Oliver Kloos, Volker Nissen, Mathias Petsch, Hagen Schorcht

Service Modelling as a Basis for Simulation

Process models are often created without regard to their possible later use in simulations. If a process model generated in this way is to be simulated, additional information must be gathered, the model adapted and transferred into an appropriate simulation environment. This process is complex, weakly structured and errorprone. Using practical example data from an outpatient clinic, this work introduces an approach for model transformation with which service process models can be transferred into a simulation environment in a structured way and supported by the system. The transformation concept focuses on process models generated without taking into consideration the requirements of future simulation. The concept prepares the process models for simulation and supports the transfer of the models into the appropriate simulation environment.

1 Introduction

Process models serve to assist in documenting and visualising process flows, which can for example help to create process transparency, preparing for quality assurance certification or guaranteeing efficient integration of new employees. However, it can be helpful, especially in the area of process analysis and optimisation, not only to visualise processes but also to change them into an executable form with the help of a simulation environment. In this way, changes in the process can first be run through and their effects analysed before they are implemented in the actual system. Furthermore, it is possible to check the behaviour of a process resulting from changes in the input variables.

Process models have achieved significant proliferation in organisations irrespective of the modelling language used. However, a simulation requires additional or other pieces of information than a pure visualisation of the processes. Also, depending on the process modelling notation, the process models do not generally possess the necessary quantitative data for carrying out simulations. However, the models can be used as the basis for generating a simulation model. In the present work a conceptual approach will be discussed for transforming a process model into a simulation model, for example by systematically gathering additional data and adapting the model to the corresponding environment. The individual steps are explained using the example of an outpatient clinic.

The rules basis represents the conceptual core of the contribution. This consists of the conversion of a process model into a transformation model, the normalisation of the latter as the starting point for subsequent simulation and the final transfer into the simulation environment.

The work is structured as follows. First, the discussion covers related contributions dealing with the simulation of process models. Next, an explanation is given of the application example, which is used to illustrate the individual transformation steps. Section 4 deals with the substance of the transformation model being considered here as well as the steps for preparing a process model for simulation. This especially includes the transformation and normalisation of the process model. Finally, a short summary is given as well as highlights of future research requirements in this area.

2 Related Work

The transformation of process models into a form suitable for the simulation environment has been a topic of academic research for quite some time (An and Jeng 2005; Damij and Damij 2008; Dickmann et al. 2007; Fetter and Thompson 1965; Greasley 2003; Heavey and Ryan 2006). The primary problem in transferring the process models into simulation models lies in the different levels of necessary detail (Heavey and Ryan 2006; Neumann et al. 2003). Here, two problems are evident, in particular when attempting transformation:

- Process models possess a higher level of detail in the number and description of the process steps, which is unnecessary in this form for simulation models (this means e.g., that multiple process steps can be combined into one module in the simulation model)
- Simulation models require detailed and quantified data on the process (e.g., probability distributions, process running time, frequency of process steps, costs etc.)

Various papers confront these problems. Greasley (2003) for example uses a process map as a conceptual model for simulation. A process map is first designed and then assigned quantitative data such as arrival time and process duration. Afterward, the actual conversion from process map to a simulation model is carried out.

Seyfert and Kavermann (2006) face the different requirements with regard to level of detail and the demand for quantitative data by separating the static logical modelling component (databank) from the dynamic, discrete event simulation (the simulation model itself). Above all, this opens up the possibility for differentiating between the subject-oriented side (domain experts) and the developer side (simulation experts).

Damij and Damij (2008) use a process chain diagram in table form to describe the clinical business process. There, the representation of the organisational units is done in columns and the corresponding processes are kept in rows (similar to swimming lanes). This means that the activities in the table cells can be uniquely assigned to organisational units and can be combined in such a way as to allow the process flow to be depicted. In order to be able to derive a simulation model, the process being modelled in the table is converted into process charts. However, the latter possess the same level of detail as the process chain diagram and merely offer a different description viewpoint.

An and Jeng (2005) also use a process chain diagram to create a simulation model in the supply chain management domain. There, the diagram structure is used to add input and output data as well as corresponding data repositories. Furthermore, a ‘system dynamics model’ is generated to represent the various influences among the elements.

Dickmann et al. (2007) introduce an approach in which a conceptual model is used which does not directly assist in simulation but aids the creation of simulation models. For this, questionnaires are used to gather quantitative data which can be applied in the simulation model.

All of the approaches mentioned here have in common that existing process models are used to derive simulation models. The process models are each represented using a different notation, which requires specific transformation methods and rules to convert the process model to a simulation model. The approaches do neither apply any generic methodologies or notations, nor were any developed to transform process models into simulation models. However, a generic approach independent of the modelling language would reduce the effort necessary for these transformations. First, the generation of model-specific transformation rules can be mini-mised and second, a generic approach can offer appropriate methods for adapting models to simplify the conversion to simulation models.

Heavey and Ryan (2006) analyse tools and methods for process modelling with regard to support for simulations. The results showed that none of the investigated tools or methods uses concepts which support the collection of information (especially quantitative data). From this, a method for process modelling was created called simulated activity diagrams, which in particular support the design of a conceptual transformation model for a simulation environment. However, this approach is not based on existing process modelling methods and tools and therefore it cannot be applied to transformation models in all cases.

Another point of emphasis in the literature in the context of process models is on the reduction or rather adjustment of the models' degree of detail. Even if the transformation into simulation models has not been explicitly discussed, the results described can indeed be applied, especially with respect to the derivation of transformation rules.

