sciendo

Hierarchical EPC models – review of model linking concepts

doi:10.2478/mape-2019-0048
Date of submission to the Editor: 06/2018
Date of acceptance by the Editor: 07/2018
MAPE 2019, volume 2, issue 1, pp. 476-486

Bartosz Szczęśniak*
ORCID ID: 0000-0002-9683-4629
Silesian University of Technology, **Poland**

INTRODUCTION

It is commonly claimed that the authors of EPC models are G. Keller, M. Nűttgens and A. W. Scheer. It was in 1992 (Keller et al., 1992) that they proposed a concept according to which a process was to be represented as a sequence of alternating events and functions. The method description they proposed contained general and informal rules for creating EPC models. The works conducted in subsequent years were aimed to make these rules more precise and formal. For flat models, a set of formalised rules was described by Van der Alst in 1999 (Van der Aalst, 1999). Rules applicable to flat and hierarchical models were proposed by Rump in 1999 (Rump, 1999) and by Nüttgens and Rump in 2002 (Nüttgens & Rump, 2002). In the years to come, EPC modelling related problems were addressed in numerous scientific papers, e.g. (Mendling & Nüttgens, 2003, Cuntz & Kindler, 2004, Gabryelczyk, 2006, Gruhn & Laue, 2007a, 2007b, Szczęśniak, 2010). Not only rules of syntax, but also the problem of semantics used in EPC models were discussed in many articles, including (Kindler, 2003, Mendling & Van der Aalst, 2006, 2007, Cuntz & Kindler, 2004).

Despite the multitude of publications addressing the problems of creating and using EPC models, there are still areas of ambiguity, and one of them is linking of EPC models. Various authors have proposed several solutions to this problem. They have been discussed further on in this paper. What has also been described is how they are used with reference to the four cases in question, illustrating different possible links between processes.

RULES FOR CREATING FLAT EPC MODELS

A flat EPC model is a sequence of alternating events and functions, between which there may also be logical operators. The functions determine the actions which must be performed in the process described. The events determine the current state of the process. Based on the formalised rules concerning the EPC model syntax, as presented in the literature of the subject (Rump 1999, Van der Aalst W.M.P., 1999, Mendling & Nüttgens, 2003), one can establish the following set of rules that must be met when creating such a model:

1. The elements of a flat EPC model are functions, events and logical operators
2. A model must have at least one function

* bartosz.szczesniak@polsl.pl

3. All elements are interlinked by means of arcs
4. A start event is one before which no other event occurs in the model
5. An end event is one which is not followed by any other event in the model
6. Every process path must start with an event
7. Every process path must end with an event
8. A model must comprise at least one start event and at least one end event
9. Events may have:
   - one incoming arc and one outgoing arc
   - only one outgoing arc (for start events only)
   - only one incoming arc (for end events only)
10. Functions have one incoming arc and one outgoing arc
11. By means of arcs, functions can only be linked to events
12. By means of arcs, events can only be linked to functions
13. Links between functions and events may be direct or indirect, the latter using logical operators
14. Logical operators can be divided into:
    - splitting operators – having one incoming arc and several outgoing arcs
    - joining operators – having one outgoing arc and several incoming arcs
15. A diagram cannot have a loop consisting of logical operators only
16. Neither splitting operator OR nor splitting operator XOR can be used after an event.

Since a process interface is an element only used to interlink two diagrams, this element has been disregarded when discussing the rules for creating flat EPC models, but it has been considered with regard to hierarchical models.

RULES FOR CREATING HIERARCHICAL EPC MODELS

Hierarchical EPC models are models linked to other EPC models in a specific manner. Such links may be related to adding details of selected functions of the output model or determining further actions that should be performed upon completion of the operations provided for therein. With regard to hierarchical EPC models, two semantically different approaches can be distinguished (Thomas et al., 2018). One assumes that process linking is to determine the points where the control flow is transferred from one process to another. The other assumes that process linking should allow for a single large and holistic EPC model to be created on the basis of linked partial models. The considerations provided further on in this paper concern the first approach.

Based on literature analysis, four concepts that can be applied when creating hierarchical models have been identified. These concepts are referred to as C1, C2, C3 and C4. Emphasis has mainly been placed on the problem of linking equivalent EPC models, and consequently, individual concepts have been discussed by disregarding the principles according to which functions are specified in detail.

