UDC 004.94:005.4
Original Scientific Paper

Comparison of Simple Graphical Process Models

Katarina Tomičić-Pupek *katarina.tomicic@foi.hr*
University of Zagreb
Faculty of Organization and Informatics Varaždin

Ivan Vidović *ivvidovic@foi.hr*
University of Zagreb
Faculty of Organization and Informatics Varaždin

Martina Tomičić Furjan *martina.tomicic@foi.hr*
University of Zagreb
Faculty of Organization and Informatics Varaždin

Abstract

Comparing the structure of graphical process models can reveal a number of process variations. Since most contemporary norms for process modelling rely on directed connectivity of objects in the model, connections between objects form sequences which can be translated into performing scenarios. Whereas sequences can be tested for completeness in performing process activities using simulation methods, the similarity or difference in static characteristics of sequences in different model variants are difficult to explore.

The goal of the paper is to test the application of a method for comparison of graphical models by analyzing and comparing static characteristics of process models. Consequently, a meta-model for process models is developed followed by a comparison procedure conducted using a graphical model comparison algorithm.

Keywords: graphical process model, meta-model, comparison

1. Introduction

Business process models represent a specific way of doing business by an organization. These models consist of concepts established by contemporary norms for process modelling. Three currently relevant modelling norms and methods for business process modelling are Business Process Model and Notation 2.0 (BPMN), Unified Modelling Language (UML) activity diagram modelling and Event-Driven Process Chain (EPC) modelling. In this paper we analyze graphical process models in accordance with BPMN 2.0 released by the Object Management Group (OMG) in February 2011 [4].

Business process models are important in developing information systems for organizations [5] since they allow business process experts to reveal knowledge about how business is executed. On the other hand, they imply the functionalities that need to be included in information system applications to ensure that process performance is adequately supported. Process analysis is often annotated to a business process optimization prior to, parallel to or during Information and Communication Technology (ICT) implementation projects. Analyzing processes, developing process models and comparing the so called "As-Is" (that is, the way the processes are actually being performed) and "To-Be" models (that is, models of redesigned processes, typically using ICT support) thus often constitute a crucial step in developing information systems (IS).

A basic question that arises at this stage of process analysis is that of comparing different variants of processes. Two categories of process variants can be distinguished: 1) static variants of process diagrams showing, for example, the "As-Is" and "To-Be" versions of process diagrams made in the process modelling phase, and 2) dynamic variants of processes

are based on the invocation of branches which depend on decisions to be made during process execution. In this paper we will only explore static differentiation by comparison between the "As-Is" version of the process and the "To-Be" version (or variant) in order to determine how the process has been improved or which structural changes have occurred.
Dynamic characteristics of process models can be – and usually are – tested using simulation methods. The authors' experience with over 15 process-related projects has revealed that when applying simulation methods for comparing two graphical process models there is a risk that all process cases will not be covered, or equally covered. Simulation parameters used in simulations result in a different number of instances which are used for testing various process scenarios. Setting appropriate simulation parameters that would cover all process scenarios in all variants of a process model is a sensitive process, which accounts for the aforementioned risk. In order to ensure that simulations of process models are comparable, identical simulation parameters for both aforementioned model variants must be used. Simulation parameters therefore need to cover the model variant with a few potential scenarios. This means that while, on the one hand, parameters may be entirely appropriate for one model, they could, on the other hand, be overstated for a model with fewer scenarios.
In this paper we address that risk by suggesting that process models are compared in terms of their static characteristics or their structure. This approach is adequate since static characteristics of process models do not depend on the number of instances used for testing process scenarios. Instead, they depend on connections of objects represented by concepts and defined by the norm selected for modelling.
A comparison of process model is justified since it is an essential step in identifying which changes have been suggested in other process versions. The scope of changes influences the decision about the acceptance probability and feasibility of new process variants because significant changes in the organization and the investment of ensuring a proper process execution environment means that a lot of resources including time, organizational and financial resources must be allocated to reorganizing the environment and deploying the process.

2. Meta-models and process models

In business processes modelling it is necessary to have a standardized notation for modelling. The main reason for standardization is ensuring that different domain experts can understand and interpret identical models in a similar way. In this paper we reflect on a business processes diagram in accordance with the modelling norm BPMN 2.0. Originally developed by the Business Process Management Initiative (BPMI), it is now maintained by OMG, an organization that has partnered with BPMI since 2005. The current version of the norm is BPMN 2.0 specification issued in January 2011. Its predecessors were the 1.2 specification (issued in January 2009), version 1.1. (issued in January 2008), and the first BPMN version – 1.0 (issued in May 2004). The 1.0 version consisted of 48 elements, while the current version has a total of 116 elements. In this paper we will take into consideration only the key elements needed to show basic relationships between objects that constitute a minimal set of basic elements in process models. We will use them to create simple process models and try to compare their two variants, that is, the "As-Is" and "To-Be" variant. In order to compare their structure we first need to understand the main structure of process models created in accordance with BPMN. For this purpose we can use meta-models.