Polyvyanyy et al. (2008) describe an approach for reducing complexity of event-controlled process chains by uniting loops, sequences and blocks. The approach is based on the identification of process fragments, e.g., consisting of AND, XOR and OR operators and the combination of them in order to reduce complexity. A similar approach for identifying incorrect graphs was pursued by Sadiq and Orlowska (1999). In this approach correct workflow graphs are reduced using so-called reduction rules until an empty graph has been generated and therefore the proof of correctness has been found. Allweyer (2007) advanced a contrary method involving model-to-model transformation which can enrich coarsely defined models by increasing the level of detail. In order to carry out the transformation, parameterised templates and transformation rules are required, which can lead to executable models using various target notations.

3 Fundamentals and Application Examples

A wide-reaching definition for the term 'process model' is used within the context of a transformation model for converting different types of process model for use in simulation environments. Based on the work of Rosemann (1996), a process is defined here as the logical temporal sequence of working steps. As a property for demarcating processes, an object is assigned to each process which then influences that process (Becker and Kahn 2003), which is subsequently referred to as a process object. According to Law (2004) simulation is the imitation of operations of various kinds of real world systems. The real world is mapped to a model that is used for experiments to get knowledge about the function of the real world.

The focus of the transformation approach is on process models which were created without regard to the requirements for simulation, for example in the preparation for an ISO quality management certification. The utilised transformation model in this context may be interpreted as a middleware between process modelling notations and simulation environments.

In order to demonstrate the applicability of the transformation model, this article looks at a use case involving the outpatient eye clinic of a privately funded hospital in the maximum care category. Modelling and improving the processes in hospitals has not been regarded very long in comparison with other sectors, which is most likely due to the lack of incentives for economic action in the health industry until a few years ago. In the course of introducing the compensation system based on diagnosis related groups (DRG), the cost and competitive pressure on hospitals has increased dramatically. As a result, there is a desire for holistic optimisation of service and administration processes of the hospital under consideration. This forms the basis for the project presented here. The goals of the project are, among other things, the support of forecasting the effects of changing patient loads and the evaluation of process changes, before they are actually implemented.

Compared to other departments, the hospital's eye clinic represents a straight-forward, closed component within the hospital. The range of outpatient services includes physician consultations for medical diagnosis generation and storage, detailed explanations of Service Modelling as a Basis for Simulation patients, initiation and follow-up of therapies, analysis and documentation of the course of treatment as well as recommendations for other care areas requesting them. The result of a previously completed project in the eye clinic was the surveying and modelling of relevant processes there. In total, 36 business processes were modelled using the extended Event-Driven Process Chain (eEPC) notation. Ten processes were identified as main processes, involving the actual treatment of patients. The other 26 processes are periphery, supporting processes, such as checking in patients, scheduling appointments and generating invoices. Because of the straightforward yet still complex elements in the 'pre-examination' process segment (duration of approx. 9 min.), this segment has been chosen as an example to illustrate the application of the transformation model.

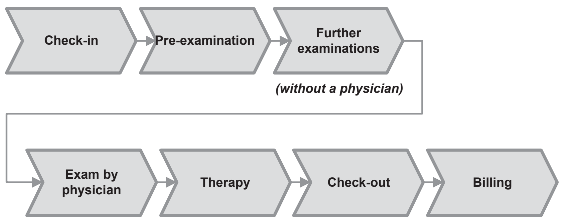

Figure 1: Placement of the pre-examination within the complete process

The 'pre-examination' process serves to gather basic ophthalmological values, such as visual acuity and intraocular pressure. The pre-examination is done on all patients visiting the clinic, representing the initial contact between the patient and the medical personnel. The process follows registration/check-in and occurs before the actual physician examination. Figure 1 shows the placement of the pre-examination within the entire patient visit process. A portion of the pre-examination segment modelled as an eEPC is given in Fig. 7 in Sec. 4.1.

4 The Transformation Concept

The transformation concept, subsequently described as ProSiT (Process to Simulation Transformation), consists of a collection of model types and a procedural model for their use. This contribution builds upon previous work (Kloos et al. 2009).

The focus of the procedural model is on three core areas: the automatic transformation of process models into ProSiT sequence diagrams, the semiautomatic normalisation of the sequence diagrams to prepare for the actual simulation and the automatic transformation of the ProSiT models into a form appropriate for the simulation environment. After the transformation of a process model into ProSiT sequence diagrams has been completed, the result is a conceptual transformation model. This latter is prepared for simulation using normalisation, which involves adding information to the model and reducing its complexity. Successfully completing these steps yields a so-called consistent transformation model.

In addition to process models, other models, such as organigrammes, function trees and class diagrams, can be included as source models, which must be converted into appropriate views within the ProSiT concept. In this way, the transformation of an organigramme can be done, allowing it to be included in the resource tree as part of the resource view of the transformation model. In the present work, only the sequence diagram will be considered.

The type of rules base for the normalisation and the transformation is presented next. The individual results are shown with the help of each process segment from the clinic. Beforehand, the elements of the ProSiT sequence diagram are explained (see Fig. 2), which are used in the presented example.

The core of the sequence diagram is the activity, which is formed from the combination of an action, an object and one or more resources. When these three elements are available, the duration of an activity can be determined. The 'Q' next to a resource indicates the amount of the resource required to carry out the activity, while the symbol next to the duration determines whether the activity is a main (M) or a supporting (S) activity.

Main activities represent the elements directly required to reach the goal of the process, whereas supporting activities make the execution of the main activities possible. The particular main activities in the example pre-examination process include the measurement of intraocular pressure; an example supporting activity is calling out the next patient. If an activity is not carried out on the process object, the triangle symbol indicating an object is flipped. The process object for the example process is the patient.

