Making EPCs fit for Workflow Management

Juliane Dehnert, TU Berlin, CIS
email: dehnert@cs.tu-berlin.de

Abstract

EPCs are widely accepted for the modeling of business processes. They provide a common understanding between different participants and are therefore suitable as a starting point for Workflow management. But Workflow management comprises more than modeling, namely the analysis of the processes, their improvement and finally their support during run-time. Although the EPC tool support through the ARIS-toolset is mainly responsible for their wide use, it is still sobering which functionality is provided. Mainly, EPCs can be drawn and watched. There is no behavior analysis, only limited simulation facilities, and no execution support at all. One reason for the pure support is the disunity about the EPC semantic. In this paper we propose the use of a pragmatic interpretation of the correctness of EPCs which enables us to provide a comprehensive approach for workflow management. The proposed approach is based on the modeling with EPCs and leads the modeler to a sound process description which can be used as base for a reliable process execution at run-time.

Introduction

In recent years, workflow management was seen as one of the most promising reserves in order to straighten up their business processes. Many companies started business re-engineering projects to identify and use existing optimization potential.

Many of these projects got stuck, as the support was limited to the gathering of the existing processes, but did not provide any help thereafter. Beside the modeling of the existing business processes, a comprehensive approach for workflow management should also provide means for the analysis, the optimization, and the execution support of the described processes.

One of the core problems was the identification of a suitable modeling technique that meets an adequate abstraction of the real world and suffices the various requirements posed within the different phases: modeling, analysis, and execution support.

Also, if the focus is set on the core - the control flow aspects only, it has been found impossible to find a modeling technique that fits the purpose of the different phases adequately.

The main demands on a modeling technique within the different phases are summarized in the following:

Modeling The most important features of a modeling technique used for the first phase - the modeling, are intuitive concepts and tool support. Only, if the elements and their composition are easy to understand and to composite, the technique will win recognition at the user side.

Analysis/Optimization For the analysis, the picture is already completely different. The significant features that support analysability are formal foundation together with a theoretical background. Preferably, there should be a wide range of criteria in combination with tools that support their implementation.

Execution Support Workflow management is completed if the execution of the modeled, and possibly optimized, process is supported at run-time. This is done through a workflow management system which reacts on external events according to the internal specification of the process. The internal representation should therefore reflect a reactive behavior, guaranteeing a reliable and prompt execution.

As already said, many of the approaches used in practice, e.g.[Sch94, LR99, Mar00, JBS97] consider mainly the modeling. Accordingly, there exist many techniques that meet the requirements of this phase. One that won most recognition in the last years are *Event driven Process Chains*. One reason for their wide acceptance can be find in their use for the representation of the SAP reference models [KT97].

The objective of the approach presented is the use of EPCs as communication language between the domain experts and the application developers throughout the whole procedure. It is clear that they are used for the modeling itself. For this purpose their applicability has been proven already.

We do not use EPCs as base for the analysis nor for the execution support. Here, dedicated classes of Petri nets are employed which were developed especially for these domains and are supported through a rich theory and supporting tools. The use of Petri nets within these phases balances the lack of formal foundation EPCs suffer from. Still, the domain experts do not have to become Petri net experts as the understanding is maintained providing precise interfaces, in form of transformations, between the different techniques.

The main challenge of the proposed approach was to overcome the deficiencies of EPCs, namely their inherent ambiguity, and to come up with a sound model that meets the primary intuition of the modeler. The clue of the presented approach is the use of a relaxed correctness criteria that provides an adequate measure for the correctness of EPCs. Once the EPC is, what we call ”relaxed sound”, it is possible to transform the corresponding Petri net into a sound Petri net which then can be used to guarantee reliable execution at run-time. Within the proposed procedure the process description is developed gradually. The performed steps can be automatized.

On the remaining pages, the proposed approach will be sketched. A more detailed version can be found in [Deh02a].