Meta-modelling can be defined as modelling data about a model. Meta-modelling requires the understanding of concepts, operators, rules and limitations of a notation that may be illustrated in a meta-model. Furthermore, an appropriate meta-modelling notation needs to be selected. Hay [1] states that the issue of notation appropriateness has given rise to extensive debates and emphasizes that different notations have been developed to serve different purposes by different audiences. Seidewitz claims that „by carefully considering a model's relationship to the thing being modelled and to other models derivable from it, we can

understand how to use models to reason about the systems we build and how to use meta-models to specify languages for expressing models" [7].

2.1. Related work

Unlike Sun and Shi, who propose a process meta-model which supports dynamic changes in the workflow process [8], we intend to focus on static characteristics of a process model, i.e. its main elements and relations between them which form a structure of sequentially connected objects.

A relevant version of a meta-model describing BPMN based models can be found in Lodhi et al [3]. Their extended meta-model refers to BPMN as a norm or notation and has a limited set of attributes that can be assigned to flow objects (i.e. activities, events and gateways). In our opinion, a meta-model needs to have more generic basic elements with the option of assigning as many additional attributes as possible that are needed to describe them. In this paper we therefore suggest another version of a meta-model for simple process models which need to be in accordance with BPMN.

2.2. Meta-model of process models in accordance with BPMN 2.0

The main contribution of the model we propose is that it does not differentiate basic flow objects into activities, events and gateways since they all serve a similar purpose in process models, i.e. they represent fixed objects which may be related via connection objects (flows). In our meta-model fixed flow objects can have a more detailed description of model objects with as many attributes for various object types as there are in BPMN 2.0. Applicable attributes assigned to objects can be stated by the attribute name and the attribute value, including, for example, duration for activities, or timer settings for timing events. The description of objects needs to consist of as many relevant attributes as necessary but must be in accordance with the BPMN rules (e.g. duration is an attribute of an activity and cannot be assigned to events). Such clear separation of fixed objects from their attributes allows us to focus on the structure of the relation between process model elements rather than on their description by means of attributes.

In this paper, for illustrating 'the model of the process model creation' we develop a logical data model following the Entity-Relationship (ER) model that uses James Martin's notation. This method was selected on the basis of the authors' own preferences as well as on relevant research. For instance, Rosemann and Green [6] argue for the Entity-Relationship (E-R) approach as the meta-language to be used in designing the meta-data model. An ER diagram is a graphical representation of entities, relationships and attributes. The entities are manifestations in the selected domain, relationships are mutual relations between the entities, and attributes describe the characteristics of the entities that take on certain values in order to describe how basic model elements and their relationships are modelled.
The basic structure of a process model in accordance with BPMN 2.0. consists of five basic categories of elements:
1. Flow Objects (Events, Activities and Gateways)
2. Data (Data Objects, Data Inputs, Data Outputs and Data Stores)
3. Connecting Objects (Sequence Flows, Message Flows, Associations and Data Associations)
4. Swimlanes (Pools and Lanes)
5. Artifacts (Group and Text Annotation).

According to BPMN 2.0. [4], "flow objects are the main graphical elements to define the behaviour of a Business Process" and can therefore be seen as fixed objects forming a structure of a process model. Types of relationships between fixed objects have certain limitations. While some fixed objects may be associated with the same kind of objects (e.g. activities may be related to other activities, gateways related to gateways, etc.), a data object cannot be connected to another data object without the mediation of an activity that allows

operational use of the data object (like input or retrieval). Objects pertain to a certain type and have additional attributes used to explain them.
Connection types between fixed objects are also defined by applied notation via connecting objects. Each connection is defined by its source and target fixed object. Connected objects form sequences. A sequence may include one or more gateways which separate the initial sequence into at least two alternative sequences. The probability of executing a sequence depends on the probability distribution of gateway arches. For the sake of simplicity we distinguish two types of connections or flows: 1) simple sequence or message flows and 2) sequence or message flows with associated data content.
A flow without assigned data content between two fixed objects only indicates the sequence of activities, the occurrence of events or activation of a gateway. A sequence flow with associated data content indicates the activation of objects and data content that is necessary to activate a fixed object. Data content associated to a flow is the result of an activated source fixed object or a prerequisite for the activation of the target fixed object.
Fixed objects can be grouped according to common characteristics by means of grouping with or without any hierarchy. In BPMN a hierarchy is defined as a pool that consists of lanes.