Concept C1

The most complete set of rules was provided in the concept developed by Laue and Gruhn based on the formal rules previously proposed (Rump 1999, Nüttgens & Rump 2002). This concept has been designated as C1. The C1 concept's rules of creation are as follows:

1. Next to functions, events and logical operators, there are also process interfaces in a model
2. A process interface either has exactly one outgoing arc (start process interface) or has exactly one incoming arc (end process interface)
3. A process interface can only be linked to events
4. A process interface's link to an event or events can only be direct, or it can be established by means of logical operators
5. Every path in a model must start with an event or a start process interface
6. Every path in a model must end with an event or an end process interface
7. Every process interface is assigned exactly one EPC model
8. Every end process interface triggers the EPC model assigned to it
9. If model EPC_A contains an end process interface which triggers model EPC_B, then model EPC_B must contain at least one start process interface pointing at model EPC_A
10. A process interface linked directly to one event is a type I process interface
11. If model EPC_A contains a type I end process interface which triggers model EPC_B, and the end process interface is preceded by event Ec1, then model EPC_B must contain a type I start process interface pointing at model EPC_A, and the start process interface is followed by the same event Ev1
12. If model EPC_B contains a type I start process interface pointing at model EPC_B, and the interface is followed by event Ec1, then model EPC_A must contain a type I end process interface pointing at model EPC_B, and the end process interface is preceded by the same event Ev1
13. A process interface linked to events by means of logical operators is a type II interface
14. If model EPC_C contains a type II end process interface which triggers model EPC_D, then model EPC_D must contain a type II start process interface pointing at model EPC_C
15. In model EPC_C, the preceding events, the type II end process interface pointing at model EPC_D along with the linking logical operators determine the logic of transfer of the control flow, further referred to as *transfer logic.* The transfer logic must be duplicated in model EPC_D being triggered
16. In model EPC_D, the events which belong to the transfer logic must be linked to the start process interface pointing at model EPC_C by means of a flow branch being a mirror image of the link contained in the transfer logic
17. If a group of EPC models is interlinked through mutual triggering of models, the former must contain at least one start event not linked with a start process interface and at least one end event not linked with an end process interface

Concept C2

The second EPC model linking concept has been identified with reference to (Mendling & Nüttgens, 2003, Thomas et al., 2018), and its principles of creation are as follows:

1. Next to functions, events and logical operators, there are also process interfaces in a model
2. A process interface either has exactly one outgoing arc (start process interface) or has exactly one incoming arc (end process interface)

3. A process interface can only be linked to an event
4. Every end process interface is assigned exactly one EPC model
5. A sequence of models created through successive assignments cannot form a cycle
6. If model EPC_A contains an end process interface which is preceded by event Ev1 and assigned EPC_B, then model EPC_B must contain a start process interface followed by event Ev1
7. If model EPC_B contains a start process interface which is followed by event Ev1, then there must be an EPC model containing an end process interface preceded by event Ev1.

Concept C3

The third model linking concept proposed has been developed on the basis of an alternative linking concept described in the literature of the subject (Szczęśniak 2013). This concept is marked as C3, and the rules defined for it are as follows:

1. Next to functions, events and logical operators, there are also process interfaces in a model
2. A process interface may have:
   - exactly one incoming arc, when it is an end process interface
   - exactly one outgoing arc, when it is a start process interface
3. Every process interface points at exactly one EPC model
4. An end process interface must be preceded by exactly one event. This event must be linked to a process interface via no logical operators
5. A start process interface must be followed by exactly one event. This event must be linked to a process interface via no logical operators
6. If model EPC_A contains an end process interface which is preceded by event Ev1 and which points at model EPC_B, then model EPC_B must contain a start process interface which is linked to event Ev1 and which points at model EPC_A
7. If model EPC_B contains a start process interface which is linked to event Ev1 and which points at model EPC_A, then model EPC_A must contain an end process interface which is preceded by event Ev1 and which points at model EPC_B
8. If an end process interface is linked to an event which precedes it or to several events by means of logical operators, then an apparent function and an apparent event should be added directly before the process interface (a sample apparent function may be e.g. “Send “Start” signal for process X”).
9. If a loop is formed by a group of EPC models on account of the links formed by means of process interfaces, it must feature at least one start event not linked to a process interface and one end event not linked to a process interface.

