From Process to Simulation - A Transformation Model Approach

Oliver Kloos, Volker Nissen, Mathias Petsch

Chair of Information Systems in Services
Ilmenau University of Technology
PO Box 10 05 65
98684 Ilmenau, Germany
{oliver.kloos,volker.nissen,mathias.petsch}@tu-ilmenau.de

Abstract: Process models are frequently used to visualise, analyse and control the flow of work in business processes. In principle, the information contained in process models could be used to simulate and optimise business processes with the help of simulation software. However, business process models are created for other purposes than simulation and, therefore, normally do not contain enough information to directly execute simulation studies using them. A process model first has to be converted to a simulation model which holds different information necessary for the simulation. In this paper, a transformation approach is presented which can be used to convert process models based on the event-driven process chain (EPC) notation to different simulation software systems.

1 Introduction

Business process models serve to document and describe the business activities of an organisation, determine the critical points of business processes and/or analyse the requirements for the introduction or implementation of new information systems [Sc00, FL03]. At the same time business process models can be used for other purposes. But, the modelling of business processes is often associated with the rationalisation of these processes (e.g. reducing processing time, lowering process costs etc.) [BRU00, BS04]. Simulation allows us to observe and analyse the behaviour of processes. Therefore, simulation can be used to create and to check the consequences of changes in the processes, for example a different amount of available resources. [BF87, DE00].

In practice, the first step in improving current business processes is the documentation and (usually manual) analysis of existing processes. The use of these process models for simulation and the optimisation using simulation software are seldom considered while modelling the business processes. Normally the methodologies for modelling processes do not include enough (quantitative) information to model and run simulation studies [St06]. Nevertheless, process models can form a good starting point to create simulation models. The main problem is how a process model can be conveniently transformed to create the foundation of a simulation model. For the purpose of simulation, parts of the process model could be aggregated and other parts ignored. On the other hand, certain additional information, such as resources and capacities must be added, which are not available from the original process model but are, nevertheless, important to run a simulation study [Al07]. Our objective is the creation of a transformation model which can be used to normalise the process model and to gather the information necessary for the simulation model. The concept of the transformation model is that of a middleware. Therefore, a process model created without considering the requirements for simulation will be converted to the transformation model, normalised and prepared for simulation and then be transferred to a simulation system. The transformation model reduces the number of transformations from different process modelling notations to various simulation systems. For example if there are four process modelling notations and three simulation systems, there would be twelve transformations. With the transformation model there would be only four transformations to the transformation model and three to the simulation systems. If a direct transformation is used, every process modelling notation needs methods to prepare a process model for a simulation. This preparation is only necessary for the transformation model. Furthermore, the meta-models of the process modelling notation do not include information necessary for simulations. Another problem are resources, for example rooms, which are a simulation constraint, but cannot be added to an extended EPC. So every process modelling notation would need to be expanded if a transformation model is not used. Some process models can of course be directly simulated, for example with the ARIS Toolset, but some process models require additional constraints which cannot be added in process models, so they have to be transformed into simulation models and simulated with other simulation systems. We do not use an existing notation for the transformation model such as the EPC or Business Process Modeling Notation (BPMN). While the EPC is restricted to a function and event sequence, BPMN offers a wide spectrum of methods to model a business process. On the one hand the allowable elements would have to be restricted; on the other hand new artefacts would need to be included. However important information for the simulation would need be added as attributes of the elements which are not visible in the graphical representation of the process model. We introduced a new notation that contains all necessary elements to create a simulation model and that shows all relevant information using a graphical representation to support the human part of the normalisation and preparation of the transformation model.

The remainder of this paper is structured as follows. First, work related to our own research is discussed. Then, we define business process models and simulations and explain the foundations of our transformation model. After that, the notation and different views necessary to generate a valid transformation model are highlighted. Next, the transformation process from a business process model to a simulation model is explained. Finally, our results are summarised and future work is indicated.

2 Related Work

The transformation of business process models to simulation environments has been an active research topic for several years [Gr03, AJ05, HR06, Di07, DD08]. The main problem concerns the different degree of model detailing [NRS03, HR06]. The following problems especially occur during the transformation from process to simulation models:

- Process models show a higher level of detail in the number and description of process steps, which are seldom necessary for simulation models (process steps are often summarised by one module in simulation models).
- Simulation models require detailed and operationalised data (such as distribution of process time, probabilities, frequency of process steps etc.).