The presented meta-model for a simple business process model contains only basic elements and their relations.

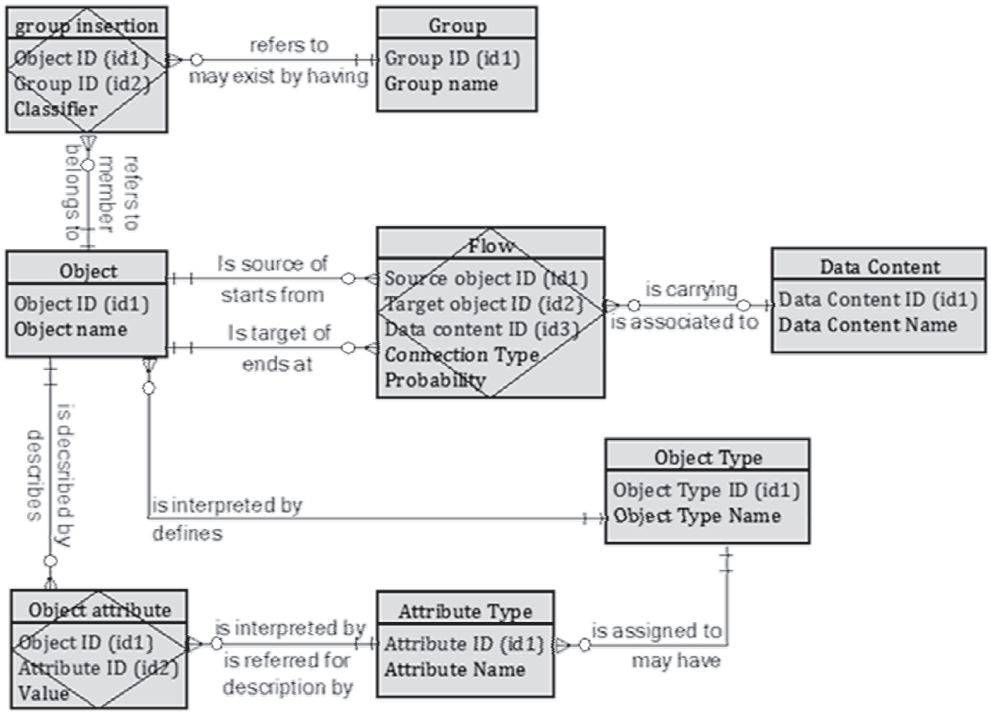

Figure 1: Meta-model of BPMN model consisting of basic model elements

Following the description of basic process model elements, figure 1 shows a meta-model of process model creation using only basic model elements. An object is any fixed object of a certain object type, i.e. an event, activity or a gateway. An object is described by its attributes. It may be a source or a target of a connection that forms a flow. When executing a process, there is a certain probability for a flow to be activated. Data content may be assigned to a flow. An object may belong to one or more groups. Groups may represent pools, lanes or other grouping objects.

The pillars of the static structure of a process model are defined with "Object-Flow" instances. If we equalize instances of "Objects" with vertices and "Flows" with arcs, the data is sufficient to describe, define and draw a directed graph.

Since most graphical models (as suggested by their name) have been derived from graphs we conclude that the pillars of the static structure of analyzed simple process models can be expressed as directed graphs. The translation is simple as most process modelling tools have a feature of listing exact "Object-Flow" instances in an appropriate form (for illustration of "Object-Flow" instances in a process model, see figures 2 and 3 in the following section). This feature enables a comparison of process models using a graph theory-based algorithm.

3. Comparison of process model structure using graph theory-based algorithm

After exploring the basic structure of process models we created process models of a single process in its two variants – the "As-Is" and "To-Be" models. Models are created using the IBM Websphere Business Modeler [2] and are shown in figures 2 and 3. They illustrate a simple process with events, activities, gateway and sequence flows with and without contents. They are represented by a circle, rectangle, rhombus and arches, respectively, with or without text which indicates the associated data content.