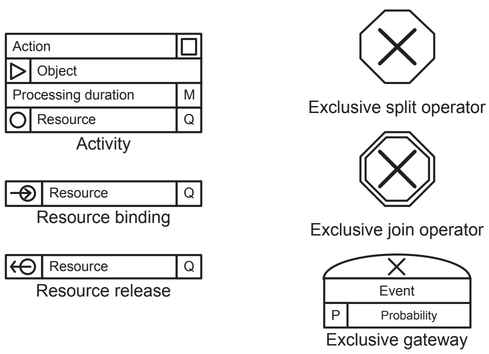

Figure 2: Notation elements from a ProSiT sequence diagram

Resource binding and release is used in a flow path to bind a resource to that path. No other activity outside of this instance of the flow path can access a resource when it is bound. Within the context of the process segment, the pre-examination must be completed, before another patient can enter the pre-examining room.

4.1 Transformation of Source Models

The concept for converting process models for use in a simulation environment consists of a multi-step transformation procedure, similar to that used by Hoyer et al. (2008). Their concept involves the conversion of eEPCs into BPMN models in two transformation steps. The first step creates a rules-based abstraction of the eEPC. These rules are applied directly to the eEPC. The second step is to apply mapping rules in order to generate a BPMN process model on the basis of the eEPC.

The remaining three elements (Fig. 2, right) represent a probability-based forking, in which only one flow path is actually carried out. An exclusive gateway containing the probability for that path's execution must be placed in each flow path after an exclusive split operator.

The first step of the ProSiT transformation concept is to adapt the process model and make it suitable for conversion using preparation rules (ePR – EPC preparation rule). For process models such as eEPCs, these rules involve the removal of events, especially those which do not demarcate the start or end of a process or which are placed after a decision operator. After the process model is prepared for transformation, the actual transformation rules are applied (eTR – EPC transformation rule) in order to create the ProSiT sequence diagram. Also, syntax rules (eSR – EPC syntax rule) are defined in order to assure the correct application of the preparation and transformation rules. Syntax rules are used to check the process model with respect to syntactic correctness and are carried out, before the other two rule types are applied. This transformation concept is illustrated in Fig. 3. The rules base for the preparation and transformation rules necessary for converting an EPC into a ProSiT sequence diagram contains 23 rules.

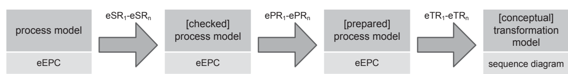

Figure 3: Transformation concept using the example of an eEPC

Syntax and preparation rules are applied within the source model, whereas transformation rules generate a new model. The syntax rules are described using the Object Constraint Language (OCL) and preparation and transformation rules are given in the form of pseudocode. An example of each of the three rule types is given below.

Syntax rule eSR_3 in Fig. 4 assures that an event that follows a forking XOR will be converted into an exclusive gateway during the transformation.

eSR_3
If an XOR operator has a predecessor and more than one successor, all successors must be of the event type.

```
context Xor inv:
if predecessors->size() = 1 and
   successors->size() > 1 then
 successors->forAll(s|s.type = Event)
endif
```

Figure 4: Syntax rule eSR_3

Syntax rules do not represent additional requirements for an EPC and are geared more toward the general syntactic demands for modelling an eEPC (Mendling and Nüttgens 2003).

ePR_4
If an event has an XOR operator as a predecessor and the XOR operator has more than one predecessor, the event is removed.

```
if event->predecessor = Xor and
   event->predecessor->predecessors->size() > 1 then
 set event->predecessor->successor = event->successor
 set event->successor->predecessor = event->predecessor
 remove event from epc
endif
```

Figure 5: Preparation rule ePR_4

An example for a preparation rule is the removal of events which follow a joining XOR operator or a function and themselves have a successor. The former case is covered by preparation rule ePR_4 shown in Fig. 5.

Using syntax rule eSR_3 and preparation rule ePR_4, the remaining events in the model which follow an XOR operator are always the result of a decision. Transformation rule eTR_9 (see Fig. 6) can now be applied.

eTR_9
If an event is preceded by an XOR operator and only has one successor, the event becomes an exclusive gateway.

```
if event->predecessor = Xor and
   event->successor <> null then
 gateway = new ExclusiveGateway
 gateway->id = event->id
 gateway->event = event->name
 add gateway to flowchart
endif
```

Figure 6: Transformation rule eTR_9

Figure 7 represents a portion of the pre-examination process, to which the preparation rules have already been applied. Figure 8 shows the process in the notation of the ProSiT sequence diagram. The information contents of the conceptual sequence diagram is similar to that of the source model. Only information relevant for simulation is converted, whereas information such as inputs and outputs or the information systems used are not transformed. If the information systems are to be used in the simulation, another transformation rule must be defined to contain the necessary transformation step.

Process models are considered semiformal models. The logic of the process is represented on the one hand by the syntax, on the other hand by the semantics of the element labels. Information about the process can be found directly in the element labels of an eEPC. Rules for reading the elements of an activity from a function label are described below as linguistic transformation rules. They are based on grammatical parts of speech and noun case.

A linguistic transformation rule is applied to the function 'measure visual acuity with an eye chart', found on the top right of Fig. 7. When this function is transformed into an activity, the action 'measure visual acuity' is formed from predicate 'measure' and the direct object 'visual acuity'. If not defined otherwise, the object of an activity is the process object. The preposition 'with' makes clear that the prepositional object 'eye chart' should be interpreted as a resource. It is then assumed by the activity. As part of the normalisation, it must be checked whether this is a resource relevant for simulation or whether it has no influence on the course of the simulation and can therefore be removed.

