pools, which can be divided into sub-partitions called lanes. These can be used for representing specific objects or roles in a process. Artifacts, in turn, are diagram elements which show additional pieces of information. BPMN naively supports several kinds of artifacts like data objects and data stores, groups and annotations.

Additionally, many different extensions of BPMN have been proposed for capturing various aspects of business processes [13]–[16]. Thus, in such an advanced and complex notation, it is hard to avoid model anomalies. Moreover, it is possible to mistake a correct model with incorrect one, especially because two BPMN models with different structure, but behaviorally equivalent, can be both correct and unambiguous [17]. It is because the BPMN notation allows for expressing the same semantics using various syntactic structures. However, it is possible to transform such equivalent structures to the equivalent ones [18]. Another way is to use some kind of a guide which provides help in modeling. This can be done using some recommendation technique during process modeling [19]–[21] or the environment capable to detect some semantic issues [22]–[25].

III. Applications of the Verification in the Area of Anomalies in Process Models

There are many possibilities of defining incoherent business logic specification and its interpretation. Even in basic process models, some anomalies can be observed [26]. An improvement is required in the mechanism which provides cohesion in detecting anomalies in business processes [27]. Anomalies have been defined in numerous papers [28], yet a unified definition was presented in IEEE standard classification for Software Anomalies [29]: *Each condition different from the expected is an anomaly*. In a business logic, an anomaly can be considered as every negative influence on modeling and models. There is a special kind of anomaly — a defect, which blocks the correct and efficient flow of objects completely.

Some anomalies can be found by searching the BPMN models for some patterns. In [30], several anti-patterns are found using a query language for BPMN. In [31], typical gateway constellations leading to problematic situations in the flow work diagram are presented. A similar solution was used in [32], where an 'anomaly pattern' are detected using anti-patterns in the data flow. Patterns as well as anti-patterns can be represented as LTL formulae and be used in formal

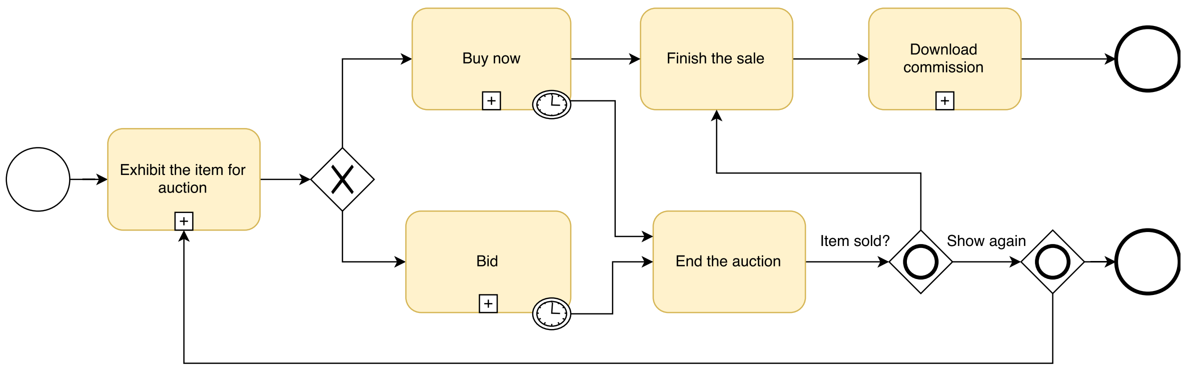

Fig. 6. An example of a process model in BPMN

verification of a business process model transformed to a high-level Petri Net [33]. The approach of Lam [34] transforms BPMN models to the New Symbolic Model Verifier (NuSMV) language in order to do a model-checking analysis. This approach has formal foundations and addresses the correctness issue of the transformation. It requires to encode properties of a model using Computation Tree Logic (CTL) formulas. Another approach, which stems from formalism or inadequacy of the tools, is presented in [35].

Control flow anomalies concern problems connected with flow control and gateways conditions [36]. In [37], a problem of control over many semantically identical connections between two workflow elements was presented. This multiplicity complicates changes in the workflow, which is not desired. Roy et al. [38] described the method of control flow error detection based on Petri Net analysis and evaluated it on a set of industrial examples. An extensive survey of process model verification techniques regarding flow correctness and variability was presented in [39].

Formal verification of workflow-oriented software models based on the semantic tableaux method that uses the deductive approach was proposed in [40].

Ou-Yang and Lin proposed a Petri-net-based approach [41], which evaluates a feasibility of a BPMN model, e.g. to reveal deadlocks and infinite loops. This approach consists in manually translating of the BPMN model to the Modified BPEL4WS representation, and then to Colored Petri-net XML (CPNXML). The CPNXML representation can be then verified using CPN Tools. The approach has some major restrictions, such as limited assessment criteria, and lack of support of the multiple merge and split conditions.

Gateways can also caused some anomalies. XOR-gateways with undefined gateway conditions can cause practical problems or even be a reason of an error. A similar thing occur when XOR-gateways conditions do not exclude each other and partially or fully overlap. What happens in flow control in a case of a lack of synchronization is multiple flow execution. For example, branches and some loop instruction cause such an anomaly [42].

A flow deadlock is a situation, in which the workflow is stopped in the current position of the path and cannot be accomplished. Another lock of flow is known as livelock. In [30] it is called an 'infinite loop'. Flow livelock keeps the operating work flow system in an infinite loop. The reasons are bad modeling conditions, which prevent leaving the loop. An approach based on business process event logs, where both deadlocks and livelocks can be detected as well as tasks which can never be executed, was presented in [43].