There are several papers that deal with these problems. Greasley [Gr03] used a process map of a prisoner custody process as a conceptual model for the simulation model. First, the process map was created and data were collected such as arrival time and function duration. Next, the process map was transferred to a simulation system. Damij and Damij [DD08] used an activity table of a clinical business process, which represents a table with organisational units as columns and activities as rows. The activities are associated with an organisational unit through the cells and connected with links to create a flow. To create a simulation environment the process was transformed into a flow chart. However, the flow chart presented contains the same information as the activity table but uses a different graphical representation. An and Jeng [AJ05] presented a flow chart, of a supply chain management domain. The functional structure of the flow chart was used to create a simulation model by adding input and output data and their corresponding data repositories. However, the authors also created a system dynamic model, which represents the positive and negative influences of the elements on each other. Both models were used to simulate the supply chain. Dickmann et al. [Di07] presented an approach which supports the process improvements projects. They also used conceptual process models, which were not created for the purpose of simulation but to create simulation models. Therefore, they used questionnaires to gather the information necessary for the simulation model.

All approaches presented use existing process models as a foundation to create simulation models. Since everyone had different notations for the process models, each required its own transformation methods and rules for targeting a simulation system. None of these approaches uses methodology or notation created with the purpose of preparing a process model for simulation. Heavey and Ryan [HR06] analysed the support of simulation in process modelling tools and methods. They outlined that none of them supports the requirements-gathering phase of simulation. As a conclusion they created a process modelling method called Simulation Activity Diagrams, which support the conceptual modelling phase of a simulation project. However this approach does not use existing process models to prepare them for a simulation.

Other papers relevant in the context of the transformation and preparation of a process model for simulation deal with the reduction of process models. While these papers do not focus on simulation they nevertheless can be used for it. Polyvyanyy et al. [RSW08] describe a complexity reduction approach of large EPCs through joining of loops, sequences, and blocks. The approach is based on pattern identification of AND, OR and XOR connectors and the aggregation of functions and events to reduced functions. A similar idea is put forward by Sadiq and Orlowska [SO99]. The objective of their approach is to discover incorrect graphs within process models. The algorithm identifies structurally correct workflow graphs with the help of so called reduction rules and reduces them to an empty graph. On the other hand Allweyer [Al07] presented a procedure to create a more detailed process model from an imprecise process model. To execute the transformation rules a template for the detailed process model is necessary. The transformation rules use data associated with the imprecise process model to create a detailed process model based on the template. These papers are relevant for the normalisation of the transformation model.

3 Foundations of the transformation model

We define a business process, according to Rosemann, as an enclosed structured, logical series of functions for the handling of a relevant business object. Therefore, a business process model is a model of a structured, logical series of functions executed on an object [Ro96]. Our second object of interest is simulation. Simulation is defined as a method to imitate the operation of a real-world system. Simulation involves the construction of a simulation model and data acquisition, the execution of a simulation study and the analysis of the simulation results [VDI95, BCN05].

The transformation model was derived under consideration of the process modelling using EPC and BPMN notations and three simulation software systems. The core of the transformation model, the activity, is based on three premises. The first premise states that every work in the real world can be described using a combination of an action, an object and one or more resources. An example for such an activity is, 'The salesperson creates the sales order in an ERP system'. The action represents a work step, in this example 'create'. Other examples are 'check', 'order' and 'authorise'. The object is the item on which the action is executed, in the example 'sales order'. Resources are a tool to support the execution of the action, 'salesperson' and 'ERP system' in the example. The combination of these three elements represents the activity in the transformation model. A more complex activity for instance is 'The nurse checks the blood pressure of the patient'. The nurse is the resource and the patient the object. The action however is more than a single verb, in this case 'check blood pressure'. So an action can be a single verb or a combination of a verb and a noun. In contrast to van der Aalst [Aa98] we define an activity differently. We also use a resource to specify an activity, but we divide the task into an action and an object. The case van der Aalst uses would be an instance of an object in a simulation. If the instance is added it would be an activity that is executed.

The second premise states that only a combination of an action, an object and at least one resource can have a processing time. A combination of only two of the three elements cannot have a processing time. A resource is needed to execute an action and the execution must be on an object. Only then the activity being processed can contain a processing time. The third premise states that when there are more actions to be executed than there are resources, a queue will be created. To specify a queue, two pieces of information are needed: the capacity of the queue and the queue processing technique. The activity assumes that if there is no information about the queue, the capacity is infinite and the queue processing technique is 'First In, First Out'. If an activity transforms several objects into a new object, the activity can be extended by input and output information. This

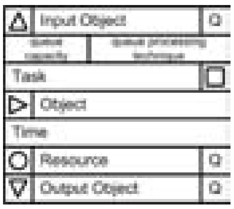


Figure 1: Complex activity

type of complex activity is shown in Figure 1; the 'Q' indicates the quantity needed. The current notation is shown in table 1 and 2.