Another form of linguistic transformation rule can be found in the functions 'noting' and 'printing' the results of an examination, which are the functions with output in Fig. 7. The verbs 'to note' and 'to print' point on the one hand to a supporting activity, meaning they can be denoted as such. On the other hand, it is assumed

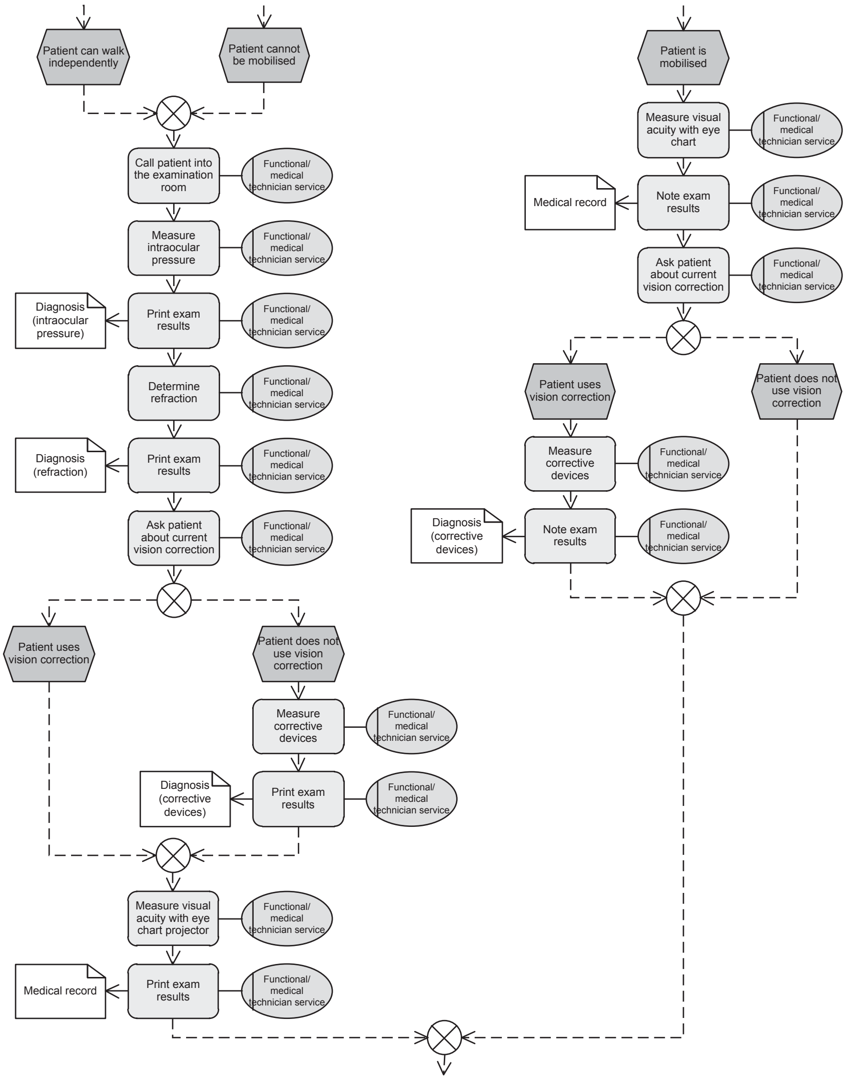

Figure 7: Prepared pre-examination process model

for the verbs that they are not acting on the process object. For these reasons the grammatical object 'examination results' from the function 'note examination results' is used as the object of the resulting activity. The result of the linguistic transformation rule is the transformation of the function into a supporting activity, which is not carried out on the process object, but rather on the direct object of the function label.

The function 'call the patient into the examination room' is also a supporting activity. However, the predicate 'call' or 'ask' does not necessarily imply a supporting activity. As part of the transformation into a ProSiT sequence diagram, this activity is denoted as a main activity. In the course of normalisation, the activity is converted into a supporting activity in order to ensure that the normalisations are carried out correctly. The object of the resulting activity is the patient, as the object appears in the function description. This latter object is removed from the label during the transformation, forming the activity 'call into the examination room'.