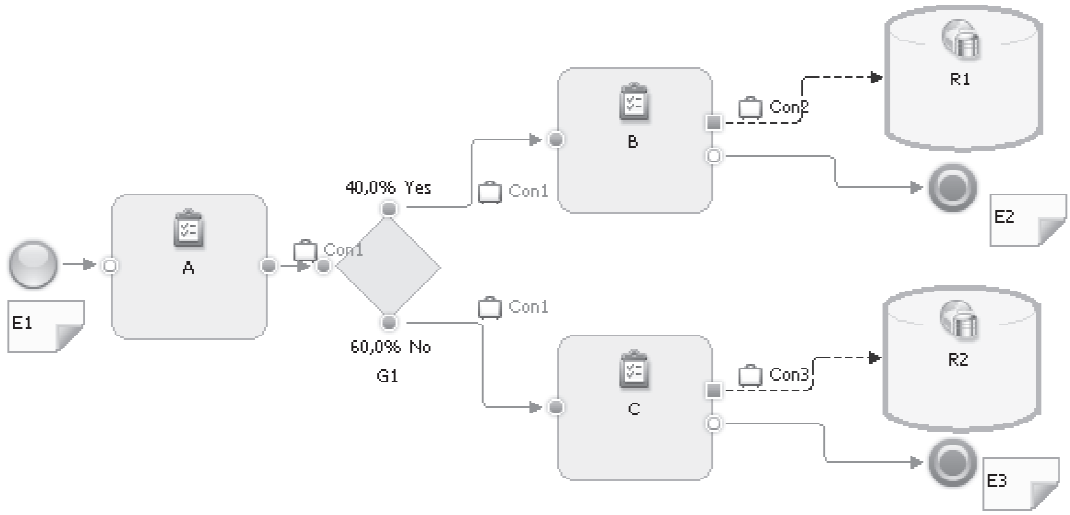

Figure 2: "As-Is" process variant

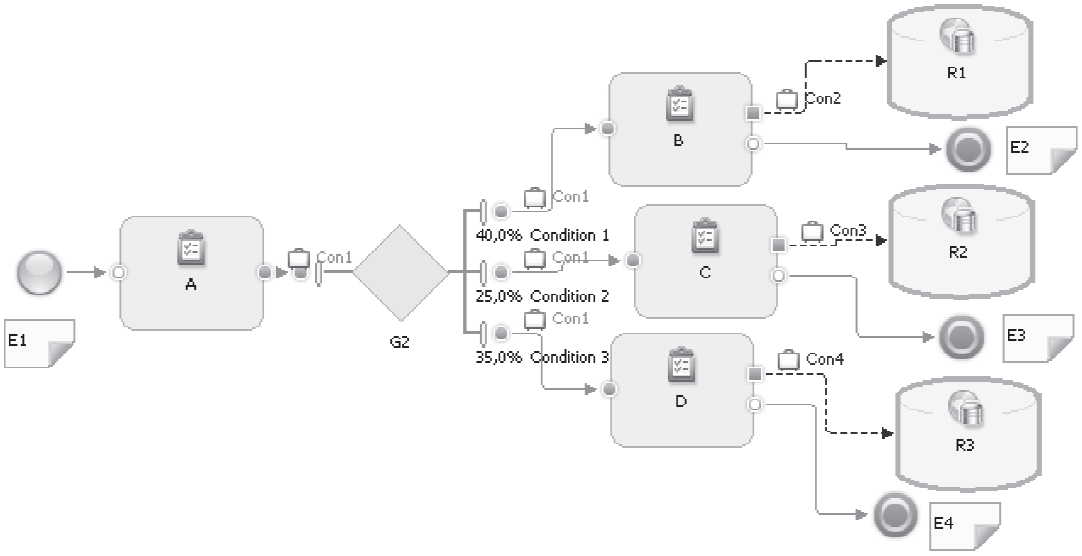

Figure 3: "To-Be" process variant

The demonstrated process models contain main elements and their relations that form a structure of sequentially connected objects which represent process performing scenarios. We assume that this connections structure in form of sequences corresponds with arcs and vertices that form paths in directed graphs. Consequently, we will try to apply a graph theory-based algorithm to compare the two variants of the same process. For that purpose, we will use a graph-theory-based algorithm, the original version of which was developed within the first author's doctoral thesis. To conduct the comparison we first need to translate the graphical model into an appropriate form.

A list of connections in process models that define the static structure of process models is given in tables 1 and 2. They illustrate "Object-Flow" instances defined by the suggested meta-model in figure 1 for process models in figures 2 and 3. The data in tables 1 and 2 need to be coded using catalogues of model elements in tables 3 and 4 in order to enable the comparison of process models using the graph theory-based algorithm.

E1 --> A	No content
A --> G1	Con1
G1 --> B	Con1
G1 --> C	Con1
B --> R1	Con2
B --> E2	No content
C --> R2	Con3
C --> E3	No content

Table 1: List of "Object-Flow" instances for process model in figure 2

E1 --> A	No content
A --> G2	Con1
G2 --> B	Con1
G2 --> C	Con1
G2 --> D	Con1
B --> R1	Con2
B --> E2	No content
C --> R2	Con3
C --> E3	No content
D --> R3	Con4
D --> E4	No content

Table 2: List of "Object-Flow" instances for process model in figure 3

First we have to catalogue model elements (objects of interest). Tables that represent catalogues of fixed objects and flows used in figures 2 and 3 are shown in tables 3 and 4.