Badica et al. consider formalized models using Role Activity Diagrams for BP business process verification [44] as well as including logic-based ones in similar multi-agent approaches [45]. Other formalized approaches in modeling complex heterogeneous information systems include [46].

Another class of anomalies in processes are anomalies in the rules used in a process model. Such rule-based anomalies are described in a number of papers [47]–[51]. These involve several problems connected with rules, such as: rule-base consistency (concerning the coherence of rules), rule-base livelock, also called 'circular rules' [48] or rule-base deadlock. This type of anomaly suggests that rule-base does not encompass the basic context in which it is used. Coverage anomalies concern the rules in which conditions can be fulfilled by the context but conclusions are modeled in such a way that no effect will ever be seen. A formal verification of business rules violations using Business Rule Language (BRL) and Depth-First Search algorithm was conducted in [52].

Business process related anomalies may also refer to the violation of temporal constraints and dependencies added to the process model as well as can be used for validation of time related processes [53]–[59]. In [59], a BPMN model is transformed to a set of Timed Automata (TA) and is verified using Clocked Computation Tree Logic (CCTL). A similar method is described in [57], where BPMN models are verified using TA-networks with respect to business performance indicators. Temporal properties of a business process can be also verified using a framework based on the declarative specification of a process model and the Answer Set Programming (ASP) technique [58]. Dynamic detection of temporal violations and providing possible solutions to a specific problem was proposed in [55]. In [56], in turn, the authors proposed a solution where an extended BPMN model is mapped onto timed automata and then verified using UPPAAL model checker.

IV. Anomalies in business processes

There is a number of business process anomalies which can occur during modeling business processes. A common classification distinguishes: syntactic and structural anomalies.

A. Syntactic Anomalies in Business Process

Syntactic anomalies constitute the problem in improper usage of modeling elements. Such anomalies can be divided into four groups:

- Incorrect usage of activities – this anomaly results from improper use of start or end event. The BPMN specification defines the start and end events as optional. However, if there is a start event used in a diagram, each activity – that do not have an end event on its path – can be considered as incorrect.
- Invalid use of gateway – this anomaly is especially connected with the data-based XOR gateway and event-based XOR gateway. The data-based gateway has to use a data to determine the token flow, so using event-based objects for data-based gateway is incorrect. In the case of the event-based gateway, it cannot be used as a merge gateway, but can only be used as a decision type gateway.
- Incorrect usage of connecting object – this anomaly is concerned with such situations as using a message flow within the pool or using conditional flow from the event type source.
- Incorrect usage of swimlanes – such an anomaly can occur when a model uses multiple pools as a single process where message flows indicate sequence of activities. This can lead to situation, where activities in a pool are not connected with sequence flows (see Fig. 7). On the other hand, there are also situations of improper use of lane as a pool.

B. Structural Anomalies

Structural anomalies are broadly described in the literature [60]–[64]; such anomalies correspond to a wrong dynamic behavior and can be divided into four types:

- Deadlock – a situation, in which the flow cannot continue because a requirement of the model is not satisfied. There are two types of deadlocks: deterministic deadlock, when concurrent flow are connected by the AND-join (parallel) gateway (see Fig. 8) and non-deterministic one, if they are connected by an OR-join (inclusive) or complex gateway.
- Lack of synchronization – a situation where there is more than one token on some sequence flow that it should be. It occurs when concurrent paths (starting with an OR-split or an AND-split) are joined by an XOR-join (see Fig. 9). If the paths were split by the AND gateway, this problem is deterministic. It is non-deterministic, if decision about splitting is made at the execution time.
- Dead Activity – a situation, in which there is an activity which cannot be executed, because there is no path leading from the start event to this activity (see Fig. 10).
- Infinite Loop – also called the closed loop – is a cycle in the process, in which token is looped and can not escape the loop. This can be caused by improper use of gateways (see Fig. 11). If such gateway is of the OR type, the loop is non-deterministic; if it is the AND type, then the loop is deterministic.

V. Verification tools

There are several automatic or interactive tools for process model verification such as: Signavio (BpStruct, LoLA) [65], UPPAAL [66], SPIN, Wolfan, NuSMV, nuXmv, and Alvis. This section provides the analysis of several of such tools.

A. Signavio

Signavio [65] uses colored Petri nets for process verification. These can handle unreachable states (such as unreachable activities) as well as deadlocks (caused by wrong usage of gateways). Signavio also uses two additional tools BPStruct and LoLA [65].

As it comes to the BPMN syntax, Signavio can check the usage of the elements from the defined BPMN subset, as well as mandatory attributes, definition of required dictionary links and consistency with attributes of the linked dictionary item, uniqueness of element names, etc. In the process structure, usage of different elements in various contexts can be checked, e.g. usage of activities before or-splits, consistent usage of signals correct usage of boundary events, message flows, etc. Additionally, absence of loops, deadlocks, multi merges, subprocess relation cycles, multiple incoming sequence flows can be checked.

There are many properties checked by Signavio's tool, for example:

- Lack of the flow source or flow target.
- Source and target of the sequence flow are not part of the same process.
- Start-event without outgoing flows.
- Event-based gateway with less than two outputs.

BPStruct supports transforming unstructured process models into well-structured ones. A model can be called well-structured, if for every node with multiple outgoing arcs (a split), there is a corresponding node with multiple incoming arcs (a join), and the other way round. Such the fragment of the model between the split and the join should form a single-entry-single-exit (SESE) component.