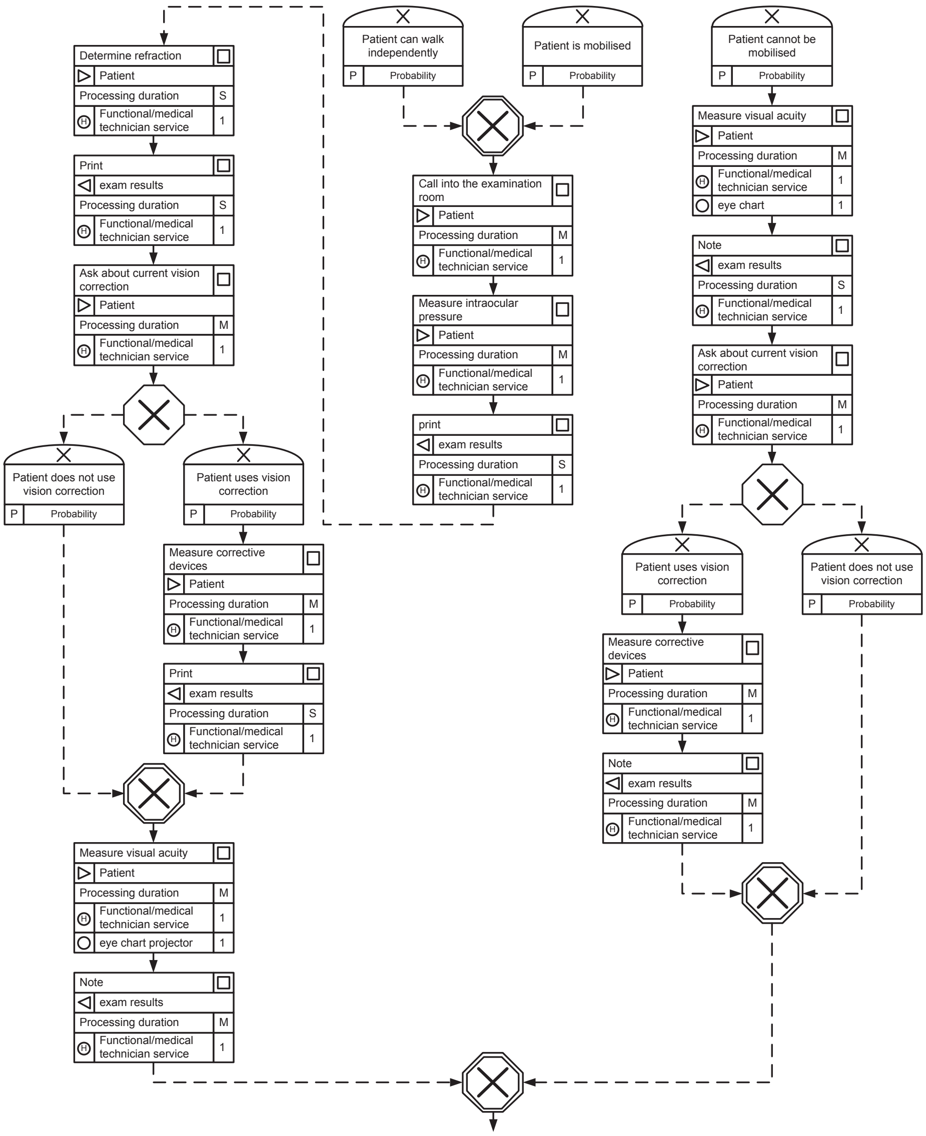

Figure 8: Pre-examination process as a transformed conceptual ProSiT sequence diagram

Linguistic transformation rules attempt to derive content from function labels, which are required as part of the normalisation for automatic or semiautomatic normalisation rules. As previously stated, an additional manual check is still required because the labels are dependent on the modeller or the modelling guidelines. The conclusion here is that linguistic rules can only be regarded as supplemental.

4.2 Normalisation of the Transformation Model

A step always occurring when simulation is intended is the determination of the duration for individual processes. An approach is presented below for supporting the process of data acquisition. In the process segment presented here, times must be determined for eight activities. Within the scope of the complete eye clinic model, 212 individual activities can be differentiated. It must be decided for each activity how exactly to record duration. Four possible alternative methods are available. Times can be exported from a system (E_{sys}), if they have already been determined (e.g., duration between incision and suturing in the operating theatre). If the activity times have not been saved in a system, they must be measured (E_{meas}), either by having it recorded in a form by those who carry it out or by an outside observer. An example of this type is taking a medical history. For other activities (e.g., creating a diagnosis), it may be sufficient to survey the executors (E_{surv}). Activities containing 'trivial' actions, for example printing examination results, can be estimated (E_{est}).

The uncertainty of the times determined by these methods is in increasing order. Previously recorded durations, especially automatically acquired times, are always preferable to those gathered through measurement or surveying.

$$E_{sys} \succ E_{meas} \succ E_{surv} \succ E_{est}$$

A linguistic procedure is used to help decide which of the gathering methods is to be applied. The first step is to export all actions from the activities in the sequence diagrams and to subsequently extract the verbs from the actions. These form the basis for decision-making. Each activity can thus be determined using a verb. It can also be determined from the activity whether it is a main or supporting activity. For the next step, a verb table must be compiled containing an assigned gathering method for each verb, depending on the activity type. The assignment of gathering methods to the individual activities can then be completed using the verb table.

A point of criticism for this approach is the subjectivity of the gathering method assignments based on a verb. A further problem results from the decoupling of the verb from the semantic context of the activity. Also, the assignment is dependent on the type of process (core, support or management process) in which the activity, or rather the verb, is used. While the 'carrying out' of a treatment (core process) can be measured, the duration of the 'execution' of a managing activity (management process) should be determined by survey.

It can be concluded from the points of criticism that the gathering method assignments cannot be considered fixed but must additionally be checked afterward. The approach should only be considered as an aid to help accelerate the choice of the gathering method. Through previous experience in using this method, it is planned to expand the verb table to include the process type (core, support, management). Also, some verbs can be indicated as synonyms (e.g., check/examine), where synonymous verbs can be grouped together. This would simplify the maintenance of the verb table. For this, each process must be assigned to one of the three process types and the verb table must be transferred to a repository. This repository can then be used to suggest the acquisition method for other process models. This concept is illustrated in Fig. 9.

The process type and name, the activity and type, the action and the verb are all extracted from the process. This information can then be used to select the acquisition method (measurement, survey, estimation) from a matrix in the repository.

