DOI: 10.1051/itmconf/20171501002

Decomposition of business process models into reusable sub-diagrams

Piotr Wiśniewski[1,*]

[1]AGH University of Science and Technology, Department of Applied Computer Science, al. Mickiewicza 30, 30-059 Krakow, Poland

Abstract. In this paper, an approach to automatic decomposition of business process models is proposed. According to our method, an existing BPMN diagram is disassembled into reusable parts containing the desired number of elements. Such elements and structure can work as design patterns and be validated by a user in terms of correctness. In the next step, these component models are categorised considering their parameters such as resources used, as well as input and output data. The classified components may be considered a repository of reusable parts, that can be further applied in the design of new models. The proposed technique may play a significant role in facilitating the business process redesign procedure, which is of a great importance regarding engineering and industrial applications.

1 Introduction

Business Process Management stands for a set of techniques which aim to design, analyse, implement and improve organisational processes. Process models can help organisations in visualising processes of the company and give opportunity for optimisation of the process structure. Owing to this, the company may achieve business goals in a more efficient way.

BPMN (Business Process Model and Notation) [1], maintained by the OMG group, is the most widely adopted notation for modelling business processes. BPMN 2.0 is quite complex, as there are many additional documents explaining the notation, such as handbooks and papers devoted exclusively to the pragmatics of the notation usage. In the paper, a brief overview of the guidelines for process modelling as well as various strategies of process design were provided.

According to the research based on industrial applications [2], most of the manually created process models are badly affected by quality issues. Therefore, an automated support in the phase of process design is highly appreciated. The proposed method may be seen as a business process modelling technique based on sub-diagrams generated as a result of a decomposition of existing models. The workflow specification begins with process elements and sub-processes of which the reusable sub-diagrams are created. These component parts are then combined to form a comprehensive process. An overview of the decomposition method is presented in Figure 1.

This paper is organised as follows. In Section 2, an overview of the existing solutions in the area of process model design, as well as their applications, is presented. Section 3 describes in detail the idea of business process modelling and their mathematical representation. The proposed decomposition algorithm is presented in Section 4. Section 5 shows the idea of a component process repository, while in Section 6 the main guidelines of sub-diagrams merge were described.

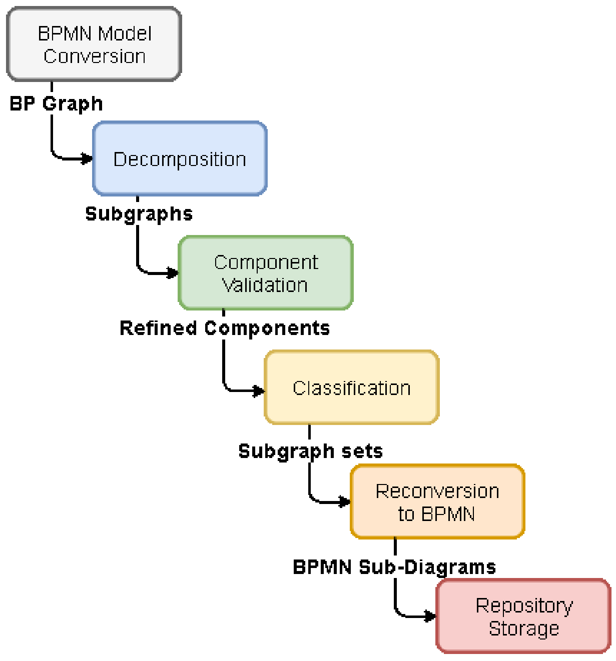

Fig. 1. Overview of the decomposition method.

2 Related Works

Decomposition of business process models is commonly used in the literature to redesign processes or create new models from existing components. One of the approaches consists in identifying maximal repeating fragments, called exact clones, which appear in multiple

[*] Corresponding author: wpiotr@agh.edu.pl

DOI: 10.1051/itmconf/20171501002

process models [3]. This is conducted by identifying SESE (Single Entry, Single Exit) structures in a process tree. Milani et al. [4] discuss criteria and heuristics for model decomposition. They also provide an evaluation regarding the influence of these criteria on the understandability of selected models. The decomposition procedure may be supported by detecting names of non-uniformly specified process element in order to enable a faster creation of consistent complex models [5]. Another solution to the discussed problem was presented in [6] where business processes were presented as a hierarchical ontology of business tasks apt for re-composition. Such a description may simplify the dynamic creation of models of collaborative business processes.