LoLA (Low Level Petri Net Analyzer) is a tool used to verify a model in Signavio. The LoLA can load the model as a Petri net and the provided properties specified in CTL*. These properties are analyzed and in the case a property is not fulfilled, it can provide a counter-example.

B. UPPAAL

UPPAAL [66], [67] is an integrated tool for modeling, simulation and verification of real-time systems. It consists of two main parts: a graphical user interface and a model-checker engine. Its model-checker is based on symbolic processing, which reduces the problem of verification to constraint programming. It is an appropriate tool for systems which can be modeled as a set of non-deterministic processes with finite

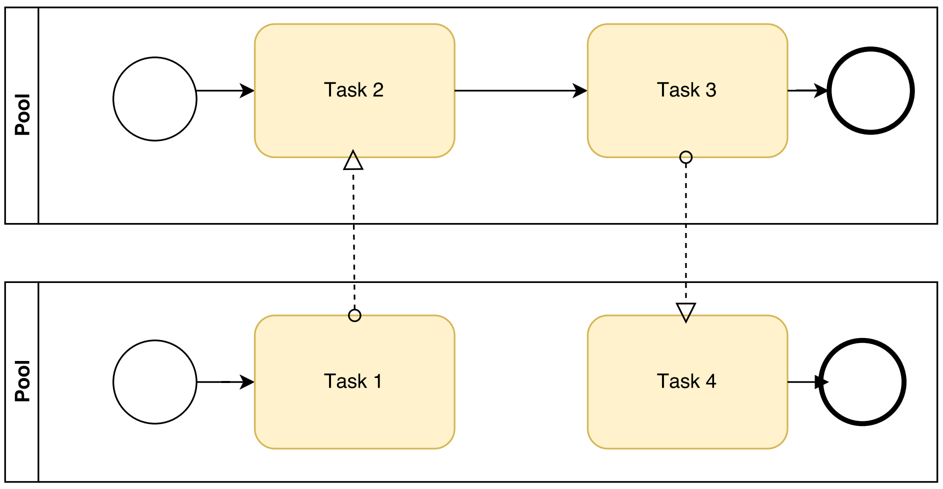

Fig. 7. Missing sequence flow

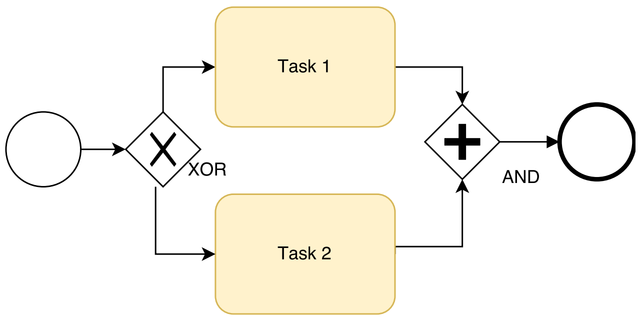

Fig. 8. Deterministic deadlock

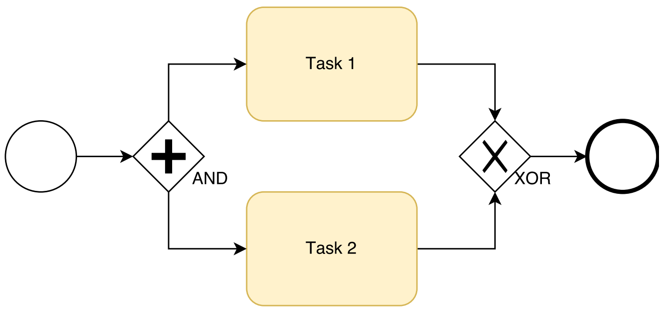

Fig. 9. Lack of synchronization

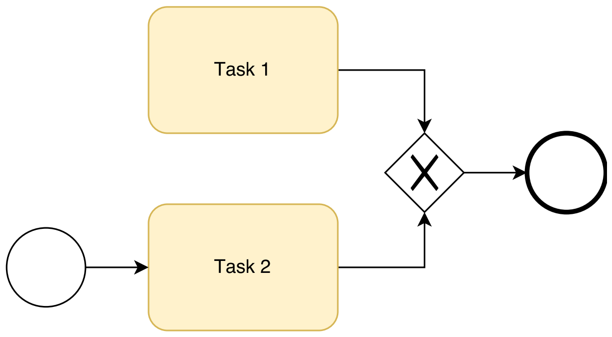

Fig. 10. Dead Activities

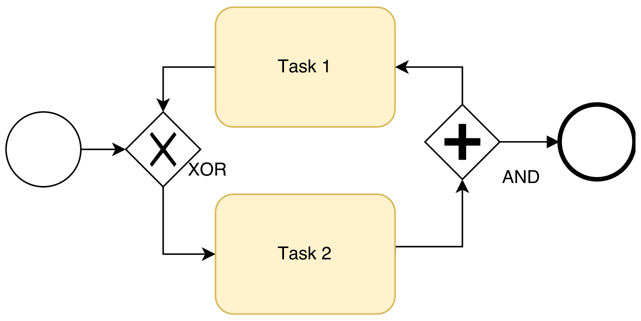

Fig. 11. Infinite Loop

control structure and real-valued clocks that communicates using channels or shared variables [68].

UPPAAL provides a graphical editor and a graphical simulator as well as path generator for system verification and visualization.