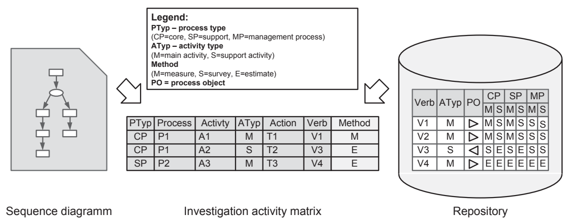

Figure 9: Approach for generating a suggestion for a data gather method

Before this concept can be applied, every activity must be looked at with regard to its significance in the process, i.e., whether it is a main or supporting activity. Also, during the normalisation missing resources must be added which are required for simulation. Both of these procedures are manual normalisation steps and should usually be completed before applying automatic and semiautomatic normalisation rules.

aNR_1
If a parallel join operator has two predecessors and one of them is a source, the source is removed and the operator is converted to a delay.

```
if pathway->predecessors->size = 2 then
 if pathway->predecessor(1) = Source or
   pathway->predecessor(2) = Source then
 remove source from flowchart
 delay = new Delay
 set pathway->predecessor->successor = delay
 set delay->predecessor = pathway->predecessor
 set delay->successor = pathway->successor
 set pathway->successor->predecessor = delay
 remove pathway
 add delay to flowchart
endif
```

Figure 10: Automatic normalisation rule aNR_1

An example of an automatic normalisation rule (aNR) independent of other normalisation rules is aNR_1 in Fig. 10. This rule serves to remove sources from the sequence diagram and is not applied to the illustrated process segment. A possible use case for this normalisation rule is ordering the radiological examination of a patient in an external (radiology) department. In the eEPC, this is an external function, which is not represented in the process model (Rosemann 1996). It must, however, be designated as a time-consuming action in the sequence diagram which does not consume internal resources. Therefore, an external function is realised as a delay using normalisation rule aNR_1.

An example of a semiautomatic normalisation rule (sNR) is shown in Fig. 11 as sNR_1. In order to apply this rule, the individual activities must have been previously classified as a main or supporting activity.

These normalisation rules were applied to the activities ‘measure visual acuity’ and ‘note’ in the top right region of Fig. 8. The result is the single activity ‘measure visual acuity’ in Fig. 12. If, after the application of the rule, the activity

sNR_1
If the successor of a main activity, which is carried out on the process object, is a supporting activity not involving the process object, query the user, whether the main activity and the subsequent supporting activity can be merged.

```
if activity->successor = Activity then
  if activity->type = primary and
    activity->object = flowchart->object and
    activity->successor->type = support and
    activity->successor->object <> flowchart->object then
      combine = ask "merge activity and successor"
      if combine = true then
        successor = activity->successor
        activity->successor = successor->successor
        successor->successor->predecessor = activity
        merge successor with activity
      endif
  endif
endif
```

Figure 11: Semiautomatic normalisation rule sNR_1

duration is to be determined, it must be taken into account that this is a merged activity.

A second semiautomatic normalisation rule sNR_2 is shown in Fig. 13. This rule checks whether sequential activities use the same resources. In the process segment in Fig. 12, this applies to the ‘functional/medical technician service’ as well as room during the pre-examination. However, the service is bound before the process segment begins but the room resource is released after the procedure represented in the process segment. Normalisation rule sNR_2 would reserve the room only until after the activity ‘Ask patient about current vision correction’. As part of manual correction, the release of the resource after this activity would be removed and manually added after a later activity.

For normalisation rule sNR_2 it becomes clear that semiautomatic normalisation rules usually require manual intervention even after being applied. These rules do not replace manual intervention, as is the case for automatic normalisation rules, but rather support the normalisation process.