As shown in the next chapter there is a need to bind resources for more than one activity. For example, assuming a flow chart shows two activities that can only be executed together. Activity one uses resources A and B and activity two uses resource A. Resource B can be released after the execution of activity one but resource A can only be released after activity two has been executed. Therefore, resource A is bound before activity one and released after activity two in the flow chart. An example process that needs a resource bind and release is shown in figure 3.

The last main group of elements in our transformation model are pathways. In the context of a simulation, there are three different types of pathways: parallel, inclusive and exclusive. Every pathway also has gateways containing the condition for the execution of a flow path, except for the parallel pathway. A parallel pathway indicates that two activities can be executed at the same time. The inclusive and exclusive pathways have two variants: probability- and condition-based. One or more flow paths can be executed in the inclusive pathway and only one in the exclusive pathway, according to the occurrence of the gateway. The condition-based gateway checks an attribute of the object. Since the gateway contains the condition, one or more flow paths can be executed. The attribute-based gateway contains the attribute of an object to be checked and the gateway contains a value. If the attribute of an object instance matches the value in the gateway, the flow path is executed. Therefore, only one flow path can be executed. The splitting pathway is displayed with a single border, while the joining pathway has a double border.

To use a condition-based or attribute-based pathway, an attribute of an object needs a value. The value is set by the attribute set. The condition-based pathway can be used to add complex conditions for the execution of flow paths. The attribute-based pathway can be used instead of an exclusive pathway. For example a process model contains two exclusive pathways that use the same condition to determine the flow path. When the first flow path is selected in the first exclusive pathway, then the first flow path of the second pathway must be selected. A process model in the notation of the EPC would use two exclusive operators with the same events, so that the corresponding flow paths are selected. However, the exclusive pathway is probability-based. To select the first flow path in both pathways the second pathway has to be an attribute-based pathway and every flow path of the first pathway needs an attribute set.

Rittgen showed that an OR join in EPC notation has three interpretations [Ri99]. Waiting

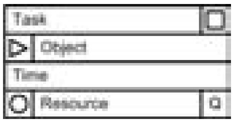
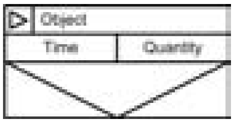
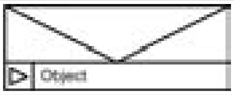
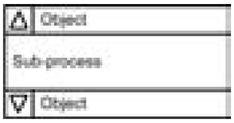
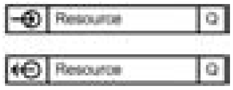
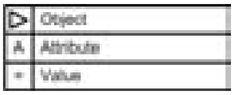
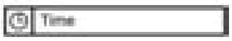
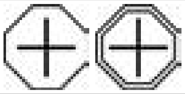
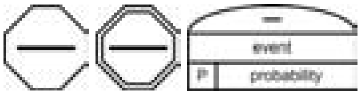
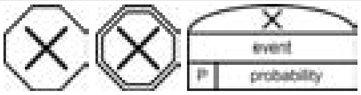
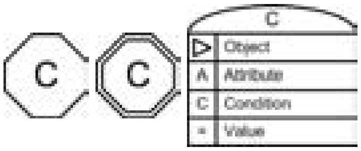
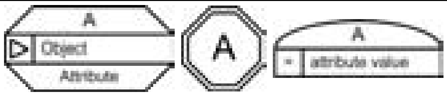
Notation	Explanation
	The *Activity* shows the mandatory fields that are necessary for the flow chart of the transformation model.
	The *Source* creates new instances of the object in the listed time interval and quantity.
	If an instance of an object arrives at the *Sink* it is removed from the model.
	A *Sub-process* passes an object to another flow chart. After the execution of the other flow chart, an object is passed back to the flow path in that it is integrated.
	The *Resource binding* and *Resource release* is used to bind a resource to the flow path it is integrated. Bound resources can only be used in that flow path.
	The *Attribute Set* sets a value of an attribute. If an attribute is set the attribute-based and condition-based gateway can check this attribute.
	The *Delay* delays the execution of a flow path without the need of resources.
	All flow paths between the *Parallel Pathway* are triggered. Activities in the different flow paths can be executed at the same time.
	One or more flow paths can be triggered in the *Inclusive Pathway*, depending on the probability of the gateway.
	Only one flow path can be triggered in the *Exclusive Pathway*. The sum the probability of all gateways of one exclusive pathway must be 100%.
 C C C Object A Attribute C Condition = Value	A flow path is triggered in the *Condition-based Pathway*, if the condition of the gateway is triggered, so one or more flow paths can be executed.
	The *Attribute-based Pathway* contains an attribute of an object. If the value in the gateway matches the attribute of the instance only that pathway is executed.