C. SPIN

SPIN model checker [69] is an open-source verification tool which can be used to verification of multithreaded applications. SPIN is also a general tool for verifying the correctness of models in an automated fashion. It is oriented on checking the interactions of processes.

The tool provides 'on-the-fly' verification (the full state graph is not required, so much larger models can be verified) as well as it supports native C code, what allows code based verification.

D. Woflan

Woflan (WOrkFLow ANalyser) [70], [71] is a workflow analysis tool, which checks if a Petri net conforms to specified restrictions for workflows [72]. Woflan can help in detecting errors made at design-time Woflan also helps in providing the diagnostics why the process is wrong and how it can be repaired. For that reason, Woflan generates high-quality diagnostic information, which can guide the designer towards the error. The advantage of the tool is that it transforms the model into the coverage graph, which often causes a significant reduction in state space.

E. NuSMV

NuSMV [73], [74] is a symbolic model checker for temporal logic. In this approach, the process is modeled as a finite state transition system and specifications are given as formulae, which can be expressed in LTL or CTL logic. This tool is used to verify if the model satisfies the provided specifications. The input file with a model definition in SMV language can be generated based on a Petri Net [75] or behavioral elements extracted from a business process specification [76].

F. nuXmv

Another symbolic model checker is nuXmv [77], an extension of the NuSMV tool, which can be used for both finite and infinite state transition systems. It offers more functionalities such as boundedness and liveness verification. It also allows a user to generate an explicit state representation in a form of an XMI file which can be then visualized as a UML diagram. Process model verification using nuXmv can be executed by generating model definition from the coverability graph of a real-time coloured Petri Net [78].

G. Alvis

Alvis is a formal modelling language, tool set, and framework created to verify and model check distributed concurrent systems [79]–[82]. The advantage of Alvis modelling framework are readable graphical and a code layers of specification of modelled system behaviour. The graphical layer presents data exchange channels of communicating distributed units and enables hierachical specification that hides complexity of huge models [83]. The code layer specifies behaviour of particular distributed units using mixture of domain specific language (called Alvis language) and Haskell. The possibility of exchanging Haskell with other standard programming languages was studied in [84].

Alvis was used in BPMN model verification [85], [86]. One can verify time constraints of concurrent system with time version of Alvis framework [87]. The Alvis tool set may be used not only to verify a model of concurrent system but to simulate a provided model together with computation of time dependent statistics of simulation results [88], [89].

VI. CONCLUSION

The relatively broad spectrum of tools for a process model verification is discussed in this paper. We have given the literature overview on process anomalies, especially concerning anomalies in BPMN models and provided the description of various verification tools used for verifying process models.

The research presented in this paper is a proposal for further studies related to verification issues of BPMN process models. Our future work will focus on practical assessment of process models especially with the existing tools [90]–[93].

REFERENCES

[1] F. Friedrich, J. Mendling, and F. Puhlmann, "Process model generation from natural language text," in *Advanced Information Systems Engineering*, ser. Lecture Notes in Computer Science, H. Mouratidis and C. Rolland, Eds. Springer Berlin Heidelberg, 2011, vol. 6741, pp. 482–496.

[2] K. Kluza and K. Honkisz, "From SBVR to BPMN and DMN models. proposal of translation from rules to process and decision models," in *Artificial Intelligence and Soft Computing: 15th International Conference, ICAISC 2016, Zakopane, Poland, June 12-16, 2016, Proceedings, Part II*, ser. Lecture Notes in Computer Science, L. Rutkowski, M. Korytkowski, R. Scherer, R. Tadeusiewicz, L. A. Zadeh, and J. M. Zurada, Eds. Springer International Publishing, 2016, vol. 9693, pp. 453–462.

[3] K. Kluza and P. Wiśniewski, "Spreadsheet-based business process modeling," in *Computer Science and Information Systems (FedCSIS), 2016 Federated Conference on*. IEEE, 2016, pp. 1355–1358.

[4] J. R. Nawrocki, T. Nedza, M. Ochodek, and L. Olek, "Describing business processes with use cases," in *BIS*, 2006, pp. 13–27.

[5] A. A. Kalenkova, M. de Leoni, and W. M. van der Aalst, "Discovering, analyzing and enhancing BPMN models using ProM?" in *Business Process Management-12th International Conference, BPM*, 2014, pp. 7–11.

[6] A. Ligęza, K. Kluza, and T. Potempa, "AI approach to formal analysis of BPMN models. towards a logical model for BPMN diagrams," in *Proceedings of the Federated Conference on Computer Science and Information Systems – FedCSIS 2012, Wroclaw, Poland, 9-12 September 2012*, M. Ganzha, L. A. Maciaszek, and M. Paprzycki, Eds., 2012, pp. 931–934.

[7] A. Suchenia and A. Ligęza, "Event anomalies in modeling with BPMN," *International Journal of Computer Technology & Applications*, vol. 6, no. 5, pp. 789–797, 2015.

[8] A. Mroczek and A. Ligeza, "A note on BPMN analysis. Towards a taxonomy of selected potential anomalies," in *Computer Science and Information Systems (FedCSIS), 2014 Federated Conference on*. IEEE, 2014, pp. 1097–1102.

[9] M. A. Mach and M. L. Owoc, "Validation as the integral part of a knowledge management process," in *Proceeding of Informing Science Conference*, 2001.