Concept C4

The fourth of the concepts, designated as C4, has been developed on the basis of the formal rules proposed in the literature of the subject (Nüttgens & Rump, 2002), however, it excludes the start process interfaces proposed but not required under these rules. Additionally, what it takes into account is the liberalisation of the prohibition on forming loops by means of interlinked models, as proposed by Laue and Gruhn (Gruhn & Laue, 2007). The rules proposed under concept C4 are as follows:

1. Next to functions, events and logical operators, there are also end process interfaces in a model
2. There is exactly one incoming arc in an end process interface
3. An end process interface points at exactly one EPC model
4. An end process interface can only be linked to events
5. A process interface's link to an event or events can only be direct, or it can be established by means of logical operators
6. If there is an end process interface pointing at model EPC_B in model EPC_A, then the preceding events must belong to a set of start events of model EPC_B
7. If a loop is formed by a group of EPC models on account of the links established by means of process interfaces, it must contain at least one start event which does not belong to the set of events preceding process interfaces in any of the models forming the loop
8. If a loop is formed by a group of EPC models on account of the links formed by means of process interfaces, it must contain at least one end event not linked to a process interface.

USING THE IDENTIFIED CONCEPTS FOR LINKING OF PROCESS MODELS

All the aforementioned concepts have been used to link EPC models with reference to four cases representing diverse levels of complexity and the potential interconnections which may appear when processes are being linked.

Case 1

This is the simplest case. It features two processes, Pr01 and Pr02. Process Pr01 starts when event 1_E01 takes place. In this process, function 1_F01 is executed, followed by event 1_E02 which completes the process. When process Pr01 is complete, process Pr02 is started. Only function 1_F02 is executed in this process, followed by event 1_E03. Figure 1 demonstrates how individual concepts are applied for case 1.

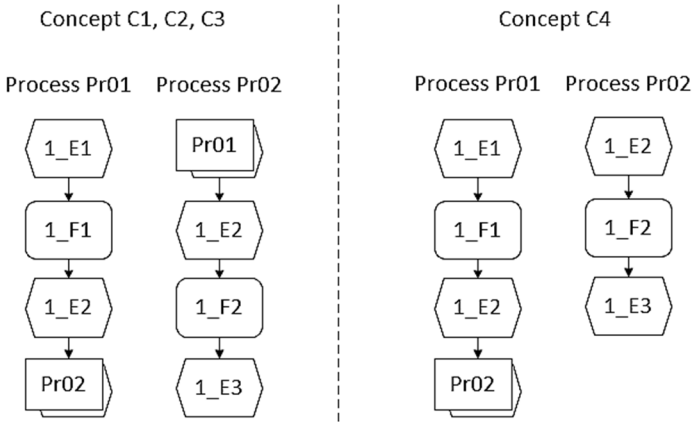

Fig. 1 Using concepts C1, C2, C3 and C4 for model linking in case 1

Case 2

There are two processes, Pr03 and Pr04, in this case. Process Pr03 starts when event 2_E1 takes place. The first function to be executed is 2_F1, after which one of the two events, 2_E2 or 2_E3, can take place. When event 2_E2 takes place, function 2_F2

is executed, followed by event 2_E4. When event 2_E3 takes place, function 2_F3 is executed, and it is followed by event 2_E5. After one of the two events, 2_E4 or 2_E5, takes place, process Pr04 is triggered. In this process, function 2_F4 is executed, followed by event 2_E6. Figures 2 and 3 show how individual concepts have been used in case 2.