Table 1: Notation of the transformation model flow chart

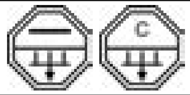
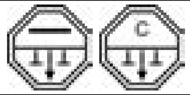
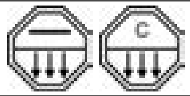
Notation	Explanation
	The successive element of the joining pathway is executed, when all triggered flow paths are completed. ('wait for all')
	The first flow path that is completely executed will be passed to the successive element of the joining pathway. All other flow paths end at the joining pathway. ('first come')
	Every flowpath that is completely executed will be passed to the successive element of the joining pathway. ('everytime')

Table 2: Joining pathways of the inclusive and condition-based pathway

for the completion of all activated flow paths is the default semantic ('wait-for-all'). Another interpretation is to only wait for the first flow path to be completed. All other flow paths will be ignored ('first-come'). The third interpretation is called 'every-time'. Each completed flow path will be passed to the next element. To cover these interpretations, the joining pathways of the inclusive and condition-based pathway must be specified during the normalisation of the transformation model. Table 2 shows the notation of the joining pathways, which are only available in a consistent transformation model. While the wait-for-all interpretation would be the default, a simulation with the ARIS Toolset shows a different interpretation. If there is an OR split and a corresponding OR join, the OR join passes on more instances than the OR split received, if the sum of the probability of all flow paths is greater than hundred percent. So an OR split in the simulation of the ARIS Toolset is interpreted as every-time.

In addition to the flow chart we define three supplemental views for the transformation model. These views are derived from the first premise of the activity. Therefore, there are an object view, a resource view and an optional action view. The object view only is necessary if a condition-based or attribute-based gateway is used. However, the resource view is inevitable because it contains the quantity of the available resources. All three views are organised in a tree with parent-child relationships. While the action view only organises actions as a function tree; the object and resource view provide additional information. The object and resource view are explained below.

An activity can only be executed with an instance of the object it is composed of. The instance of an object is created by the source. But there could be a process model with an activity that uses an object not created by a source and only used by a single activity. An example for this is a process model that characterises the processing of a sales order. To check if the customer is available in IT system there could be an activity 'check customer number' executed by a salesperson. A source would create an instance of a sales order, but the object of the activity is 'customer number'. To solve this problem, objects are divided into 'process objects' and 'non-process objects' in the object view and flow chart. Process objects are created by a source or the output of an activity and removed by a sink or as used as an input of an activity. However, a non-process object is not created or used as an input and only used in a single activity. So only process objects can be used for condition- or attribute-based gateways. Therefore, the process objects have to contain the attributes in the object view, so that they can be used in the flow chart. Because non-process objects cannot be used by gateways there is no need to add attributes to these objects. To symbolise that a non-process object is used in the flow chart, the triangle identifying the process object is mirrored.

Resources can be divided into four different types: human, stationary, movable and immaterial resources. This division results in three criteria human vs. non-human, material vs. immaterial and stationary vs. movable. While a human resource is always a human, material and movable, whether it moves or not, an immaterial resource is immaterial and non-human. Because of the immateriality there is no place in the real world it is situated, ignoring the fact that it could be saved in a data storage. The stationary and moveable resources are non-human and material, whereas the first is stationary, the second is moveable. An example of a human resource is a doctor or salesperson. A stationary resource could be a room or a machine, while a movable resource is for example a forklift truck. An example for the immaterial resource is an IT system.

All four types are children of the type resource, for example a room would be the child of the stationary resource. To execute a simulation study, it is necessary to define the quantity of each resource used. An unlimited resource does not need to be included in a simulation study. While the flow chart only holds the number of necessary resources for each activity, the resource view contains their quantities. Currently in development is the attribute definition for the four types of resources, which can be transformed for a simulation software system. Humans could have a shift schedule, while stationary resources could have an availability or capacity. Stationary resources could also be used to create a layout. Using this layout the human and movable resources could be assigned paths they must take to get to another stationary resource.