Object ID number	Object name
1	A
2	B
3	C
4	G1
5	R1
6	R2
7	E1
8	E2
9	E3
10	D
11	E4
12	R3

Table 3: Fixed object catalogue

Data content ID	Data content
1	No content
2	Con1
3	Con2
4	Con3
5	Con4

Table 4: Data content catalogue

Lists of relationships of objects in the model need to be expressed in the following form: [[source fixed object, target fixed object], flow identification, probability of activating the flow], using catalogues of fixed objects and data content.

Lists that carry information about the relationships in the model by displaying arcs (representing flows in the process model) that connect vertices (representing fixed objects) are given below:

Model variant 1: [[[7,1],1,100], [[1,4],2,100], [[4,2],2,40], [[4,3],2,60], [[2,5],3,100], [[2,8],1,100], [[3,6],4,100], [[3,9],1,100]]

Model variant 2: [[[7,1],1,100], [[1,4],2,100], [[4,2],2,40], [[4,3],2,25], [[4,10],2,35], [[2,5],3,100], [[2,8],1,100], [[3,6],4,100], [[3,9],1,100], [[10,11],1,100], [[10,12],5,100]]

Once the process model is translated into the aforementioned form, a comparison of the two process variants can be performed. The list of object connections (based on the "Object-Flow" instances) is the input into the algorithm that is based on graph theory. The algorithm uses a list of connections, translates them into graphs, creates and compares their adjacency matrices and shows which connections of which graph (or model) is in both adjacency matrices (i.e., at their intersection). Although the algorithm was originally developed for comparing data models, we assume that it will be possible to use the same algorithm to conduct the comparison of example process models. This assumption is based on the recognition that basic rules of connecting elements of a process model are similar enough to rules which apply to connecting vertices by means of arcs in graphs. If our assumption is right, then the algorithm should identify common elements in both process models, as well as those which are different.

The algorithm does not recognize the fourth element of listed connections and will therefore not be able to differentiate connections with the same elements but with different probabilities. We do not consider this as a weakness of the algorithm because the fourth element does not state anything about the structure of models. The fourth element states the probability of a connection being activated (e.g. probability that a person would walk some path does not influence the fact that the path exists). Therefore we do not consider the fourth element relevant for the process model structure comparison.

In further research, despite the fact that the probability of an arc does not influence the structure but only the probability of activation of a sequence part in simulations, the algorithm could be extended if the implementation of the algorithm shows that it is suitable for comparing process models.

The basic idea implemented in the algorithm is to: express flows as lists of vertices and arcs in order to draw directed graphs, derive adjacency matrices, identify semantically similar flow objects from process models selected for comparison (i.e. event E1 has a similar role as a starting event in both models), if needed, perform permutations for re-numerating vertices and extracting adjacency sub-matrices based on semantic similarity, compare sub-matrices and find matching values in matching rows and columns and finally, interpret matching values as structurally similar relationships. This procedure is shown more detailed in a flow chart (figure 4).