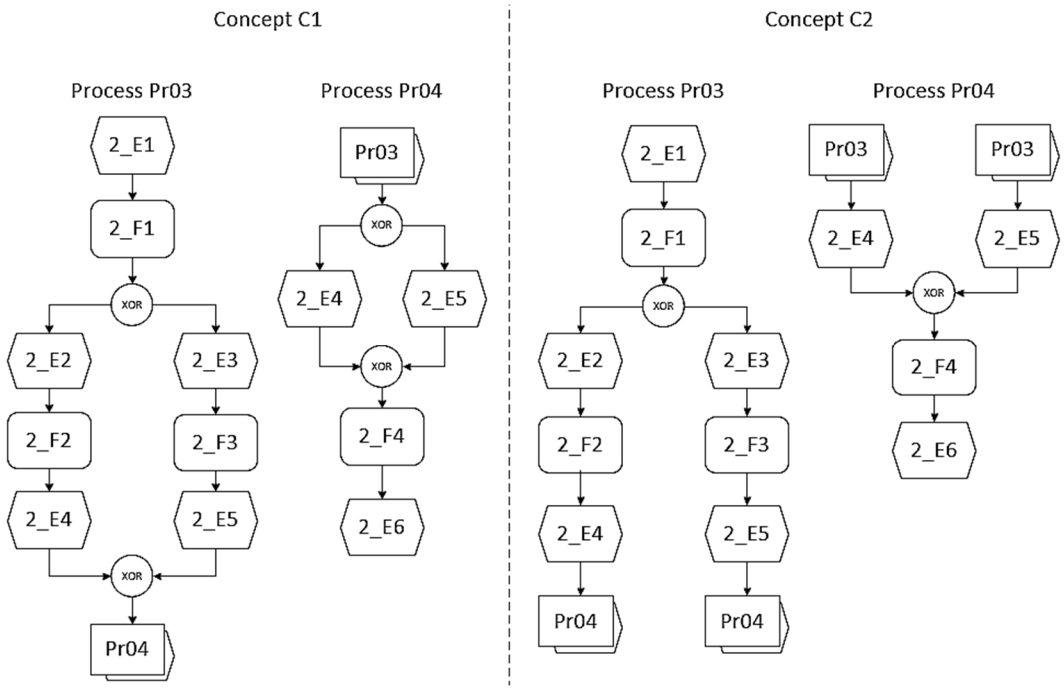

Fig. 2 Using concepts C1 and C2 for model linking in case 2

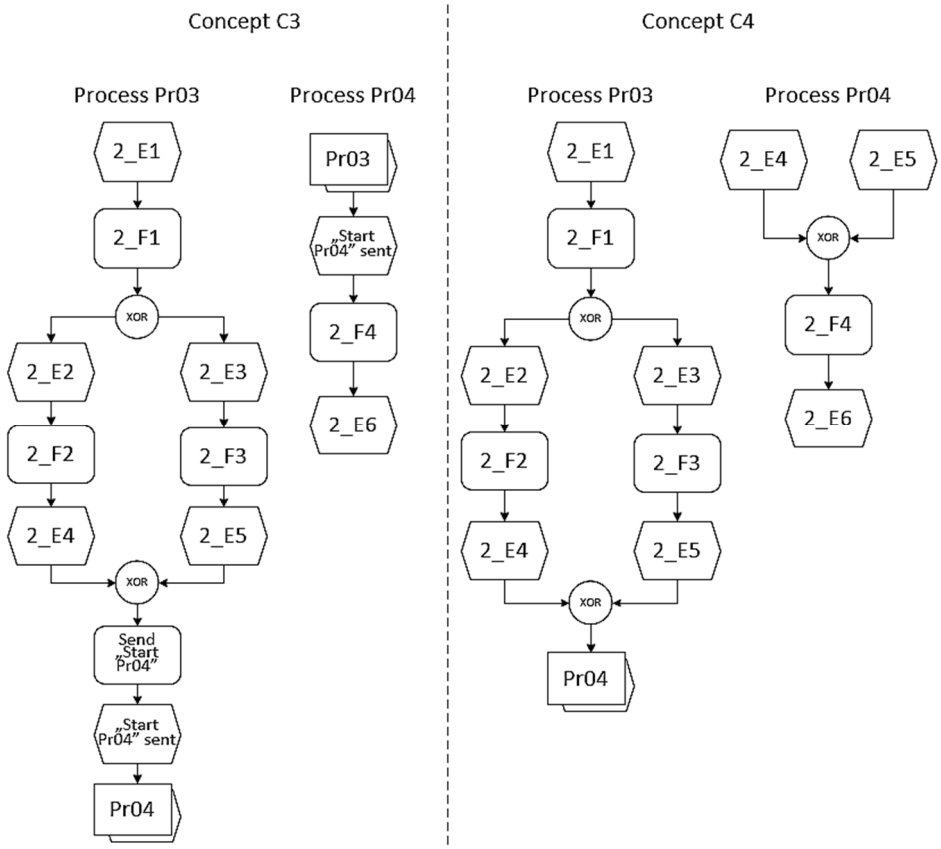

Fig. 3 Using concepts C3 and C4 for model linking in case 2

Case 3

This case comprises three processes: Pr05, Pr06 and Pr07. Process Pr05 starts when event 3_E1 takes place. In this process, function 3_F1 is executed, followed by event 3_E2. Once event 3_E2 has taken place, processes Pr06 and Pr07 are triggered simultaneously. In process Pr06, function 3_F3 is executed, followed by event 3_E3. In process Pr07, function 3_F4 is executed, followed by event 3_E4. Figures 4 and 5 illustrate the way in which individual concepts have been used in case 3.