This paper is also related to structured and formal process representations which facilitate process modelling, as well as their reconstruction. One of the user-friendly approaches is the spreadsheet-based representation [7] where the set of activities is listed in a table with specified branches and routing conditions. Process models apt for re-composition can be also described in a declarative specification [8], which focuses on constraints and dependencies between activities instead of determining an explicit sequence flow. They can be also represented as a randomly-ordered list of tasks, along with their input and output conditions, which are then used for a constraint-based process composition [9]. Another approach is a semantic model for BPMN specified in the CSP process algebra [10]. This is especially useful for specifying time constraints between process activities.

Process model repositories are an important tool in the area of business process redesign. They can be developed in a form of a database storing relevant process fragments with a reference to the process metadata [11]. This approach leads to an automated generation of complex models. In case of such a composition, the correctness of the generated process model must be ensured. Therefore, it is necessary to mention the identification and evaluation of issues regarding the flow consistency [12] as well as the layout of the generated business process model [13]. The automated process modelling should also include verification whether the created model contains any typical anomalies such as deadlocks, livelocks or the lack of synchronisation [14] [15] or event anomalies [16]. The analysis of the formal aspects of a BPMN model can be also conducted using Alvis Modelling Language [17].

3 Business Process Models

Highly developed software systems are present in many areas of industrial and business applications, which range from human-machine interface systems, through production management, to web services covering a large domain of implementations. To ensure the reliability and efficiency of such systems and simplify their design and development phases, advanced process engineering approaches are in use.

The engineering process usually starts with business process modelling which constitutes a set of techniques used to represent a certain workflow by providing visual diagrams depicting a business process with its dependencies. Thus, a business process model presents the way operations are performed to accomplish the planned objective of an organisation.

An example field of application for business process models is the specification of Manufacturing Execution Systems (MES) [18]. Building a complete modelling framework for the integration of a system technical and functional model, as well as production process model ensures the understandability of the specification for all the participants involved in the implementation process.

3.1 Business Process Model and Notation

Business process models in BPMN notation are represented by diagrams composed of a limited set of graphical symbols [19]. Visualisation of activity flows allows business users and developers to understand processes more clearly. It is a transparent visual tool dedicated to modelling complex processes.

BPMN model elements can be divided into four major groups: flow objects, connecting objects, swim lanes and artefacts. Flow objects represent the principal elements of the process diagram. In this set, three major object types can be distinguished [1]:
- activities (rounded rectangles) which represent a work that must be performed in the process,
- events (circles) which represent incidents occurring at the time of process execution,
- gateways (rhombi) which control the flow of tokens in the process.

There are two kinds of activities: tasks and sub-processes. A task is an atomic action that can be performed by a person or executed as an automated activity. On the other hand, a sub-process is a compound activity representing a complex work, divisible into smaller parts which are defined as a separate process at the lower level.

The set of events covers three types of elements: start events which indicate the beginning of a process with optional triggering conditions, end events that determine the end of a path in a process or subprocess, as well as intermediate events which may happen at any time within a process.

Gateways control the process flow by splitting it into branches and joining them afterwards. The most typical BPMN gateways are data-based, which means that they evaluate an expression of process data, while they cannot decide regarding the flow of the process [20]. Gateways can be exclusive (XOR) which enable only one outgoing sequence flow based on given process data, inclusive (OR) that precede potentially parallel branches, as well as parallel (AND) used to represent concurrent flows executed without any conditions.

A set of the most typically used BPMN flow objects is presented in Figure 2.