[10] M. Mach-Król and K. Michalik, "Validation and verification of temporal knowledge as an important aspect of implementing a temporal knowledge base system supporting organizational creativity," in *Computer Science and Information Systems (FedCSIS), 2015 Federated Conference on*. IEEE, 2015, pp. 1315–1320.

[11] M. Mach-Król and K. Michalik, "Verification of temporal knowledge bases as an important aspect of knowledge management processes in organization," in *Advances in Business ICT: New Ideas from Ongoing Research*. Springer, 2017, pp. 1–15.

[12] OMG, "Business Process Model and Notation (BPMN): Version 2.0 specification," Object Management Group, Tech. Rep. formal/2011-01-03, January 2011.

[13] A. Yousfi, C. Bauer, R. Saidi, and A. K. Dey, "ubpmn: A bpmn extension for modeling ubiquitous business processes," *Information and Software Technology*, vol. 74, pp. 55–68, 2016.

[14] R. Martinho, D. Domingos, and J. Varajão, "Cf4bpmn: a bpmn extension for controlled flexibility in business processes," *Procedia Computer Science*, vol. 64, pp. 1232–1239, 2015.

[15] R. M. Pillat, T. C. Oliveira, P. S. Alencar, and D. D. Cowan, "Bpmnt: A bpmn extension for specifying software process tailoring," *Information and Software Technology*, vol. 57, pp. 95–115, 2015.

[16] K. Kluza, K. Jobczyk, P. Wiśniewski, and A. Ligęza, "Overview of time issues with temporal logics for business process models," in *Computer Science and Information Systems (FedCSIS), 2016 Federated Conference on*. IEEE, 2016, pp. 1115–1123.

[17] K. Kluza and K. Kaczor, "Overview of BPMN model equivalences: towards normalization of BPMN diagrams," in *8th Workshop on Knowledge Engineering and Software Engineering (KESE2012) at the at the biennial European Conference on Artificial Intelligence*

(ECAI 2012): August 28, 2012, Montpellier, France, J. Canadas, G. J. Nalepa, and J. Baumeister, Eds., 2012, pp. 38–45. [Online]. Available: http://ceur-ws.org/Vol-949/

[18] V. S. W. Lam, "Equivalences of BPMN processes," *Service Oriented Computing and Applications*, vol. 3, no. 3, pp. 189–204, 2009.

[19] S. Bobek, G. J. Nalepa, and O. Grodzki, "Integration of activity modeller with bayesian network based recommender for business processes," in *Proceedings of 10th Workshop on Knowledge Engineering and Software Engineering (KESE10) co-located with 21st European Conference on Artificial Intelligence (ECAI 2014), Prague, Czech Republic, August 19 2014*, ser. CEUR Workshop Proceedings, G. J. Nalepa and J. Baumeister, Eds., vol. 1289, 2014. [Online]. Available: http://ceur-ws.org/Vol-1289/kese10-05_submission_10.pdf

[20] S. Bobek, M. Baran, K. Kluza, and G. J. Nalepa, "Application of bayesian networks to recommendations in business process modeling," in *Proceedings of the Workshop AI Meets Business Processes 2013 co-located with the 13th Conference of the Italian Association for Artificial Intelligence (AI*IA 2013), Turin, Italy, December 6, 2013*, L. Giordano, S. Montani, and D. T. Dupre, Eds., 2013. [Online]. Available: http://ceur-ws.org/Vol-1101/

[21] K. Kluza, M. Baran, S. Bobek, and G. J. Nalepa, "Overview of recommendation techniques in business process modeling," in *Proceedings of 9th Workshop on Knowledge Engineering and Software Engineering (KESE9) co-located with the 36th German Conference on Artificial Intelligence (KI2013), Koblenz, Germany, September 17, 2013*, G. J. Nalepa and J. Baumeister, Eds., 2013. [Online]. Available: http://ceur-ws.org/Vol-1070/

[22] G. J. Nalepa, K. Kluza, and U. Ciaputa, "Proposal of automation of the collaborative modeling and evaluation of business processes using a semantic wiki," in *Proceedings of the 17th IEEE International Conference on Emerging Technologies and Factory Automation ETFA 2012, Kraków, Poland, 28 September 2012*, 2012.

[23] K. Kluza, K. Kaczor, G. Nalepa, and M. Slazynski, "Opportunities for business process semantization in open-source process execution environments," in *Computer Science and Information Systems (FedCSIS), 2015 Federated Conference on*, Sept 2015, pp. 1307–1314.

[24] K. Kluza, G. J. Nalepa, M. Ślażyński, K. Kutt, E. Kucharska, K. Kaczor, and A. Łuszpaj, "Overview of selected business process semantization techniques," in *Advances in Business ICT: New Ideas from Ongoing Research*. Springer, 2017, pp. 45–64.

[25] W. T. Adrian, S. Bobek, G. J. Nalepa, K. Kaczor, and K. Kluza, "How to reason by HeaRT in a semantic knowledge-based wiki," in *Proceedings of the 23rd IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2011*, Boca Raton, Florida, USA, November 2011, pp. 438–441. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6103361&tag=1

[26] J. Mendling, H. Verbeek, B. F. van Dongen, W. M. van der Aalst, and G. Neumann, "Detection and prediction of errors in epcs of the sap reference model," *Data & Knowledge Engineering*, vol. 64, no. 1, pp. 312–329, 2008.

[27] A. Hallerbach, T. Bauer, and M. Reichert, "Capturing variability in business process models: the provop approach," *Journal of Software Maintenance and Evolution: Research and Practice*, vol. 22, no. 6-7, pp. 519–546, 2010.