4 Transformation Process

The first step before executing a simulation study is the definition of the problem, representing the purpose of the simulation. Based on the defined problem an individual or multiple process models are selected for analysis in the simulation study. It should be noted here that the use of the transformation model requires existing process models as input. The next step is an automatic transformation of the process models to the notation of the transformation model. Corresponding transformation rules are mentioned below. Whereas the process models are created for other purposes, the transformation model has to be fed with additional information necessary for the execution of a simulation study. Therefore, the transformation model created via the transformation rules is only a conceptual model. The conceptual model is characterised that it uses the notation of the transformation model, but the elements used lack information, such as execution time in the activity or the probability in the exclusive gateway, which were not found in the process models. Therefore, data has to be collected and used to normalise the transformation model into a consistent model. On the one hand all elements used in the consistent transformation model have the necessary information, but on the other hand normalisation rules are executed such that elements

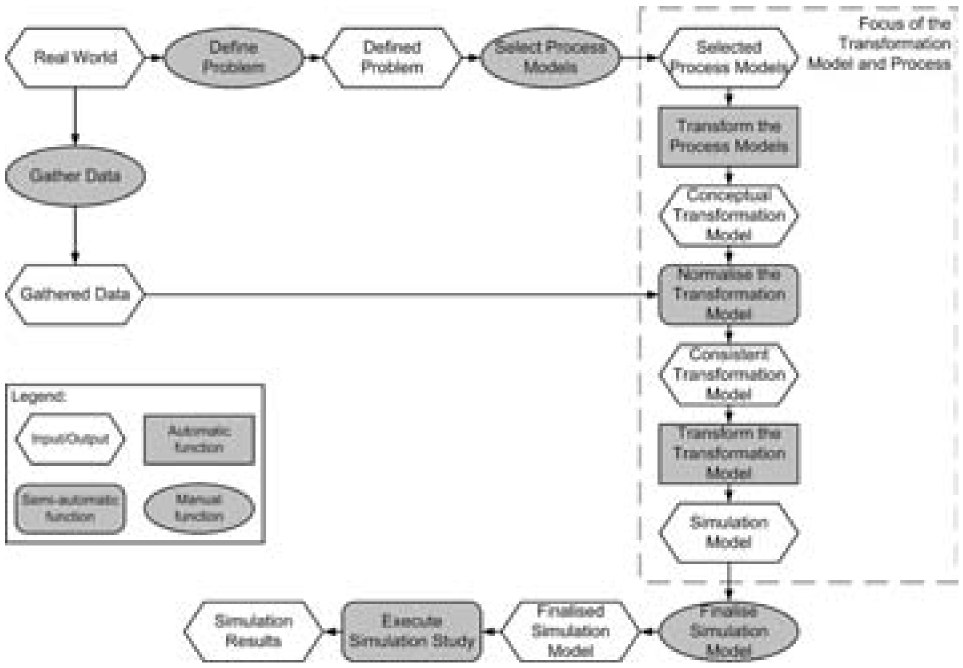

Real World
Define Problem
Defined Problem
Select Process Models
Selected Process Models
Focus of the Transformation Model and Process
Transform the Process Models
Conceptual Transformation Model
Gather Data
Gathered Data
Normalise the Transformation Model
Consistent Transformation Model
Transform the Transformation Model
Simulation Model
Legend:
Input/Output
Automatic function
Semi-automatic function
Manual function
Simulation Results
Execute Simulation Study
Finalised Simulation Model
Finalise Simulation Model

Figure 2: Procedure that leads from process models to simulation

could be changed or added to the consistent transformation model. Examples for the normalisation are shown below. This normalisation step can be executed semi-automatically. While the merge of activities can be executed semi-automatically, additional resources for the activities have to be added manually. When a consistent transformation model has been created, the model can be automatically transformed into a simulation model. This simulation model depends on the software system being used to execute a simulation study. Before the simulation study can be executed, the simulation model can be expanded with additional information, depending on the simulation software system. Examples for this information are the duration of the simulation, resource utilisation or diagrams to visualise the simulation results. With the finalised simulation model the simulation study can be executed to gather knowledge, which can in turn be used to interpret the real world.

The procedure of the described transformation process is shown in figure 2. The figure also shows the main focus of the transformation model, the transformation from process models, the normalisation of the transformation model and the transformation to a simulation model. While both transformations can be executed automatically, normalisation of the transformation model needs human interactions and is therefore only semi-automatic. With this procedure there is no need to change the existing process models, but only to adapt and enrich their contents in the process of converting them to a simulation model.

Next, examples are given for using the transformation model and process. The examples cover transformation rules from the EPC and rules to normalise the transformation model.

While transformation rules for BPMN are currently created, we identified 23 transformation rules to convert the elements of an EPC to the elements of a flow chart of the transformation model. The transformation rules for a specific simulation software system are not discussed in this paper.