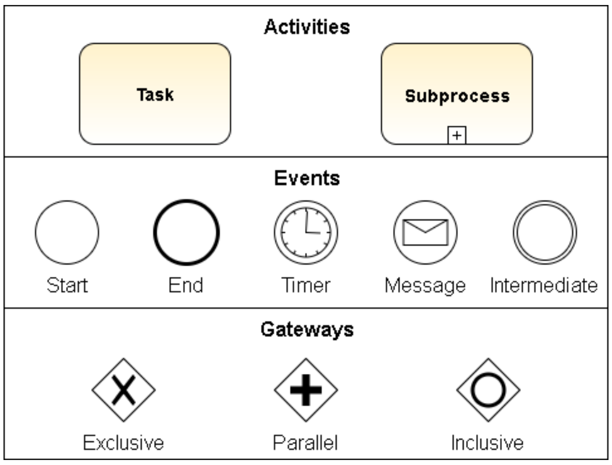

Fig. 2. Set of common BPMN flow objects. Based on [20].

All the flow objects need to be connected using so-called connecting objects of two major types:
- sequence flow which shows the execution order of a process,
- message flow which represents the exchange of messages between process participants.

To distinguish business functions or parts of the system related to certain flow objects, the concept of swimlanes is usually used in BPMN. Business process participants, which can either be entities within an organisation or different collaborators in a process [21] are represented by pools. Sub-partitions of a pool, which represent specific objects or roles, are called lanes. They organise and categorise flow objects, connecting objects and artefacts in a pool. Lanes are present in BPMN diagrams in a form of a rectangle extended either vertically or horizontally along the length of a pool.

Artefacts show additional pieces of information which enable process designers to include more details in a model. Three main types of artefacts were defined in the standard [1]: data objects, groups and annotations.

3.2 Running Example

To illustrate the proposed method, a sample supply process was selected. Its BPMN model is presented in Figure 3. The process goal is to purchase goods in case if the available inventory is below the desired value.

After the start, a warehouse employee checks the inventory: if the available quantity is above the minimal value, the process is ended; otherwise a requester creates a purchase order which is then reviewed by a supervisor. If the created order does not fulfil the requirements, it should be reprocessed by the requester, otherwise the funds are reserved by the accounting department and the supervisor sends the order to the supplier. If the product is not available at the supplier's stock, a negative response is sent to the supervisor and the purchase order is cancelled. Otherwise, the customer receives an invoice and a packing slip, which need to be recorded before releasing the funds and issuing the payment. As soon as the payment is sent to the supplier, the process is completed.

3.3 Mathematical Model of a BPMN diagram

To use the proposed process decomposition method, it is necessary to transform a BPMN model into a form of a graph. Therefore, a formal model of the process should be determined before the conversion. A simple BPMN model can be mathematically represented as a tuple $\mathbb{P} = (\mathbb{O}, \mathbb{F})$ [23] where:
- $\mathbb{O}$ is the set of flow objects, such as: activities (including task and sub-processes), events and gateways,
- $\mathbb{F} \subset \mathbb{O} \times \mathbb{O}$ is the set of sequence flows.

For the decomposition method, a task with boundary events is represented in the formal model as a single flow object having multiple outputs. In addition, the proposed approach includes neither message flows nor artefacts, as the resulting sub-diagrams are generated only for one determined process participant and additional information may differ depending on a specific process model.

3.4 Conversion to a Business Process Graph

Directed graphs are one of the common representation for business process models [24]. Since the proposed decomposition approach considers only the layout of the process model and not its execution, it was possible to generalise the existing model of atomic and complex process graphs. In the proposed structure called business process graph, the existence of loops as well as multiple start and end events is allowed. Therefore, there are no restrictions regarding the cyclicity of the graph and degrees of its vertices. In this case, the resulting graph model may serve as a precise copy of a BPMN diagram.

Based on the formal model presented in Section 3.3, a business process graph can be defined as a connected, directed graph $\mathrm{G_P} = (\mathrm{V_O}, \mathrm{E_F})$ where:
- V_O is a non-empty set of vertices representing all flow objects in a process,
- $\mathrm{E_F}$ is a non-empty set of edges representing all sequence flows in a process.

Transforming the BPMN model into a business process graph consists in listing all process flow objects along with the following properties:
- Object ID and Name (optional),
- Pool/Lane,
- Object Type (task, XOR split gateway, etc.),
- Condition (optional, used for OR and XOR gateways),
- Input/Output Data Entities (optional).

All the objects represented by vertices should be then connected with directed edges which correspond to the sequence flow of the process. A directed edge $e \in E_F$ is a pair:

$$(v_1, v_2) \in V_O \tag{1}$$

where v_1 represents its starting point and v_2 its endpoint. In the business process graph edges contain also optional information about a condition value, provided in case of alternative flows starting with XOR and OR gateways.