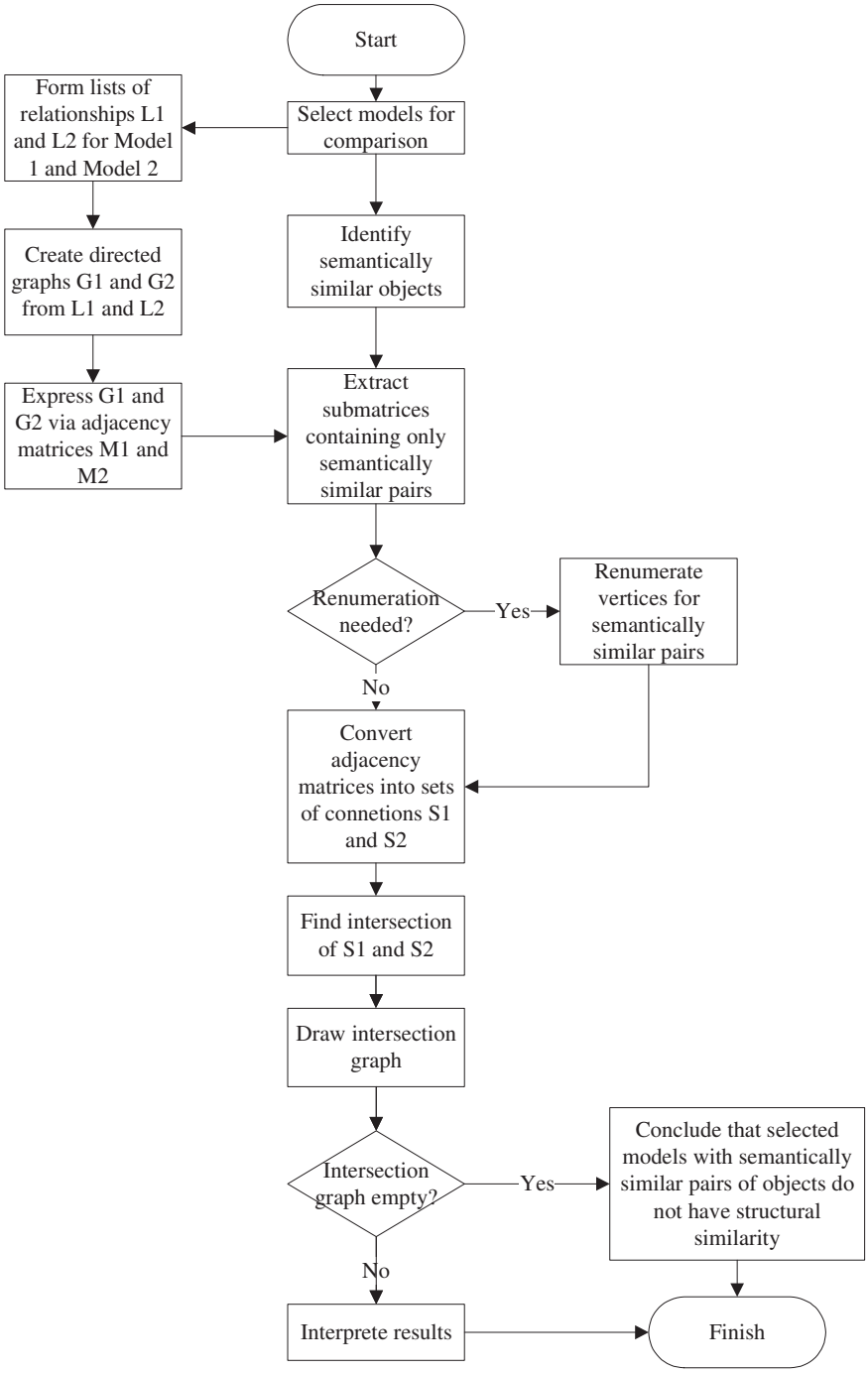

Figure 4: Comparison algorithm flow chart

After applying the algorithm developed in wxMaxima [9] we obtained graphs shown in figure 5 and the following results:

- Arc similarity from 1st graph/process model to 2nd: 100% *meaning that from 8 connections in model "As Is" (represented by 8 arcs in graph 1) all 8 (represented by 8 arcs in the intersection graph) are structurally similar to connections in model "To Be" (8*100%/8=100%)*
- Arc similarity from 2nd graph/process model to 1st: 72,727% *meaning that from 11 connections in model "To Be" (represented by 11 arcs in graph 2) only 8 (represented*

*by 8 arcs in the intersection graph) are structurally similar to connections in model "As Is" (8*100%/11=72,727%)*

- Vertices similarity from 1st graph/process model to 2nd: 100% *meaning that from 9 fixed objects in model "As Is" (represented by 9 vertices in graph 1) all 9 (represented by 9 vertices in the intersection graph) are structurally similar to fixed objects in model "To Be" (9*100%/9=100%)*
- Vertices similarity from 2nd graph/process model to 1st: 75% *meaning that from 12 fixed objects in model "To Be" (represented by 12 vertices in graph 2) only 9 (represented by 9 vertices in the intersection graph) are structurally similar to fixed objects in model "As Is" (9*100%/12=75%)*
- Vertices of 1st graph/process model in intersection: [1,2,3,4,5,6,7,8,9]
- Names of vertices of 1st graph/process model in intersection: [A,B,C,G1,R1,R2,E1,E2,E3]
- Vertices of 1st graph/process model NOT in intersection: []
- Names of vertices of 1st graph/process model NOT in intersection: []
- Vertices of 2nd graph/process model in intersection: [1,2,3,4,5,6,7,8,9]
- Names of vertices of 2nd graph/process model in intersection: [A,B,C,G1,R1,R2,E1,E2,E3]
- Vertices of 2nd graph/process model NOT in intersection: [10,11,12]
- Names of vertices of 2nd graph/process model NOT in intersection: [D,E4,R3].