Normally, resources used in the EPC cover humans and information systems. Stationary or movable resources are therefore initially unavailable. Focussing on information systems first, the information system in the EPC can be transferred to the resource view of the transformation model as a child of the immaterial resource. The quantity of the information system is one. To avoid the fact that only one operation can be executed at the same time with the information system, the attribute capacity has to be assigned. However, if the focus of the simulation study is not on the utilisation of information systems, it can be more appropriate not to transfer the resource information system from the EPC to the transformation model at all, or remove the resources during normalisation. Therefore, some analysis is required during this step to decide on the better transformation option.

As for humans in the process, it is suggested to use organisational units from an organigram, which executes the functions of the EPC. The two alternatives discussed here are positions and person types. While positions cover a person of a specific department such as a salesperson, purchaser or accounting clerk, person types deal with hierarchy, such as supervisor, staff member or proposer. A specific person can, therefore, have a particular person type and, simultaneously, a certain position in the company. To prevent the situation in which there are more humans represented in the simulation model than in the actual enterprise, only one of these alternatives should be implemented in the transformation model. The transformation rule for both possibilities is the same. A position or person type is added as a child of the human resource in the resource view. The quantity of humans in the resource view can be determined by summing up the amount of people assigned to the position or person types associated with the resource view.

All start events are converted to sources and all end events are converted to sinks. The subject used in the event is the object in the source and sink. Events after a separating OR or XOR operator are used in the gateways of the inclusive and exclusive pathways. All other events are removed and not implemented in the transformation model. The separating operators are converted to the corresponding separating pathways and the joining operators to joining pathways. An AND operator will be a parallel pathway, an OR operator an inclusive pathway and an XOR operator an exclusive pathway. An automatic conversion of operators for condition- or attribute-based pathways is not possible. The activity is created based on the function. Positions, person types or information systems will be the resources of the activity. A resulting transformation rule would be: 'If an XOR-Operator has one successor and more than one predecessor, then it will be an exclusive split pathway.'

An example for the transformation of an extended EPC to the transformation model is shown in figure 3. While the first flow chart shows an EPC, the second flow chart shows a conceptual transformation model, which is automatically created. To execute the three activities a room is needed, so the rooms are added to the third flow chart, the consistent transformation model. The room and the dental assistant are reserved for the flow path. All three activities have to be executed, so a second instance can only use the reserved room after it is released. A semi-automatic transformation rule would be: 'If a resource is used in multiple activities in a sequence, determine if the resource should be reserved for the flow path.' If the extended EPC is simulated, there is a logical problem with the room, for example only one room is available and there are two instances. Instance one executes the first and second functions. Then the second instance executes function one and after that the first instance would execute the revision. In reality the first patient would have to leave the room after the examination, so the second patient can have his preliminary enquiry. Then the second patient would leave the room, so that the first patient can have his revision. To prevent this sequence the room is reserved in the consistent transformation model.

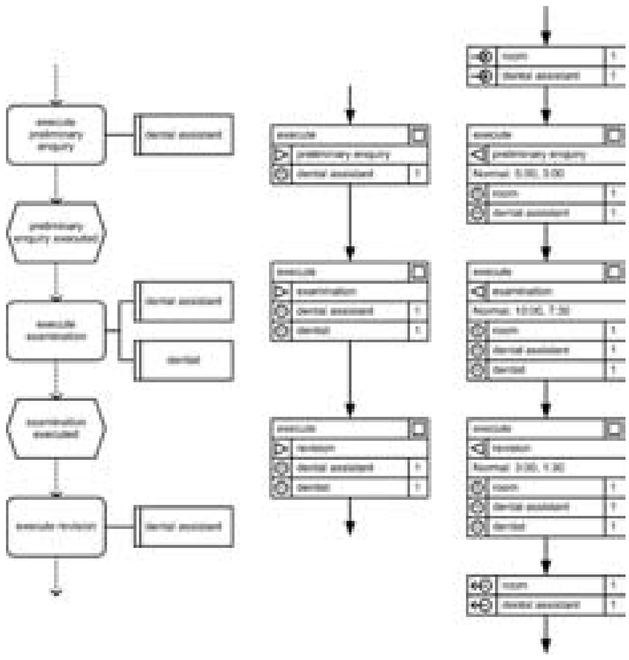


Figure 3: Example process with an extended EPC and the consistent transformation model

Another approach for normalising the transformation model is to add attributes to the objects and change the pathways. An inclusive pathway can be converted to a condition-based pathway and an exclusive pathway can be converted to an attribute-based pathway. This approach can be used when different input streams are the focus of the simulation study. The values of the object attributes depend on the input stream and therefore the probability of the pathway. This approach can help reduce the maintenance effort of the simulation models. There could also be different terms used in the process model, for example bill or invoice. These terms have to be harmonised during the normalisation of the transformation model.