DOI: 10.1051/itmconf/20171501002

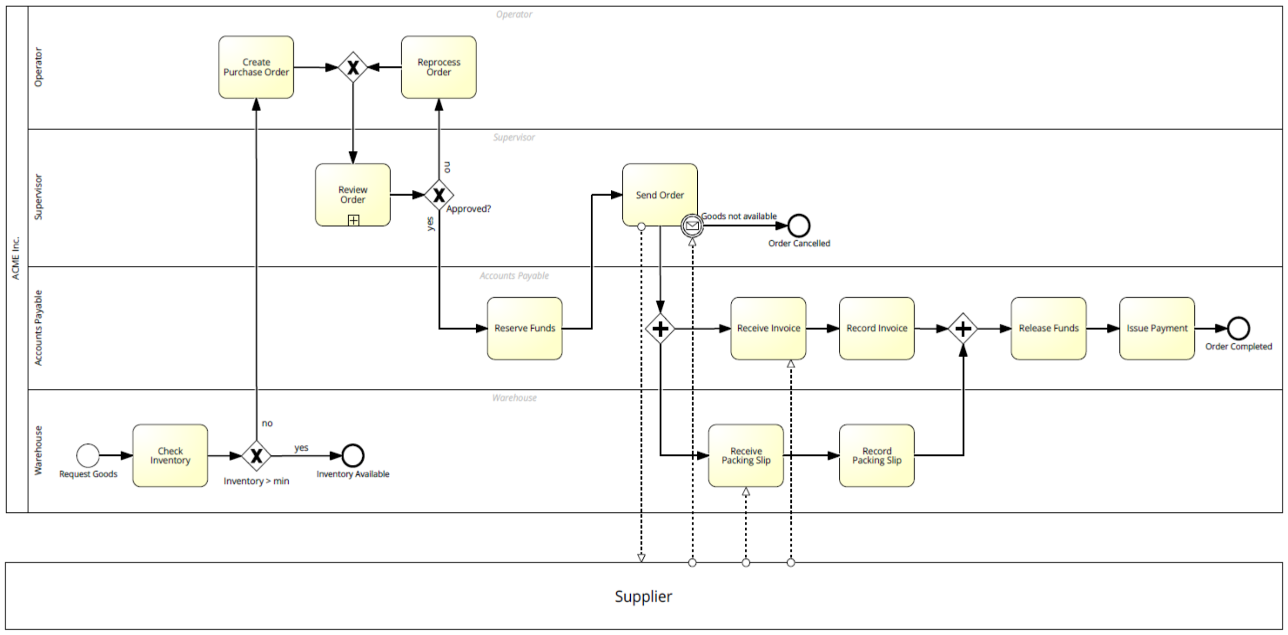

Fig. 3. Sample model of a supply process. Based on [22].

In the next step, it is necessary to define adjacency matrix D [25] of the generated business process graph. Assuming that the analysed process consists of n flow objects, D is a square matrix of size n x n, defined by Formula (2):

$$D = (d_{ij}) \tag{2}$$

where rows and column indexes $i, j = 1..n$ correspond to the IDs of process objects. Each element of matrix D is assigned to a value according to Formula (3):

$$\forall\, v_i, v_j \in V_O \,:\, d_{ij} = \begin{cases} 1, & if\ (v_i, v_j) \in E_F \\ 0, & otherwise \end{cases} \tag{3}$$

Such a matrix stores all the information regarding flow connections between objects in the analysed process.

4 Model Decomposition

The process model decomposition algorithm consists in finding the set S_k of all the subgraphs induced by k vertices, where $2 \le k < n$. In the first step, it is necessary to define appropriate subsets of k vertices. Then, an adjacency matrix of each subgraph is created by deleting all rows and columns from matrix D which do not correspond to vertices for the selected subset. The following subsections present methods for generating sub-diagrams having two and three elements, as well as the concept of a general solution.