A Comprehensive Process Model for Workflow Management

The proposed procedure is iterative and consists out of five steps. Support for modeling and analysis is provided through the steps one to three. The execution of the process then gets supported within step four and five. Figure 1 shows the general process model.

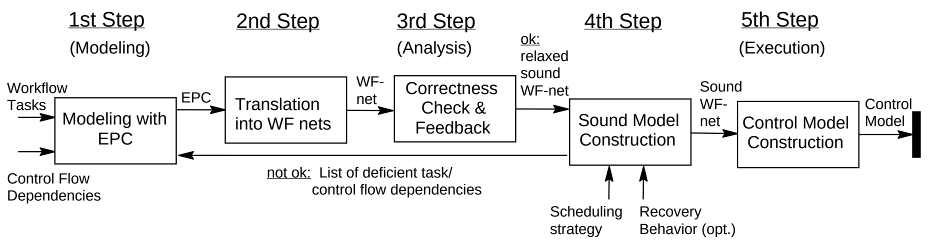

Figure 1: A process model for Workflow Management (control-flow perspective)

The five steps are: “Modeling with EPC,” “Transformation into WF-nets,” “Correctness Check and Feedback,”“Sound Model Construction,” and “Control Model Construction.” They are explained in the following.

1st Step: Modeling with EPCs

For the modeling of business processes we propose the use of Event-driven Process Chains (EPCs) of the Architecture of integrated Information Systems (ARIS) described in [Sch94]. EPCs are widely accepted in practice due to its comprehensive tool support. They provide a set of workflow modeling pattern with an intuitive graphical notation which has proven to meet the intuition of the application developer.

We assume the reader to be familiar with the EPC-notation. We therefore, do not describe their modeling features but just give some small examples that illustrate their modeling power. The examples have been chosen with respect to a current matter of dispute within the EPC-community, regarding a formalization of EPCs (c.f. [NR02, DR01, MR00, Rum99, Aal99]). The examples emphasize the lack of a formal semantic of EPCs as they contain ambiguity and vagueness. Before going on, we want to ask the reader to consider the examples and to decide whether he or she thinks the details may depict correct behavior or not. The first picture (c.f. Figure 2:a) is a detail of a larger EPC from [LSW98] and addresses the first controversial issue, namely the semantic of the OR-connector. The second detail (c.f. Figure 2:b) was found within the modeling results of a SAP R/3 implementation project. It addresses the question whether EPCs have an operational semantic or not. The third detail (c.f. Figure 2:c) finally contains a combination of EPC-elements that was precluded in the first description [KNS92].

Within this paper, we represent the opinion that these examples comprise correct behavior. Therefore, we start from the assumption that EPCs itself do **not** have an

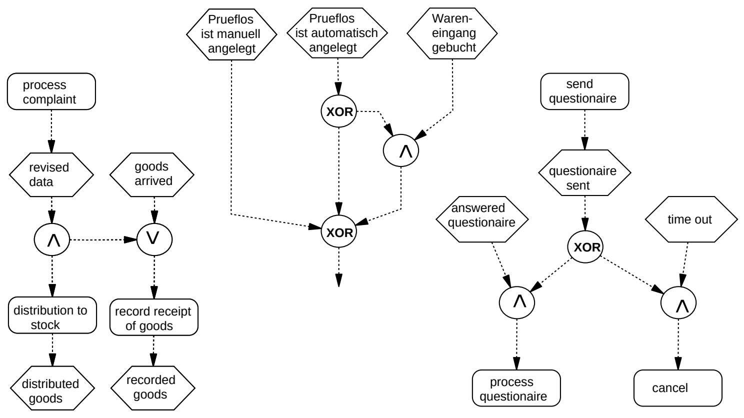

Figure 2: (a) An EPC with an OR-connector (b) One event only occurs together with another one (c) an XOR-connector following an event

operational semantic but just describe desired executions. This means, an EPC can be interpreted as a pattern that describes accepted executions. Deficient executions are only described implicitly, as the set of executions which do not fit the described pattern.