An example for a more complex normalisation is shown in figure 4, based on Rosemann [Ro96]. The EPC is modelled according to the Guidelines of Business Process Modelling [BRU00]. However, the event 'goods arrived' has to be a source. But goods only arrive when an order is created. If this process is simulated goods may arrive without the creation of an order. The normalisation of the conceptual transformation model converts the joining parallel pathway and the start event into a delay. So every order has a delivery time. On the one hand this example shows on a manual normalisation in which resources are such as the loading ramp or the forklift truck are added. On the other hand an automatic normalisation process is shown in which the parallel join pathway and the source are replaced by a delay. The corresponding normalisation rule would be: 'If a parallel join pathway has one activity as a predecessor and one source as predecessor, then it is replaced by a delay.'

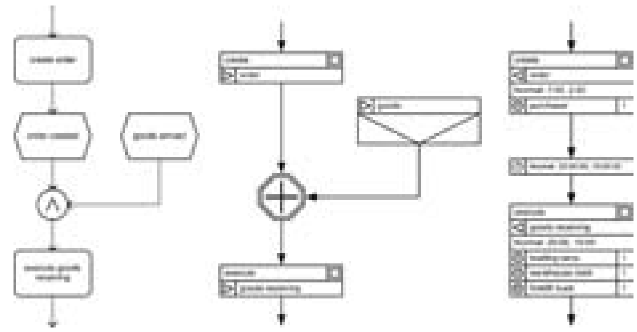


Figure 4: Example for the normalisation of the transformation model

After the normalisation of the transformation model has been executed, a direct conversion to a simulation model becomes possible.

5 Conclusion and Future Work

This paper introduced a process and notation for semi-automatically transforming process models into simulation models where the process models were not originally created with simulation in mind. Process models are converted to transformation models and normalised, in order to ultimately transform them into simulation models using arbitrary simulation software. The transformation model contains a flow chart and three views, which were presented here. Also, some initial transformation rules were demonstrated with reference to process models in the notation of the extended EPC. Additionally, some information was provided on how to reduce the complexity of the original process model and further normalise the transformation model.

The transformation model introduced is still a work in progress and the next step in our research is the creation of a meta-model for the flow chart and the three views in the transformation model. Based on meta-models the transformation rules could be created for the transformation of process models into the transformation model. However, specific linguistic transformation rules are necessary if a meta-model element contains more than one element of the transformation model. These rules use word classes and cases. While these rules are only valid for a single language or perhaps also for a single notation, a research goal is the creation of linguistic transformation rules for different languages and notations. As for tool support of the notation, we are currently developing a prototypical implementation based on the Eclipse framework. At the moment the prototype supports modelling of the flow chart, an automatic conversion of the EPC markup language [MN05] to the flow chart and automatic export to the simulation software Arena. Preliminary transformation rules for BPMN are identified but not described in concrete transformation rules. In future research, these transformation rules for BPMN and the UML activity diagram will be developed based on the meta-model, so that the most frequently employed modelling notations can be conveniently converted to a simulation model.

Other research themes to be addressed include developing further rules to normalise the transformation model. A starting point here is the creation of a procedure model, which covers all steps of the normalisation. Those steps must include reducing the complexity of the transformation model so that only the relevant aspects for the simulation remain as well as adding relevant additional information through attributes of the resources.

Currently, the transformation rules to create a simulation model take into account process-oriented simulation software. However, in the domain of logistics and production, resource-oriented simulation software is dominant. Therefore, some initial research is currently done to create transformation rules for this and other types of simulation software.

References

[Aa98] Aalast, W.M.P. van der: The Application of Petri Nets to Workflow Management. In: Journal of Circuits, Systems and Computers 8(1), 1998; pp. 21-66.

[Al07] Allweyer, T.: Erzeugung detaillierter und ausführbarer Geschäftsprozessmodelle durch Modell-zu-Modell-Transformationen. In (Nüttgens, M.; Rump, F. J.; Gadatsch, A. Eds.): 6. GI-Workshop EPK 2007 : Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten, St. Augustin, 2007; pp. 23–38.

[AJ05] An, L.; Jeng, J.-J.: On developing system dynamics model for business process simulation. In (Kuhl, M.E.; Steiger, N.M.; Armstrong, F.B.; Joines, J.A. Eds.): Proceedings of the 2005 Winter Simulation Conference, Orlando. IEEE Press, Piscataway, 2005; pp. 2068–2077.

[BCN05] Banks, J.; Carson, J.; Nelson, B.L.: Discrete-event system simulation. Prentice Hall, Upper Saddle River, 2005.

[BF87] Bratley P.; Fox, B.L.: A guide to simulation. Springer, New York, 1987.