4.1 Bi-gram Decomposition

In the most basic example, the process graph is decomposed into all the possible pairs of vertices connected by an edge.

$$S_2 = \left\{ \left(\{v_i, v_j\}, (v_i, v_j) \right) : d_{ij} = 1 \right\} \tag{4}$$

4.2 Tri-gram Decomposition

A more complex case occurs when there is a need to generate sub-diagrams consisting of three process flow objects which can be connected to multiple elements. In such a situation, a component trigram for an object v can be in one of the following forms:
1. v and its two direct predecessors,
2. v with one predecessor and one successor,
3. v and its two direct successors.

Providing the process graph $G_P = (V_O, E_F)$, having n vertices, as well as its incidence matrix D, the decomposition algorithm consists in determining a set S_3 by performing the following operation for each vertex v of graph G_P:
1. Get the sets of all direct predecessors and successors of v, denoted as $Pred(v)$ and $Succ(v)$, respectively.
2. Calculate cardinality of the determined sets, setting n_p as the number of predecessors and n_s as the number of successors.
3. If $n_p > 0$ and $n_s > 0$ then add to set S_3 all subgraphs induced by a set of vertices:

$$\{v_p, v, v_s\} \tag{5}$$

where $v_p \in Pred(v)$ and $v_s \in Succ(v)$.
4. If $n_p \ge 2$ then add to S_3 all subgraphs induced by a set of vertices:

$$\{v_{p1}, v_{p2}, v\} \tag{6}$$

where $v_{p1}, v_{p2} \in Pred(v)$ and $v_{p1} \ne v_{p2}$.
5. If $n_s \ge 2$ then add to S_3 all subgraphs induced by a set of vertices:

$$\{v, v_{s1}, v_{s2}\} \tag{7}$$

where $v_{s1}, v_{s2} \in Succ(v)$ and $v_{s1} \ne v_{s2}$.

In the analysed example presented in Figure 3, 21 flow objects were identified. After running the decomposition algorithm for tri-grams 29 different subgraphs were generated. An example subgraph is shown in Figure 4.

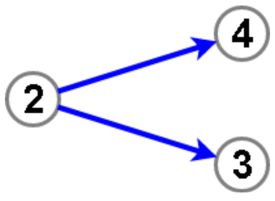

Fig. 4. Three-element subgraph generated as a result of process graph decomposition.

Properties of the flow objects corresponding to vertices of the generated subgraph are listed in Table 1 while Table 2 presents edges included in the subgraph.

Table 1. Properties of flow objects.

Object ID	2	3	4
Pool/Lane	Warehouse		Operator
Object Type	XOR split	end event	task
Object Name	--	Inventory Available	Create PO
Condition	Inventory > min	--	--

Table 2. Properties of sequence flows.

Source Object	2	2
Dest. Object	3	4
Condition Value	yes	no

4.3 K-gram Decomposition

The general problem can be solved by determining for a given number $k < n$, all k-element subsets of V_O and generating connected subgraphs of G_P induced by these subsets. A related solution was presented in [26], however in this paper, a use of constraint programming technique is proposed.

Following input data should be prepared to solve a constraint satisfaction problem:

- decision variable: k-element vector s_k representing a subset of process graph vertices,
- constraints: all elements of s_k are different and connected,
- domain: adjacency matrix of graph G_P.

4.4 Resulting Sub-Diagrams

The generated subgraphs can be retransformed into BPMN models by executing the inversed method presented in Section 3.4 – vertices of each subgraph represent flow objects and non-zero values of matrix D correspond to the sequence flow of the process.

Each of the sub-diagrams can have a different number of inputs and outputs, according to the flow objects from which it is composed. In general, flow objects can be divided into five main groups:

1. Sources: elements that can contain only output sequence flow, such as start events.
2. Sinks: elements that can contain only outputs, such as end events.
3. SESE (Single Entry Single Exit): tasks.
4. SEME (Single Entry Multiple Exit): split gateways, tasks with boundary events, subprocesses.
5. MESE (Multiple Entry Single Exit): merge gateways.

Regarding a best practice of BPMN process modelling [2], implicit splits and joins should be avoided and replaced by gateways. Therefore, if the decomposed model is constructed correctly, all its tasks are SESE elements, except those containing boundary events, as described in Section 3.3. Sub-Diagrams that contain at least one SEME and one MESE object represent the MEME type (Multiple Entry Multiple Exit).