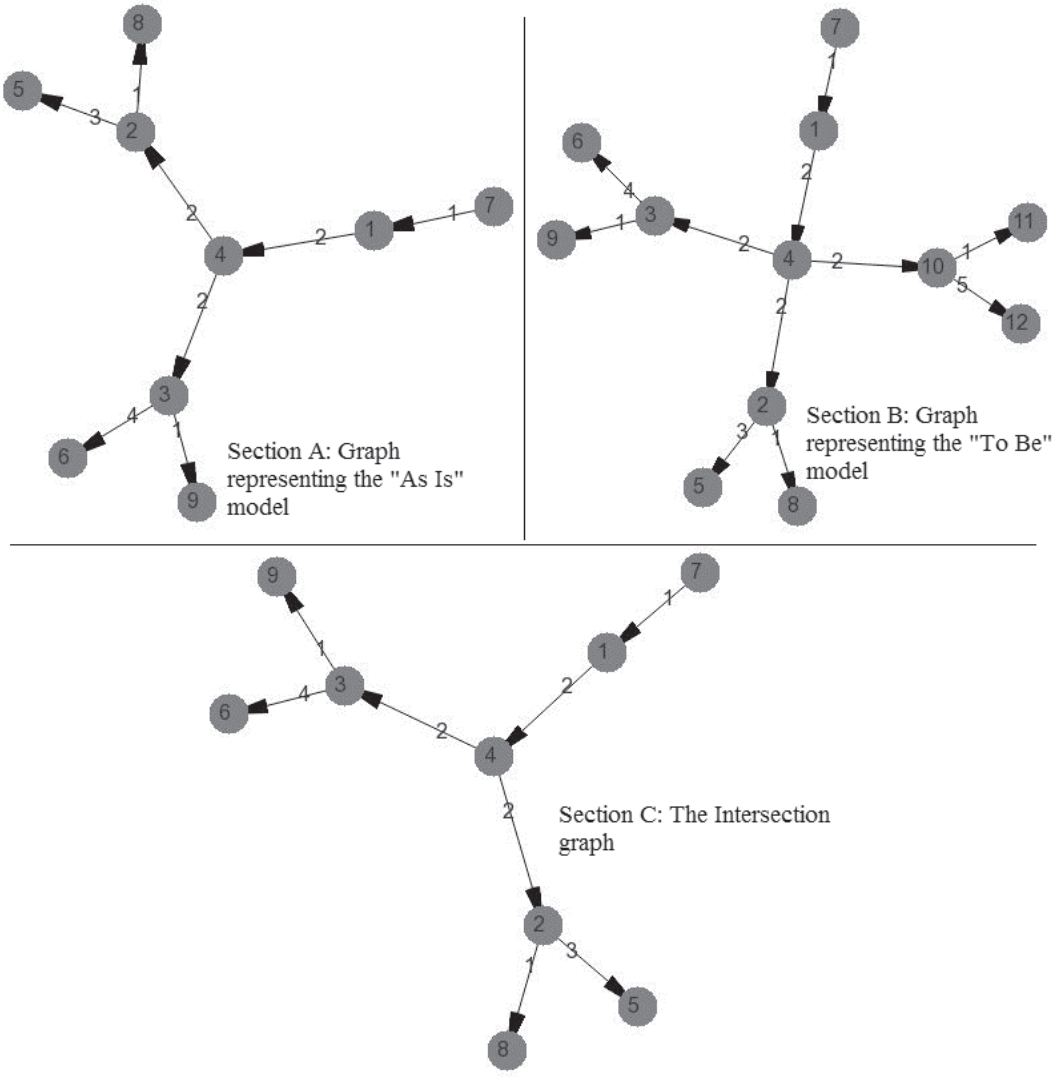

Figure 5: Graphs representing both process variants and the intersection graph

Figure 5 shows directed graphs drawn to represent models "As Is" and "To Be" (sections A and B) and the intersection graph containing only overlapping connections and vertices (section C). Arcs in graphs represent flows or connections in the process model and vertices (circles with numbers) represent fixed objects. Vertices are numerated corresponding with the object identification numbers form table 3 (fixed object catalogue). Numbers associated to arcs correspond with data content identification numbers listed in table 4 (data content catalogue).

The structural similarity of compared graphs (i.e. similarity of models) is measured by arc similarity and similarity of vertices. Arc similarity is calculated as a ratio of common arcs (number of arcs in the intersection) to arcs in each model. The similarity of vertices is calculated as a ratio of common vertices to vertices in each model. The comparison showed that the first model, the "As Is" is 100% similar to the second model, the "To Be" model. This means that all activities and their connections are all preserved in the "To Be" model. On the other hand, there is a 75% resemblance between objects in "To Be" and the "As Is" model, meaning that the "To Be" model contains some other fixed objects (i.e. activities, gateways or events) that represent new functionalities. The resemblance of connections of the "To Be" version of the same process is 72,727% when it is compared to the "As Is" model. A typical practical example that could be described using this case would be introducing a third payment option (represented with activity D) instead of 2 payment options (activities B and C) to customers in a new version of the process.