[28] A. Suchenia, T. Potempa, A. Ligęza, K. Jobczyk, and K. Kluza, "Selected approaches towards taxonomy of business process anomalies," in *Advances in Business ICT: New Ideas from Ongoing Research*. Springer, 2017, pp. 65–85.

[29] I. Group *et al.*, "1044-2009-ieee standard classification for software anomalies," *IEEE, New York*, 2010. [Online]. Available: https://standards.ieee.org/findstds/standard/1044-2009.html

[30] R. Laue and A. Awad, "Visualization of business process modeling anti patterns," *Electronic Communications of the EASST*, vol. 25, 2009.

[31] S. Kühne, H. Kern, V. Gruhn, and R. Laue, "Business process modeling with continuous validation," *Journal of Software Maintenance and Evolution: Research and Practice*, vol. 22, no. 6-7, pp. 547–566, 2010.

[32] N. Trčka, W. M. Van der Aalst, and N. Sidorova, "Data-flow anti-patterns: Discovering data-flow errors in workflows," in *Advanced Information Systems Engineering*. Springer, 2009, pp. 425–439.

[33] A. Kheldoun, K. Barkaoui, and M. Ioualalen, "Formal verification of complex business processes based on high-level petri nets," *Information Sciences*, vol. 385–386, pp. 39 – 54, 2017.

[34] V. S. W. Lam, "Formal analysis of BPMN models: a NuSMV-based approach," *International Journal of Software Engineering and Knowledge Engineering*, vol. 20, no. 7, pp. 987–1023, 2010.

[35] N. Lohmann and K. Wolf, "How to implement a theory of correctness in the area of business processes and services," in *Business Process Management*. Springer, 2010, pp. 61–77.

[36] S. A. White, "Process modeling notations and workflow patterns," *Workflow handbook*, vol. 2004, pp. 265–294, 2004.

[37] L. Olkhovich, "Semi-automatic business process performance optimization based on redundant control flow detection," in *Telecommunications, 2006. AICT-ICIW'06. International Conference on Internet and Web Applications and Services/Advanced International Conference on*. IEEE, 2006, pp. 146–146.

[38] S. Roy and A. S. M. Sajeev, "A formal framework for diagnostic analysis for errors of business processes," in *Transactions on Petri Nets and Other Models of Concurrency XI*, M. Koutny, J. Desel, and J. Kleijn, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 226–261.

[39] H. Groefsema and D. Bucur, "A survey of formal business process verification: From soundness to variability," in *Proceedings of International Symposium on Business Modeling and Software Design*. SciTePress, 2013, pp. 198âĂŞ–203.

[40] R. Klimek, "A system for deduction-based formal verification of workflow-oriented software models," *International Journal of Applied Mathematics and Computer Science*, vol. 24, no. 4, pp. 941–956, 2014.

[41] C. Ou-Yang and Y. D. Lin, "BPMN-based business process model feasibility analysis: a petri net approach," *International Journal of Production Research*, vol. 46, no. 14, pp. 3763–3781, 2008.

[42] R. Liu and A. Kumar, "An analysis and taxonomy of unstructured workflows," in *Business Process Management*. Springer, 2005, pp. 268–284.

[43] O. Allani and S. A. Ghannouchi, "Verification of BPMN 2.0 process models: An event log-based approach," *Procedia Computer Science*, vol. 100, pp. 1064 – 1070, 2016.

[44] A. Badica and C. Badica, "Formal verification of business processes represented as role activity diagrams," in *Federated Conference on Computer Science and Information Systems – FedCSIS 2011, Szczecin, Poland, 18-21 September 2011, Proceedings*, M. Ganzha, L. A. Maciaszek, and M. Paprzycki, Eds., 2011, pp. 277–280.

[45] A. Badica and C. Badica, "Fsp and fltl framework for specification and verification of middle-agents," *Applied Mathematics and Computer Science*, vol. 21, no. 1, pp. 9–25, 2011.

[46] J. Stepaniuk, J. G. Bazan, and A. Skowron, "Modelling complex patterns by information systems," *Fundam. Inform.*, vol. 67, no. 1-3, pp. 203–217, 2005.

[47] D. Xu, K. Xia, D. Zhang, and H. Zhang, "Model checking the inconsistency and circularity in rule-based expert systems," *Computer and Information Science*, vol. 2, no. 1, p. 12, 2009.

[48] A. K. Zaidi and A. H. Levis, "Validation and verification of decision making rules," *Automatica*, vol. 33, no. 2, pp. 155–169, 1997.

[49] M. Dohring and S. Heublein, "Anomalies in rule-adapted workflows-a taxonomy and solutions for vbpmn," in *Software Maintenance and Reengineering (CSMR), 2012 16th European Conference on*. IEEE, 2012, pp. 117–126.

[50] A. Ligęza and G. J. Nalepa, "A study of methodological issues in design and development of rule-based systems: proposal of a new approach," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 1, no. 2, pp. 117–137, 2011.

[51] M. A. Mach and P. J. Kalczynski, "Technique for reducing the number of rules in a temporal knowledge base." in *BIS*, 2006, pp. 442–454.

[52] A. Rachdi, A. En-Nouaary, and M. Dahchour, "Verification of common business rules in bpmn process models," in *Networked Systems: 4th International Conference, NETYS 2016, Marrakech, Morocco, May 18-20, 2016, Revised Selected Papers*, P. A. Abdulla and C. Delporte-Gallet, Eds. Cham: Springer International Publishing, 2016, pp. 334–339.