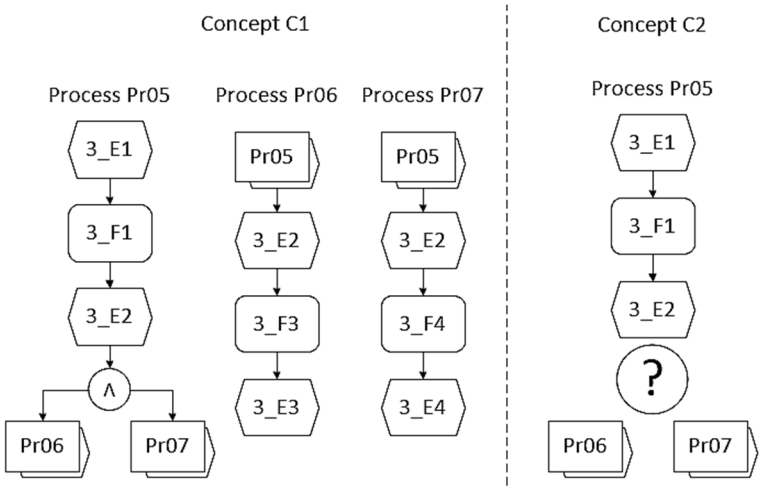

Fig. 4 Using concepts C1 and C2 for model linking in case 3

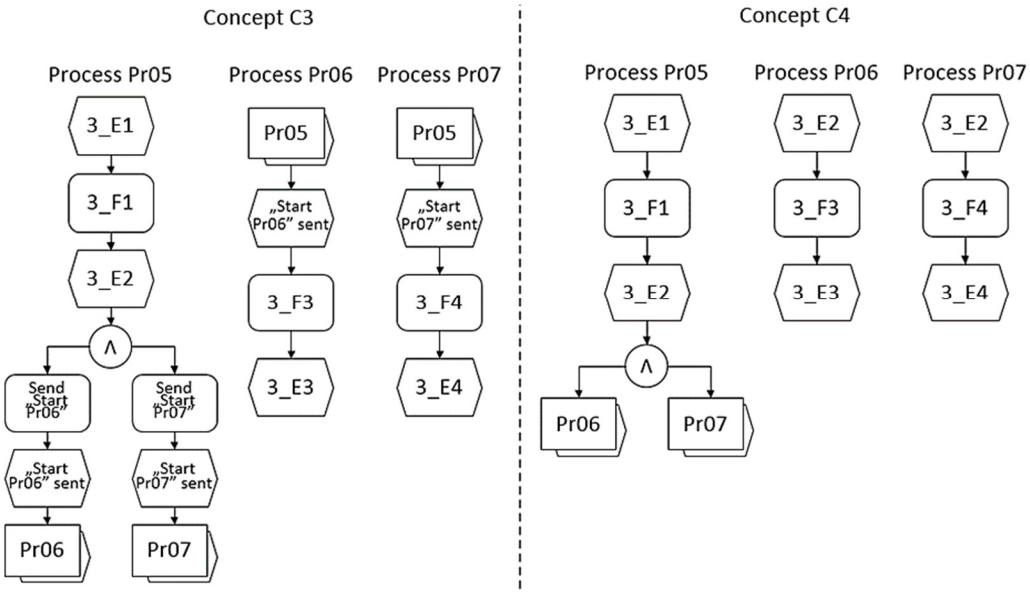

Fig. 5 Using concepts C3 and C4 for model linking in case 3

Case 4

Two processes: Pr08 and Pr09, take place in this case. Process Pr08 starts after event 4_E1 has taken place. Functions 4_F1 and 4_F2 are executed in parallel in this process. Upon completion of function 4_F1, one of the events 4_E2 or 4_E3 takes place. Event 4_E4 takes place after function 4_F2 has been completed. After event 4_E4 and one the events 4_E2 or 4_E3 takes place, process Pr09 is started. If event 4_E2 took place upon triggering this process, function 4_F3 is executed, followed by

event 4_E5. If event 4_E3 took place upon triggering the process, then function 4_F4 is executed, followed by event 4_E6. After one of the two events 4_E5 or 4_E6 and event 4_E4 have taken place, function 4_F5 is executed, followed by event 4_E7. Figures 6 and 7 show how individual concepts have been used in case 4.