Since the algorithm was able to recognize the overlapping elements between both models, i.e. it showed which model elements exist in both models and which ones are different in comparison with the other model, we can conclude that the algorithm is suitable for comparing basic process models. We recognize that further research and algorithm upgrades are needed to ensure that the algorithm takes into consideration different probabilities of flows that can influence matching pairs of overlapping objects when comparing process models.

4. Application possibilities and further research

An important issue concerning the testing of process variants is that not all the leading process modelling tools have an option of running simulations for testing various process scenarios. If a tool has the feature for simulating activity sequences, process analysts mostly use simulation tools for testing performance indicators of various process variants. Simulations allow them to compare dynamical performance indicators of processes (like process instances failure or completion rate and probability based outcomes of business objects). This can lead process analysts to: a) overlook the impact of the structure of relations between flow objects while testing new process cases or scenarios and b) fail to check sequences of activities for improvement possibilities. Experience in real process-related projects has shown that the purely performance-improvement-orientation can lead to a risk of suggesting too many changes in the process structure aimed to improve the performance, which then cannot be implemented. With regards to the aforementioned issues, how can the process owner measure the similarity or difference between two (or more) process variants in order to evaluate, predict or decide on the acceptance probability of new process variants?
Comparing and measuring the similarity could influence the evaluation, prediction or decision concerning the acceptance probability of new process variants. If a new version process is "very different" than the current variant in use, then a lot of resources must be allocated to reorganizing the environment and deploying the process. This raises the question of measuring the similarity of process models. The basic suggestion in this paper is to measure similarity by means of the number of semantically and structurally similar objects and their connections. If the similarity is low, more resources need to be allocated to ensure that the new version of a process can be deployed. If the similarity is high, the transition to a new way of executing a process may be more feasible. If the measured similarity is between the upper

and lower limit (which can be set for each organization individually) additional criteria must be taken into consideration.
A tool that could measure and report the scope of change in the structure of relations between model objects by comparing two process models variants of the same process would be useful in those cases. However, no such features are available in leading process modelling tools.

The algorithm we applied allows the comparison of model structure. In other words, it indicates which process elements and connections are similar in compared models and can express the similarity between the two compared models by means of a similarity percentage. In further research we intend to analyze and improve the algorithm so as to enable recognition of paths and sub paths in graphs as entire start-to-end sequences in process models. Additional research in the field of developing more formal constructs for describing process models structure is likely to contribute to the understanding of the significance of process sequences and their modelling.

5. Conclusion

In this article we showed a meta-model for simplified business process models containing basic elements that are needed to describe process model structure. By analyzing the meta-model of a simple process model we established that basic elements of process models can be translated into elements of common directed graphs. This is justified by the fact that most graphical models have been derived from graphs and graph theory. By using an algorithm based on graph theory we also tested the appropriateness of applying a comparison algorithm to comparing two variants of a same process model. The algorithm was tested successfully since it recognized structurally identical and different model objects in the two compared process models. We conclude that more research is needed in the field of applying meta-modelling in identifying other possibly suitable model types for testing the comparison procedure on more complex models. The possibility of extending the comparison procedure in order to take into account the similarity between probabilities assigned to various sequences also represents a challenging research topic.

References

[1] Hay, D.C. *Data model patterns*. Elsevier, San Francisco, USA, 2006.
[2] IBM WebSphere (http://www-01.ibm.com/software/websphere/#, 04.01.2011.)
[3] Lodhi, A; Köppen, V; Saake, G. An Extension of BPMN Meta-model for Evaluation of Business Processes. *Scientific Journal of Riga Technical University Computer Science*, 43: 27-34, 2011.
[4] OMG and BPMN (http://www.omg.org/spec/index.htm, 15.02.2012.)
[5] Phalp, K; Shepperd, M. Quantitative analysis of static models of processes. *The Journal of Systems and Software*, 52:105-112, 2000.
[6] Rosemann, M; Green, P. Developing a meta- model for the Bunge–Wand–Weber ontological constructs. *Information Systems,* 27(2):75-91, 2002.
[7] Seidewitz, E. What models mean. *IEEE Software*, 20(5):26-32, 2003.
[8] Sun, R-Z; Shi, M-L.A Process Meta-Model Supporting Dynamic Change of Workflow. *Journal of Software,* 14(1):62-67, 2003.
[9] wxMaxima (http://andrejv.github.com/wxmaxima/index.html, 10.10.2011.)