Based on the defined groups, it is possible to determine the number of potential inputs and outputs in each of the generated sub-diagrams, by counting the number of ingoing and outgoing edges of each process flow object and comparing it to its group. This number is used to define functions $\sigma_{I/O}$, described by Formula 8:

$$\sigma_I = \begin{cases} 0, & \textit{if there are no available entries} \\ 1, & \textit{if there is one available entry} \\ 2, & \textit{otherwise} \end{cases} \quad (8)$$

Respectively, function σ_O is defined for exits. Table 3 presents to which group a sub-diagram belongs depending on values of functions σ_I and σ_O.

Figures 5-7 present three sub-diagrams of different groups generated based on the example BPMN model presented in Figure 3:

1. SESE (Figure 5) – one available entry for "Reserve Funds" task and one for "Send Order" task.
2. SEME (Figure 6) – one available entry for split AND gateway and two available exits, one per each task.
3. MEME (Figure 7) – one available entry for "Reprocess Order" task, multiple available entries for join XOR gateway and multiple available exits for "Review Order" subprocess.

Table 3. Sub-Diagram groups depending on functions $\sigma_{I/O}$.

σ_I \ σ_O	**0**	**1**	**2**
0	subprocess	source	source
1	sink	SESE	SEME
2	sink	MESE	MEME

DOI: 10.1051/itmconf/20171501002

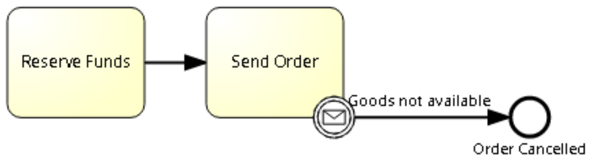

Fig. 5. A SESE sub-diagram.

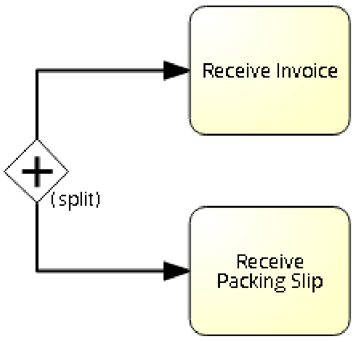

Fig. 6. A SEME sub-diagram.

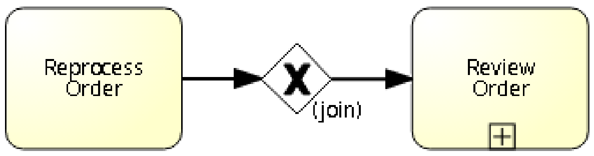

Fig. 7. A MEME sub-diagram.

5 Towards a Sub-Diagram Repository

The decomposition algorithm presented in Section 4 generates all the possible sub-diagrams of k elements. This solution ensures that every possible configuration of process elements is ready to be reused in new process models. However, some of the generated solutions may potentially cause process inconsistencies, as they do not contain enough information about the process flow.

Let us analyse one of the tri-grams generated based on the example software testing process model which is presented in Figure 8. In this case, the resulting process fragment will not be fully reusable because of the AND gateway without any output element assigned. Taking into consideration the fact, that the process flow description contains information about the gateway type, as stated in Section 3.4, such a tri-gram configuration prevents from executing only one activity after the "Sign Agreement" task which limits the component utility. At the same time, the number of branches after the AND gateway is not specified which makes the sub-diagram underdetermined.

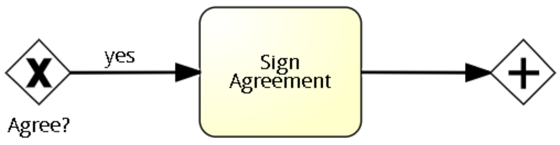

Fig. 8. Tri-gram generated as a result of process model decomposition.

To ensure the correct repository creation without omitting any relevant process sub-diagrams, a user-based component validation was proposed. The operator who performs the decomposition is presented a set of potentially inconsistent component models. The selected sub-diagrams need to satisfy at least one of the following conditions:
1. Presence of a split gateway without any output branch.
2. Presence of a join gateway without any input branch.
3. More than two swimlanes considered.