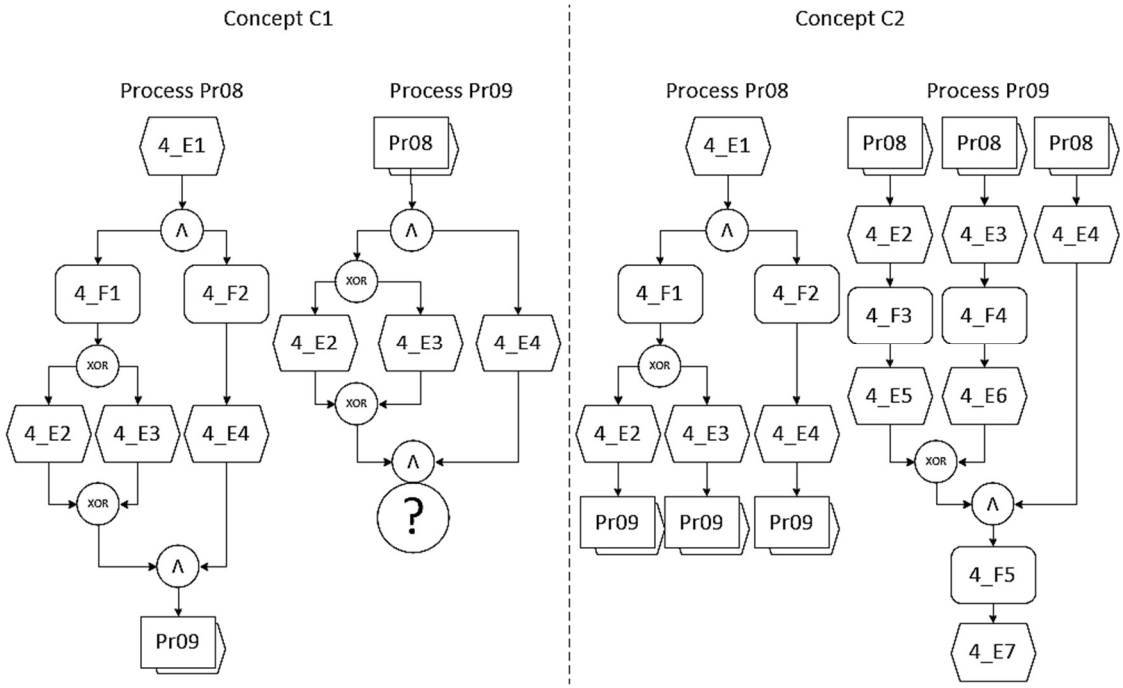

Fig. 6 Using concepts C1 and C2 for model linking in case 4

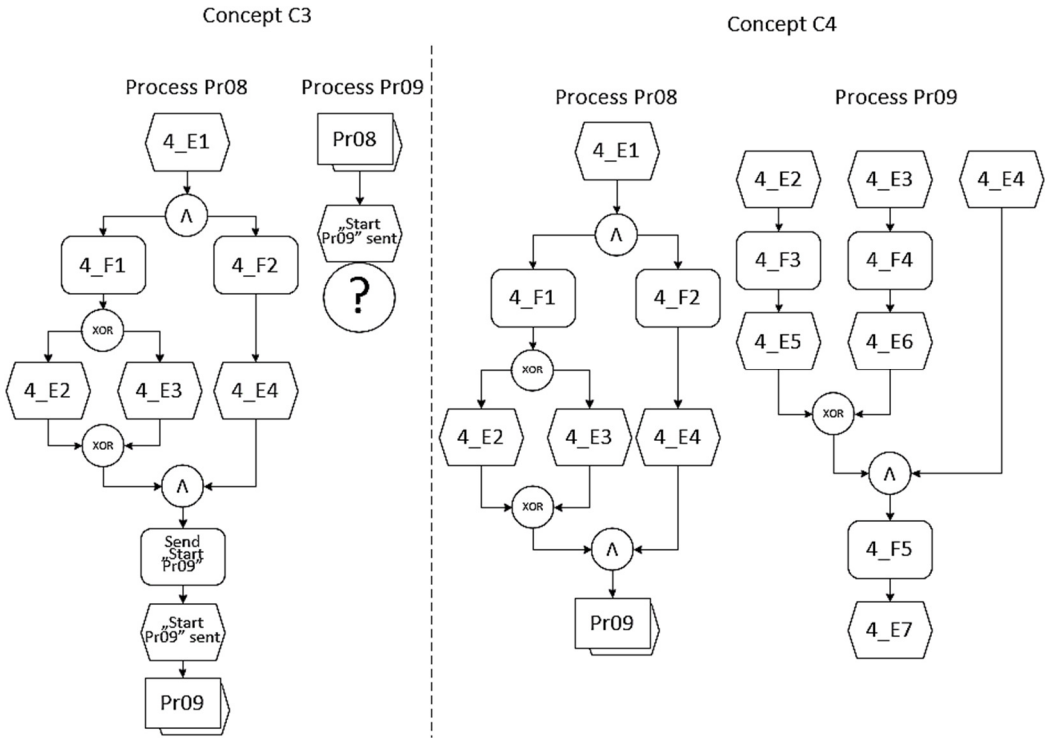

Fig. 7 Using concepts C3 and C4 for model linking in case 4

CONCLUSIONS

When using the aforementioned concepts for linking of EPC models with reference to the four cases in question, one must tackle situations which raise doubts. One of such situations is that of using concept C2 in case 3. This is when linking of models is impossible, since the end process interface must be linked to exactly one event, and this link must be direct, i.e. without using any logical operators. Problems also occur when using concepts C1 and C3 in case 4. According to concept C1, the transfer logic must be duplicated in the model being triggered. Its duplication is based on an assumption that start events are linked is such a way as to obtain a single path starting the model's executable part. Consequently, regardless of the combination of the start events taking place, the process described using the model being triggered starts identically. The same issue emerges