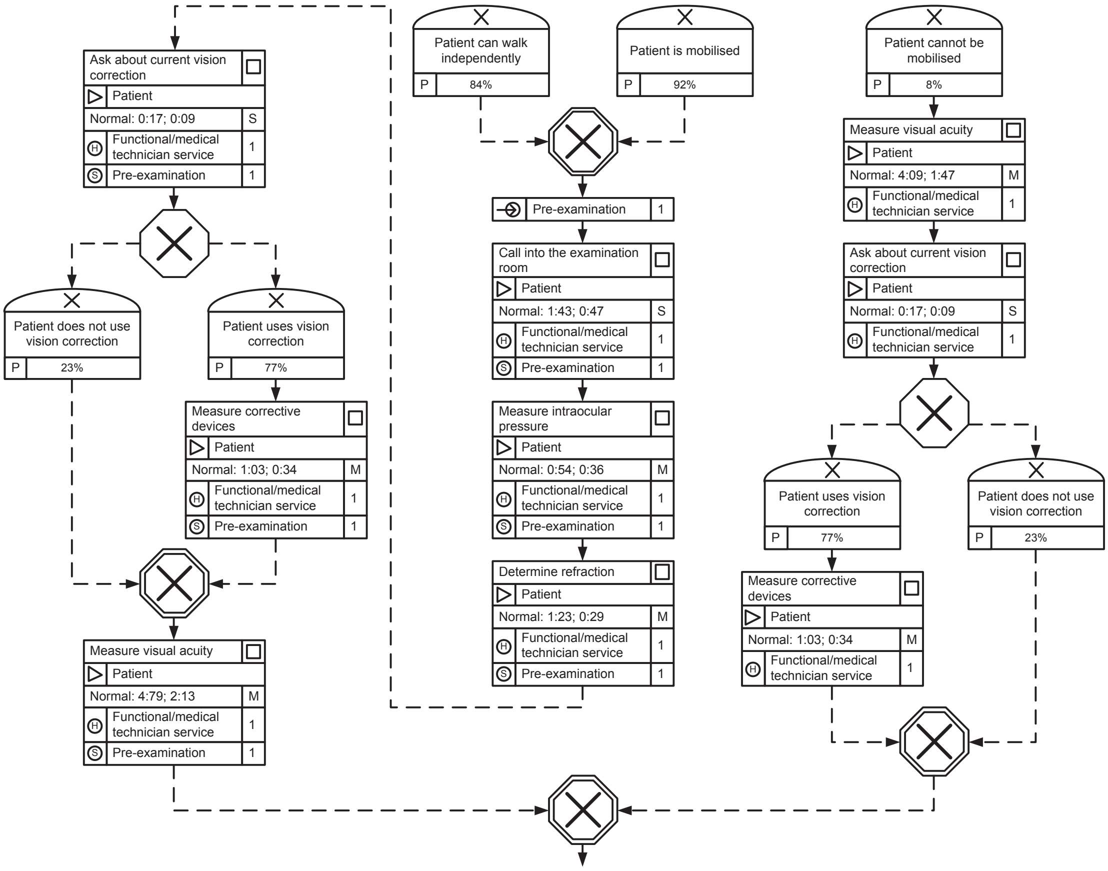

Figure 12: Normalised consistent ProSiT sequence diagram

4.3 Transfer into the Simulation Environment

The last step focussed on by the procedural model for the transformation model is the transfer of the consistent transformation model into a simulation environment. AnyLogic™ simulation software was used for the application example in the present work.

The conversion of the consistent transformation model contains two sets of rules, applicability and transformation rules. This concept is shown in Fig. 14. Applicability rules are necessary for each target simulation environment because the simulation environments support different element types in the sequence diagrams. Inclusive operators are especially prone to remain unsupported in some environments. One solution to support the inclusive operators is to convert them to exclusive operators. Every possible combination of flow paths has to be separately modelled. If this solution is used, activities that are used once in the transformation model could then be multiple times modelled in the simulation model.

In addition to this general solution to support an inclusive operator, special characteristics of the different simulation environments have to be considered. A parallel split operator in the transformation model supports more than one outgoing flow path. A corresponding element in the simulation environment AnyLogic is the Split element. However, the Split element supports only two outgoing flow paths. To model

sNR_2

If a resource is used by sequential activities, ask the user, whether the resource should be bound for all of these activities.

```
resource = activity->getResource
firstActivity = activity
lastActivity = activity
nextElement = activity
while nextElement = activity
  nextActivity = nextElement
  if nextActivity->getResource = resource then
    lastActivity = nextActivity
    nextElement = nextActivity->successor
  else
    break
endwhile

if firstActivity <> lastActivity then
  bind = ask "bind resource for first to last Activity"
  if bind = true then
    binding = new ResourceBinding
    binding->resource = resource
    release = new ResourceRelease
    release->resource = resource
    add binding before firstActivity in flowchart
    add release after lastActivity in flowchart
  endif
endif
```

Figure 13: Semiautomatic normalisation rule sNR_2

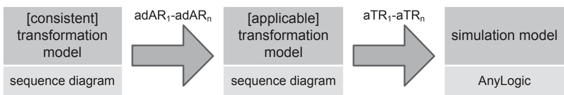

Figure 14: Concept for converting the consistent transformation model into a simulation model

a parallel split operator with five outgoing flow paths that is used in the transformation model four nested Split elements have to be used in the simulation environment. Thus, four nested Combine elements are necessary to merge the parallel flow paths. A similar solution is inevitable if an exclusive split operator has more than five outgoing flow paths.

Figure 15 shows the resulting simulation model if the transformation rules are executed on the consistent and applicable transformation model. The next step would be the validation of the simulation model. If the validation was successful, then the simulation model can be used for simulation studies.

5 Summary and Future Work

This contribution introduced a transformation concept for converting a process model into a form suitable for simulation environments. For this, a model transformation concept was explained and visualised using the example of a process segment from an outpatient eye clinic. Additionally, linguistic transformation rules were presented which can extract additional information from element labels, as was a concept for selecting a data gathering method for determining the duration of an activity. The introduction of the transformation model places significant emphasis on a central sequence diagram, which has been prepared for conversion using normalisation rules.

While process models, e.g., using eEPC or BPMN notation, can be directly simulated using modelling tools, it is not certain that all necessary information for the simulation is contained within the model. In addition, other modelling languages exist for describing service processes which cannot provide the simulation-relevant information. An example is the modelling notation for clinical pathways as found in hospitals. These generally do not contain any information about the people and devices needed to carry out activities (Sarshar and Loos 2004). The transformation concept presented here is especially useful for preparing models for simulation, which are represented in notations that do not contain the necessary information.

At present, further normalisation rules are being investigated, meaning that a set of rules is not yet available. These normalisation rules are to be evaluated in the future with the help of various process models. The additional effort expended by using the transformation model should be compensated through these normalisation rules and aids. Transformation rules for other process modelling notations and simulation environments are currently being worked on. Process modelling notations that are investigated are BPMN and the UML activity diagram.