In the next step, the component processes are classified considering two criteria: the potential number of inputs and outputs, as well as the diagram similarity. The former is used for grouping models according to the values of functions σ_I and σ_O defined in Section 4.4. The latter is performed by applying the vector model technique for business processes [27]. In this case, it is necessary to define three input entities for the algorithm:
- document collection: two sub-diagram graphs $G_1, G_2 \in S_k$,
- set of terms Θ: the union of all graph vertices with their properties listed in Section 3.4,
- weights: for each term $\theta \in \Theta$ a value of 1 is assigned if the corresponding vertex is a part of the selected subgraph and 0 otherwise.

Considering these two criteria, selecting the appropriate sub-diagram consists either in choosing its group, as shown in Table 3 or specific flow objects that should belong to the component. In the second case, the user is shown a set of similar sub-diagrams, out of which a final selection is needed.

6 Merge Guidelines Outline

Composition of the BPMN model using the repository of reusable parts should always be conducted with respect to business process modelling guidelines to avoid typical anomalies [28]. Incoherent business process models may be affected by syntactical anomalies which include incorrect usage of activities, gateways, connecting objects or swimlanes. Another group are structural anomalies, related to the improper dynamic behaviour of the process, for example the presence of deadlocks and infinite loops.

Considering the correctness of the composed process model, it is recommended that its generation is supported by a graphical tool connected to the repository. The interface should suggest recommended connection using the position-based classification technique [29]. An example of a forward model completion is presented in Figure 9.

On the other hand, the system should also prevent the user from creating links between flow objects which may lead to structural anomalies. A potential deadlock is shown in Figure 10.

7 Conclusions

The approach presented in this paper offers a set of activities which support creation of reusable component business process models. The proposed method uses complete BPMN diagrams as input, based on which the component models are generated.

Such a decomposition algorithm may be used to create a component model repository which facilitates process redesign as well as provides a possibility to avoid common modelling anomalies.

In the future work it is planned to include the message flow in the decomposition which will increase the support of the method for various process elements. Another valuable extension would be to allow generation of sub-diagrams representing a specific set of activities, based on their semantic description.

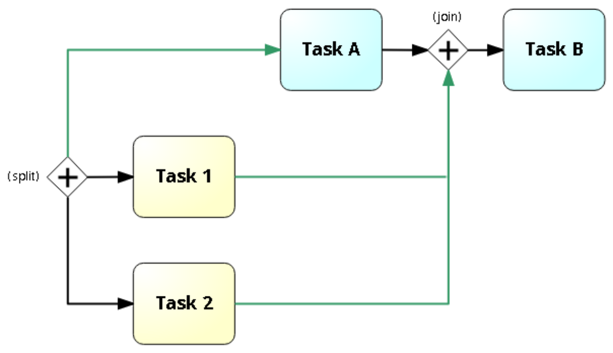

Fig. 9. BP modelling recommendation based on forward model completion technique. Recommended links were marked with green colour.

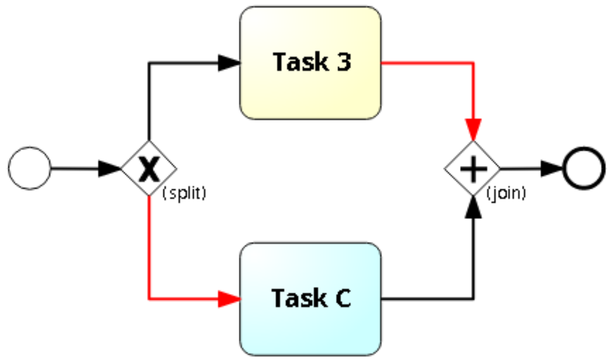

Fig. 10. Forbidden connections between sub-diagrams.