[53] S. Cheikhrouhou, S. Kallel, N. Guermouche, and M. Jmaiel, "The Temporal Perspective in Business Process Modeling : An Evaluative Survey and Research Challenges," *Service Oriented Computing and Applications*, vol. 9, no. 1, pp. 75–85, 2015.

[54] R. Klimek and P. Szwed, "Verification of archimate process specifications based on deductive temporal reasoning," in *Computer Science and Information Systems (FedCSIS), 2013 Federated Conference on*. IEEE, 2013, pp. 1109–1116.

[55] Y. Du, P. Xiong, Y. Fan, and X. Li, "Dynamic checking and solution to temporal violations in concurrent workflow processes," *IEEE Transac-*

tions on Systems, Man, and Cybernetics - Part A: Systems and Humans, vol. 41, no. 6, pp. 1166–1181, 2011.

[56] K. Watahiki, F. Ishikawa, and K. Hiraishi, "Formal verification of business processes with temporal and resource constraints," in *Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on*, 2011, pp. 1173–1180.

[57] L. E. Mendoza Morales, C. Monsalve, and M. Villavicencio, "Application of formal methods to verify business processes," in *Formal Methods: Foundations and Applications: 19th Brazilian Symposium, SBMF 2016, Natal, Brazil, November 23-25, 2016, Proceedings*, L. Ribeiro and T. Lecomte, Eds. Cham: Springer International Publishing, 2016, pp. 41–58.

[58] L. Giordano, A. Martelli, M. Spiotta, and D. T. Dupre, "Business process verification with constraint temporal answer set programming," *Theory and Practice of Logic Programming*, vol. 13, no. 4-5, pp. 641–655, 2013.

[59] L. E. Mendoza Morales, "Business process verification: The application of model checking and timed automata," *CLEI Electronic Journal*, vol. 17, no. 2, 2014.

[60] W. M. van der Aalst, A. Hirnschall, and H. Verbeek, "An alternative way to analyze workflow graphs," in *Advanced Information Systems Engineering*. Springer, 2002, pp. 535–552.

[61] L. Hong and Z. J. Bo, "Research on workflow process structure verification," in *e-Business Engineering, 2005. ICEBE 2005. IEEE International Conference on*. IEEE, 2005, pp. 158–165.

[62] H. Lin, Z. Zhao, H. Li, and Z. Chen, "A novel graph reduction algorithm to identify structural conflicts," in *System Sciences, 2002. HICSS. Proceedings of the 35th Annual Hawaii International Conference on*. IEEE, 2002, pp. 10–pp.

[63] G.-W. Kim, J. H. Lee, and J. H. Son, "Classification and analyses of business process anomalies," in *Communication Software and Networks, 2009. ICCSN'09. International Conference on*. IEEE, 2009, pp. 433–437.

[64] E. Börger, O. Sörensen, and B. Thalheim, "On defining the behavior of or-joins in business process models." *Journal of Universal Computer Science*, vol. 15, no. 1, pp. 3–32, 2009.

[65] Signavio, "Following academic signavio," http://www.signavio.com/bpm-academic-initiative/, 2009, accessed: 2017-05-01.

[66] C. Pan, J. Guo, L. Zhu, J. Shi, H. Zhu, and X. Zhou, "Modeling and verification of can bus with application layer using uppaal," *Electronic Notes in Theoretical Computer Science*, vol. 309, pp. 31–49, 2014.

[67] Basic Research in Computer Science at Aalborg University in Denmark and the Department of Information Technology at Uppsala University in Sweden, "Uppaal," http://www.uppaal.org/, 1999, accessed: 2017-05-01.

[68] G. Rodriguez-Navas, J. Proenza, and H. Hansson, "An uppaal model for formal verification of master/slave clock synchronization over the controller area network," in *Proc. of the 6th IEEE International Workshop on Factory Communication Systems, Torino, Italy, IEEE Computer Society Press, Los Alamitos*, 2006.

[69] Bell Labs, "Spin," http://spinroot.com/spin/whatispin.html, 1990, accessed: 2017-05-01.

[70] W. M. Van Der Aalst, "Woflan: a petri-net-based workflow analyzer," *Systems Analysis Modelling Simulation*, vol. 35, no. 3, pp. 345–358, 1999.

[71] H. M. Verbeek, T. Basten, and W. M. van der Aalst, "Diagnosing workflow processes using woflan," *The computer journal*, vol. 44, no. 4, pp. 246–279, 2001.

[72] Eindhoven University of Technology, "Woflan – the workflow analyser," http://www.win.tue.nl/woflan/, 1998, accessed: 2017-05-01.

[73] A. Cimatti, E. Clarke, F. Giunchiglia, and M. Roveri, "Nusmv: A new symbolic model verifier," in *International conference on computer aided verification*. Springer, 1999, pp. 495–499.

[74] A. Cimatti, E. Clarke, F. Giunchiglia, and M. Roveri, "Nusmv: a new symbolic model checker," *International Journal on Software Tools for Technology Transfer*, vol. 2, no. 4, pp. 410–425, 2000.

[75] M. Szpyrka, A. Biernacka, and J. Biernacki, "Methods of translation of petri nets to nusmv language." in *CS&P*, 2014, pp. 245–256.

[76] P. Szwed, "Evaluating efficiency of archimate business processes verification with nusmv," in *Information Technology for Management*. Springer, 2016, pp. 179–196.