when concept C3 is applied. In this case, adding an apparent function and an apparent event before an end process interface causes that in the model triggered after the start process interface only one apparent event takes place, assuming only one way to start execution of the process described by means of the model being triggered.

Therefore, one can propose concept C1a, based on concept C1, where it is not necessary to duplicate the transfer logic in an exact manner in the model being triggered. Using the modified concept designated as C1a for case 4 has been illustrated in Figure 8.

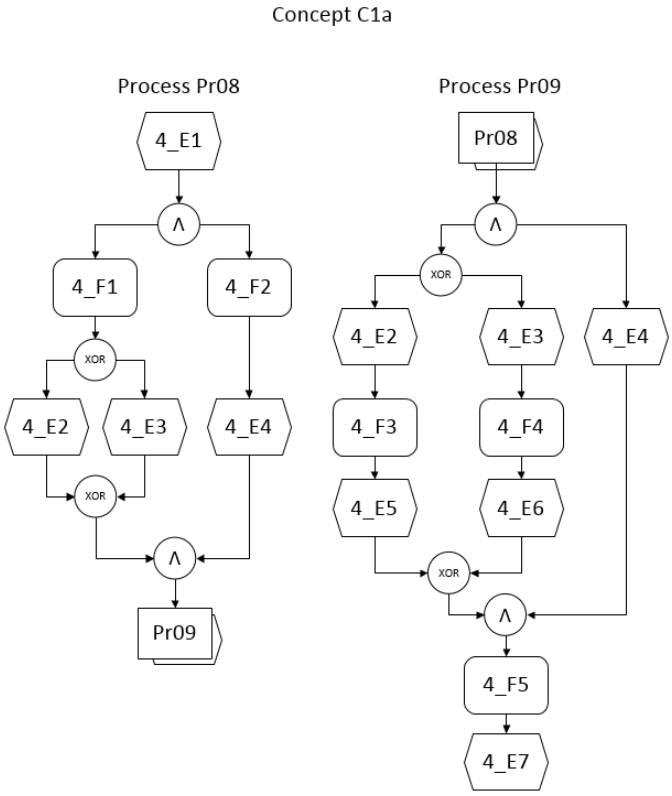

Fig. 8 Using concepts C1a for model linking in case 4

Besides concept C4, the modified C1a concept can also be applied without any issues in all the aforementioned cases.

REFEENCES

Cuntz, N. and Kindler, E. (2004). On the semantics of EPCs: Efficient calculation and simulation, In: M. Nüttgen, F. J. Rump, ed., EPK 2004 – Geschäft sprozess management mit Ereignis gesteuerten Prozess ketten. Proceedings des GI-Workshops und Arbeit's kreistr effens, pp. 7-26.