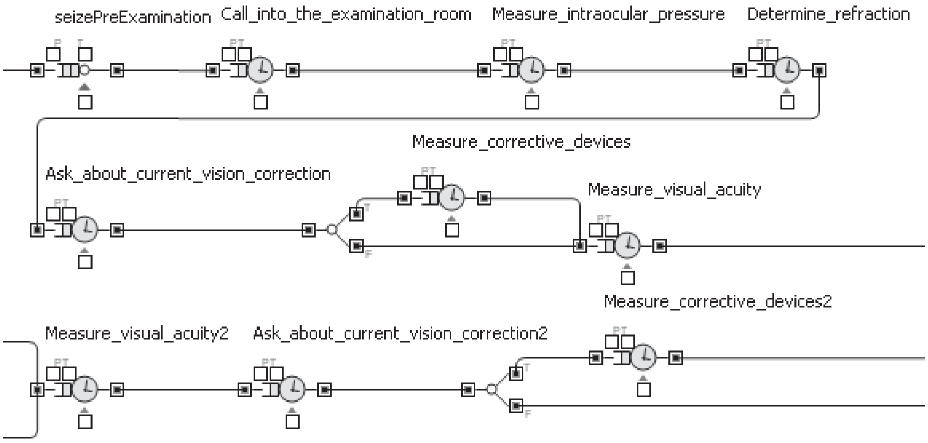

Figure 15: The resulting simulation model

Transformation rules are available for the simulation software Arena™. These transformation rules are evaluated, while transformation rules for the simulation software Plant Simulation™ are currently investigated.

Another area of research are process simulation patterns, similar to workflow patterns. On the one hand, these can be used to evaluate a process model's ability to be simulated; on the other hand, the patterns can be used to determine the feasibility of a particular process simulation in a simulation environment. The result of this investigation can then be used to create applicability and transformation rules.

Linguistic examination of process models represents a further aspect of research in the context of transformation models. Its primary goals are to determine additional information from within a model and to further aid in the creation of transformation rules.

While this contribution focused on the sequence diagram of the transformation model, further perspectives of the transformation approach have been defined. The object view defines the structure of the objects and their attributes as used in the elements of the sequence diagram, for example an activity or an attribute-based operator. Resources are defined in the resource view. Potential models for the representation of resources, especially to define resources that are used in the simulation model, are currently investigated. Another view, the task view, focuses on the linguistic transformation rules and potential normalisation rules. However, whether linguistic rules can be used to prepare a process model for simulation and/or other purposes, for example the analysis or optimisation of business processes, needs a separate investigation.

References

Allweyer T. (2007) Erzeugung detaillierter und ausführbarer Geschäftsprozessmodelle durch Modell-zu-Modell Transformationen. In: Nüttgens M., Rump F. J., Gadatsch A. (eds.) Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten (EPK2007). St. Augustin, pp. 23–38

An L., Jeng J.-J. (2005) 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, pp. 2068–2077

Becker J., Kahn D. (2003) The Process in Focus. In: Becker J., Kugeler M., Rosemann M. (eds.) Process Management – A Guide for the Design of Business Processes. Springer, Berlin, pp. 1–12

Damij N., Damij T. (2008) 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, EMS '08. Liverpool, pp. 305–310

Dickmann C., Klein H., Birkhölzer T., Fietz W., Vaupel J., Meyer L. (2007) 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, pp. 272–282

Fetter R. B., Thompson J. D. (1965) The Simulation of Hospital Systems. In: Operations Research 10(41), pp. 689–711

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

Heavey C., Ryan J. (2006) 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, pp. 801–808

Hoyer V., Bucherer E., Schnabel F. (2008) Collaborative e-Business Process Modelling – Transforming Private EPC to Public BPMN Business Process Models. In: ter Hofstede A., Benatallah B., Paik H.-Y. (eds.) BPM 2007 International Workshops. Brisbane, pp. 185–196

Kloos O., Nissen V., Petsch M. (2009) From Process to Simulation – A Transformation Model Approach. In: Esswein W., Mendling J., Rinderle-Ma S. (eds.) 3rd International Workshop on Enterprise Modelling and Information Systems Architectures. Ulm, pp. 83–96

Law A. M. (2004) Simulation modelling and analysis, 4th ed. McGraw-Hill, Boston

Mendling J., Nüttgens M. (2003) EPC Syntax Validation with XML Schema Languages. In: Nüttgens M., Rump F. J. (eds.) EPK 2003 Geschäftsprozessmanagement mit Ereignisgesteuerten Prozessketten. Bamberg, pp. 19–30

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

Polyvyanyy A., Smirnov S., Weske M. (2008) Reducing Complexity of Large EPCs. In: Loos P., Nüttgens M., Turowski K., Werth D. (eds.) Modellierung betrieblicher Informationssysteme – Modellierung zwischen SOA und Compliance Management. Saarbrücken, pp. 195–207

Rosemann M. (1996) Komplexitätsmanagement in Prozeßmodellen: Methodenspezifische Gestaltungsempfehlungen für die Informationsmodellierung. Gabler, Wiesbaden

Sadiq W., Orlowska M. E. (1999) Applying Graph Reduction Techniques for Identifying Structural Conflicts in Process Models. In: Jarke M., Oberweis A. (eds.) Advanced Information Systems Engineering: 11th International Conference, CAiSE'99. Heidelberg, pp. 195–209

Sarshar K., Loos P. (2004) Klassifikation von Sprachen zur Modellierung medizinischer Behandlungspfade. In: Rebstock M. (ed.) Modellierung betrieblicher Informationssyteme – MobIS 2004. Essen, pp. 43–59

Seyfert W., Kavermann A. (2006) Mit Simulation die Geschäftsprozesse einer Notaufnahme gestalten. In: Biethahn J. (ed.) Simulation als betriebliche Entscheidungshilfe: Neuere Werkzeuge und Anwendungen aus der Praxis. Göttingen, pp. 31–39

Oliver Kloos, Volker Nissen, Mathias Petsch, Hagen Schorcht

Chair of Information Systems in Services
Ilmenau University of Technology
Helmholtzplatz 3
D-98683 Ilmenau
Germany
{Oliver.Kloos | Volker.Nissen | Mathias.Petsch | Hagen.Schorcht}@tu-ilmenau.de