[77] R. Cavada, A. Cimatti, M. Dorigatti, A. Griggio, A. Mariotti, A. Micheli, S. Mover, M. Roveri, and S. Tonetta, "The nuxmv symbolic model checker," in *International Conference on Computer Aided Verification*. Springer, 2014, pp. 334–342.

[78] A. Biernacka, J. Biernacki, M. Szpyrka, T. E. Simos, Z. Kalogiratou, and T. Monovasilis, "State-based verification of rtcp-nets with nuxmv," in *AIP Conference Proceedings*, vol. 1702, no. 1. AIP Publishing, 2015, p. 100011.

[79] M. Szpyrka, P. Matyasik, and R. Mrówka, "Alvis – modelling language for concurrent systems," in *Intelligent Decision Systems in Large-Scale Distributed Environments*, ser. Studies in Computational Intelligence, P. Bouvry, H. Gonzalez-Velez, and J. KoÅĆodziej, Eds. Springer-Verlag, 2011, vol. 362, ch. 15, pp. 315–341.

[80] M. Szpyrka, P. Matyasik, and M. Wypych, "Generation of labelled transition systems for alvis models using haskell model representation," in *Proceedings of the 22nd International Workshop on Concurrency, Specification and Programming (CS&P 2013)*, vol. 1032. Warsaw, Poland: CEUR Workshop Proceedings, 2013, pp. 409–420.

[81] M. Szpyrka, P. Matyasik, R. Mrówka, and L. Kotulski, "Formal description of Alvis language with α^0 system layer," *Fundamenta Informaticae*, vol. 129, no. 1-2, pp. 161–176, 2014.

[82] T. Szmuc and M. Szpyrka, "Formal methods – support or scientific decoration in software development," in *Proc. of Mixdes 2015, the 22nd International Conference Mixed Design of Integrated Circuits and Systems*, Torun, Poland, June 25–27 2015, pp. 24–31.

[83] M. Szpyrka, P. Matyasik, J. Biernacki, A. Biernacka, M. Wypych, and L. Kotulski, "Hierarchical communication diagrams," *Computing and Informatics*, vol. 35, no. 1, pp. 55–83, 2016.

[84] M. Wypych, M. Szpyrka, and P. Matyasik, "Extension of Alvis compiler front-end," in *International Conference of Computational Methods in Sciences and Engineering (ICCMSE 2015)*, ser. AIP Conference Proceedings, vol. 1702. Athens, Greece: AIP Publishing, March 20-23 2015, pp. 100 015–1–100 015–4.

[85] M. Szpyrka, G. J. Nalepa, A. Ligęza, and K. Kluza, "Proposal of formal verification of selected BPMN models with Alvis modeling language," in *Intelligent Distributed Computing V. Proceedings of the 5th International Symposium on Intelligent Distributed Computing – IDC 2011, Delft, the Netherlands – October 2011*, ser. Studies in Computational Intelligence, F. M. Brazier, K. Nieuwenhuis, G. Pavlin, M. Warnier, and C. Badica, Eds. Springer-Verlag, 2011, vol. 382, pp. 249–255. [Online]. Available: http://www.springerlink.com/content/m181144037q67271/

[86] K. Kluza, G. J. Nalepa, M. Szpyrka, and A. Ligęza, "Proposal of a hierarchical approach to formal verification of BPMN models using Alvis and XTT2 methods," in *7th Workshop on Knowledge Engineering and Software Engineering (KESE2011) at the Conference of the Spanish Association for Artificial Intelligence (CAEPIA 2011): November 10, 2011, La Laguna (Tenerife), Spain*, J. Canadas, G. J. Nalepa, and J. Baumeister, Eds., 2011, pp. 15–23. [Online]. Available: http://ceur-ws.org/Vol-805/

[87] M. Szpyrka, P. Matyasik, and M. Wypych, "Alvis language with time dependence," in *Proceedings of the Federated Conference on Computer Science and Information Systems*, Krakow, Poland, 2013, pp. 1607–1612.

[88] P. Matyasik, "Alvis virtual machine," in *Proceedings of the Federated Conference on Computer Science and Information Systems*, 2014, pp. 1639–1645.

[89] M. Szpyrka, P. Matyasik, L. Podolski, and M. Wypych, *Simulation of Multi-agent Systems with Alvis Toolkit*. Zakopane, Poland: Springer International Publishing, 2017, pp. 599–608. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-59060-8_54

[90] M. Mach-Król, "Tools for building a temporal knowledge base system supporting organizational creativity," *Procedia Computer Science*, vol. 65, pp. 1031–1037, 2015.

[91] E. Kucharska, K. Grobler-Dębska, J. Gracel, and M. Jagodziński, "Idea of impact of erp-aps-mes systems integration on the effectiveness of decision making process in manufacturing companies," in *International Conference: Beyond Databases, Architectures and Structures*. Springer, 2015, pp. 551–564.

[92] E. Dudek-Dyduch, E. Kucharska, L. Dutkiewicz, and K. Rączka, "Almm solver-a tool for optimization problems," in *International Conference on Artificial Intelligence and Soft Computing*. Springer, 2014, pp. 328–338.

[93] E. Kucharska, K. Grobler-Dębska, and K. Rączka, "Almm-based methods for optimization makespan flow-shop problem with defects," in *Information Systems Architecture and Technology: Proceedings of 37th International Conference on Information Systems Architecture and Technology–ISAT 2016–Part I*. Springer, 2017, pp. 41–53.