Gabryelczyk, R. (2006). ARIS w dokumentowaniu procesów biznesu. Warszawa: Diffin.

Gruhn, V. and Laue, R. (2007a). What business process modelers can learn from programmers. Science of Computer Programming, Volume 65(1), pp. 4-13.

Gruhn V. and Laue, R. (2007b). Forderungen an hierarchische EPK-Schemata, EPK CEUR Workshop Proceedings, Volume 303, pp. 59-76.

Keller, G., Nűttgens, M. and Scheer, A.W. (1992). Semantische Prozeβ Modellierung auf der Grundlage "Ereignisgesteuerter Prozeβketten (EPK)". In: A. W. Scheer, ed., Veröffentlichungen des Instituts fűr Wirtschaftsinformatik Saarbrűcken, Volume 89.

Kindler, E. (2003). On the semantics of EPCs: A framework for resolving the vicious circle. Technical Report, Reihe Informatik. Paderborn: University of Paderborn.

Mendling, J., Nüttgens, M. (2003). EPC Modelling based on Implicit Arc Types. In: M. Godlevsky, S. W. Liddle, H. C. Mayr, ed., Information Systems Technology and its Applications, International Conference ISTA'2003, June 19-21, Kharkiv, Ukraine, Proceedings. LNI 30 GI 2003, pp. 131-142.

Mendling, J. and Van der Aalst W.M.P. (2006). Towards EPC Semantics based on state and context. In: M. Nüttgens, F. J. Rump, J. Mendling, ed., Proc. of the 5th GI Workshop on Event-Driven Process Chains (EPK 2006), Vienna, Austria, pp. 25-48.

Mendling, J. and Van der Aalst W.M.P. (2007). Formalization and verification of EPCs with OR-joins based on state and context. NTNU Advanced Information Systems Engineering. Proceedings Book Series: Lecture Notes in Computer Science, Volume 4495, Trondheim, pp. 439-453.

Nüttgens, M. and Rump, F.J. (2002). Syntax und Semantik Ereignisgesteuerter Prozessketten (EPK), In: J. Desel, M. Weske, ed., Promise 2002 – Prozessorientierte Methoden und Werkzeuge für die Entwicklung von Informationssystemen. Proceedings des GI-Workshops und Fachgruppentreffens, Volume P-21, Bonn, pp. 64-77.

Rump, F. (1999). Geschäft sprozeß management auf der Basis ereignisgesteuerter Prozeß ketten - Formalisierung, Analyse und Ausführung von EPKs, Stuttgart: Teubner.

Szczęśniak, B. (2010). Syntax errors in the EPC diagrams of the integrated management system documents. Scientific Journals Maritime University of Szczecin, Volume 24/2010, pp. 111-117.

Szczęśniak, B. (2013). Linking EPC models – an alternative approach. Scientific Journals Maritime University of Szczecin, 34(106), pp. 79-84.

Thomas, O., Becker, J., Jannaber, S., Riehle, D. M. and Leising, I. (2018). Collaborative Specification Engineering: Kollaborative Entwicklung einer Sprachspezifikation der Ereignisgesteuerten Prozesskette unter Verwendung einer Wiki basierten Onlineplattform. In J. Becker, B. Hellingrath, S. Klein, H. Kuchen, H. Trautmann, G. Vossen, ed., Arbeitsberichte des Instituts für Wirtschaftsinformatik, Volume 140.

Van Der Aalst, W.M.P. (1999). Formalization and verification of event-driven process chains. Information and Software Technology, Volume 41, pp. 639-650.

Abstract. EPC models are currently among the leading standard business process modelling solutions. As long as the rules for creating flat EPC models are explicit and raise no significant doubts, when it comes to linking models of different types, the literature of the subject delivers diversified solutions. Four such alternative concepts that can be applied in this respect have been identified and described in this article. They have been used to link EPC models for four cases proposed. These cases differ as to the degree of complexity, and they represent different kinds of potential links between processes being modelled. Using the concepts in question with reference to the cases defined has made it possible to identify situations in which it is impossible to link models, or when it raises certain doubts. What has also been proposed is that one of the concepts identified in the paper can be modified in such a manner that using it to link EPC models does not pose any problem in any of the cases discussed.

Keywords: business process modelling, EPC, BPM, linking of process models