References

1. OMG, *Business Process Model and Notation (BPMN): Version 2.0 specification. Technical Report* (Object Management Group, 2011)
2. H. Leopold, J. Mendling, O. Günther, IEEE Software **33**.4, p. 26-33 (2016)
3. M. Dumas, L. García-Bañuelos, M. La Rosa, R. Uba, Information Systems **38**.4, p. 619-633 (2013)
4. F. Milani et al., Business & Information Systems Engineering **58**.1, p.7-17 (Springer, 2016)
5. A. Koschmider, E. Blanchard, User assistance for business process model decomposition, In: *RCIS*, p.445-454 (IEEE, 2007)
6. R. KL Ko, E. W. Lee, S. G. Lee., IEEE Transactions on Services Computing **5**.2, p. 246-259 (2012)
7. K. Kluza, P. Wiśniewski, Spreadsheet-based business process modelling. In: *2016 Federated Conference on Computer Science and Information Systems* p. 1355-1358 (IEEE, 2016)
8. F. M. Maggi, A. J. Mooij, W. MP van der Aalst, "User-guided discovery of declarative process models." In: *2011 IEEE Symposium on Computational Intelligence and Data Mining*, p.192-199 (2011)
9. P. Wiśniewski, K. Kluza, M. Ślażyński, A. Ligęza, Constraint-Based Composition of Business Process Models. In: BPAI Workshop held in International Conference on Business Process Management (accepted)
10. P. YH Wong, J. Gibbons, Science of Computer Programming **76**.8, p. 633-650 (2011)
11. M. Skouradaki, V. Andrikopoulos, F. Leymann, Representative BPMN 2.0 process model generation from recurring structures, In: *IEEE International Conference on Web Services*, p. 468-475 (IEEE, 2016)
12. A. Burattin, et al., Software & Systems Modelling, p. 1-22 (Springer, 2017)
13. V. Bernstein, P. Soffer, Identifying and quantifying visual layout features of business process models, In: *International Conference on Enterprise, Business-Process and Information Systems Modelling*, p. 200-213 (Springer, 2015)
14. A. Ligęza, T. Potempa, Artificial Intelligence for Knowledge Management with BPMN and Rules, In: *IFIP International Workshop on Artificial Intelligence for Knowledge Management,* p.19-37 (Springer, 2012)
15. A. Suchenia et al., Selected Approaches Towards Taxonomy of Business Process Anomalies. In: *Advances in Business ICT: New Ideas from Ongoing Research*. p. 65-85 (Springer, 2017)
16. A. Suchenia, A. Ligęza, International Journal of Computer Technology & Applications **6**.5, p. 789-797 (2015)
17. M. Szpyrka, G. J. Nalepa, A. Ligeza, K. Kluza, Proposal of Formal Verification of Selected BPMN Models with Alvis Modelling Language. In: *IDC*, p. 249-255 (2011)
18. M. Ricken, B. Vogel-Heuser, Modelling of manufacturing execution systems: An interdisciplinary challenge, *2010 IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, p.1-8 (IEEE, 2010)
19. A. Ligęza, Pomiary Automatyka Robotyka **15**.12, p. 210-212 (2011)
20. B. Silver, B. Richard, *BPMN method and style.* Vol. 2. (Cody-Cassidy Press, 2009)
21. A. Lodhi, V. Küppen, G. Saake, Scientific Journal of Riga Technical University. Computer Sciences **43**.1, p. 27-34 (2011)
22. BPM Initiative Model Collection, http://bpnmai.org/

23. A. Ligęza, Decision Making in Manufacturing and Services **5**.1-2, p. 57-67 (2011)
24. I. M. Weber, *Semantic Methods for Execution-level Business Process Modelling: Modelling Support Through Process Verification and Service Composition.* Vol. 40 (Springer, 2009)
25. K. Ruohonen, *Graph theory*, Graafiteoria lecture notes, (TUT, 2013)
26. P. Vasiliu, P. Tiberiu, "Mircea cel Batran" Naval Academy Scientific Bulletin, **18**.2, p. 419 (2015)
27. B. van Dongen, R. Dijkman, J. Mendling, *Measuring similarity between business process models*, In: Seminal Contributions to Information Systems Engineering, p. 405-419, (Springer, 2013)
28. A. Suchenia, P. Wiśniewski, A. Ligęza, Overview of Verification Tools for Business Process Models. In: *2017 Federated Conference on Computer Science and Information Systems*, p. 303-310 (PTI, 2017)
29. K. Kluza, M. Baran, S. Bobek, G. J. Nalepa, Overview of Recommendation Techniques in Business Process Modelling, In: *Knowledge Engineering and Software Engineering* (Würzburg University, 2013)