[BRU00] Becker, J.; Rosemann, M.; Uthmann, C.: Guidelines of Business Process Modeling. In (Aalst, W.; Desel, J.; Oberweis, A. Eds.): Business Process Management : Models, Techniques, and Empirical Studies. Springer, Berlin, 2000; pp. 241–262.

[BS04] Becker, J.; Schütte, R.: Handelsinformationssysteme : Domänenorientierte Einführung in die Wirtschaftsinformatik. Redline Wirtschaft, Frankfurt, 2004.

[DD08] Damij, N.; Damij, T.: Improvement of a clinical business process. In (Al-Dabass, D.; Nagar, A.; Tawfik, H.; Abraham, A.; Zobel, R. Eds.): Second UKSIM European Symposium on Computer Modeling and Simulation, 2008. EMS ’08, Liverpool. IEEE Press, Piscataway, 2008; pp. 305–310.

[DE00] Desel, J.; Erwin, T.: Modeling, Simulation and Analysis of Business Processes. In (Aalst, W.; Desel, J.; Oberweis, A. Eds.): Business Process Management : Models, Techniques, and Empirical Studies. Springer, Berlin, 2000; pp. 247–288.

[Di07] Dickmann, C.; Klein, H.; Birkhölzer, T.; Fietz, W.; Vaupel, J.; Meyer, L.: Deriving a Valid Process Simulation from Real World Experiences. In (Wang, Q.; Pfahl, D.; Raffo, D.M. Eds.): International Conference on Software Process, ICSP 2007, Minneapolis. Springer, Berlin, 2007, pp. 272–282.

[FL03] Frank, U.; van Laak, B.L.: Anforderungen an Sprachen zur Modellierung von Geschäftsprozessen. Arbeitsberichte des Instituts für Wirtschaftsinformatik Nr. 34, Universität Koblenz Landau, 2003.

[Gr03] Greasley, A.: Using business-process simulation within a business-process reengineering approach. In: Business Process Management Journal 9(4), 2003; pp. 408–420.

[HR06] Heavey, C.; Ryan, J.: Process modelling support for the conceptual modelling phase of a simulation project. In (Perrone, L.F.; Wieland, F.P.; Liu, J.; Lawson, B.G.; Nicol, D.M.; Fujimoto, R.M. Eds.): Proceedings of the 2006 Winter Simulation Conference, Monterey. IEEE Press, Piscataway, 2006; pp. 801–808.

[MN05] Mendling, J. Nüttgens, M.: EPC Markup Language (EPML) : An XML-Based Interchange Format for Event-Driven Process Chains. In: Information Systems and E-Business Management 4(3), 2005; pp. 245–263.

[NRS03] Neumann, S.; Rosemann, M.; Schwegmann, A.: Simulation von Geschäftsprozessen. In (Becker, J.; Kugeler, M.; Rosemann, M. Eds.): Prozessmanagement - Ein Leitfaden zur prozessorientierten Organisationsgestaltung. Springer, Berlin 2003; pp. 449–468.

[Ri99] Rittgen, P.: Modified EPCs and their semantics. Arbeitsberichte des Instituts für Wirtschaftsinformatik Nr. 19, Universität Koblenz-Landau, 1999.

[Ro96] Rosemann, M.: Komplexitätsmanagement in Prozeßmodellen : Methodische Gestaltungsempfehlungen für die Informationsmodellierung. Gabler, Wiesbaden, 1996.

[RSW08] Polyvyanyy, A.; Smirnov, S.; Weske, M.: Reducing the complexity of large EPCs. In (Loos, P.; Nüttgens M.; Turowski, K.; Werth, D. Eds.): Modellierung betrieblicher Informationssysteme : Modellierung zwischen SOA und Compliance Management. Köllen, Bonn 2008; pp. 195–207.

[Sc00] Scheer, A.-W.: ARIS - Business Process Modelling. Springer, Berlin, 2000.

[SO99] Sadiq, W.; Orlowska, M.E.: Applying Graph Reduction Techniques for Identifying Structural Conflicts in Process Models. In (Jarke, M.; Oberweis, A. Eds.): Advanced Information Systems Engineering. Springer, Berlin 1999; pp. 195–209.

[St06] Staud, J.: Geschäftsprozessanalyse : Ereignisgesteuerte Prozessketten und objektorientierte Geschäftsprozessmodellierung für Betriebliche Standardsoftware. Springer, Berlin, 2006.

[VDI95] VDI: Simulation von Logistik-, Materialfluß- und Produktionssysteme - Begriffsdefinitionen. Verein Deutscher Ingenieure, Berlin, 1995.