We claim that, the non-operational semantic fits an intuitive modeling understanding and is one reason why EPCs are said to be easy to learn and to understand. Modeling business processes, the modeler normally starts by describing ”What the execution should look like”, hence describes a set of good/accepted executions. This procedure fits the non-operational semantic. Modeling then means not to think about all possible executions but just specify the set of accepted executions.

We will explain the non-operational semantic again by the help of an example. Figure 3 a) shows an EPC modeling the process ”Handling of incoming order”. It represents a reduced version of a real life process of a telephone company. The process models the ordering of a mobile phone which involves two departments: the accountancy handling the payment and the sales handling the distribution.

The process starts with the event *new order*. After that, the execution is split into two parallel paths (AND split), the right one models the accounting, whereas the left one models the sales. In the accounting, the customer standing is checked (*check_credit*) first. The result of this task is either *ok* or *not_ok*. In case the result is positive, the payment is arranged (*arrange_payment*), in the latter case the instance is canceled.

The left path models the tasks on the sales side. Here, the order is either handled executing tasks *pick*, *wrap* and *deliver*, or *cancel*.

The two AND-connectors at the end make sure that only executions are accepted where both sides, the accountancy and the sales department, either cancel the instance or proceed the order. The process ”Handling of incoming order” is finished by archiving information on that instance (*archive*).

An accepted execution of the EPC from Figure 3 is *check_credit, arrange_payment,*

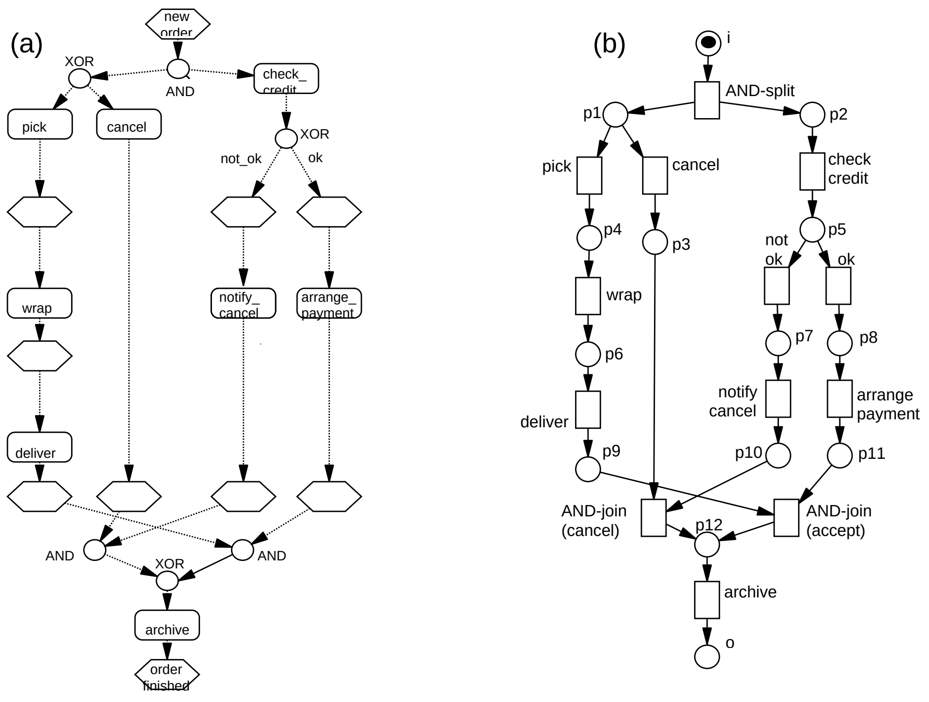

Figure 3: Example: ”Handling of incoming order” a) EPC b) WF-net

pick, wrap, deliver, archive but also *cancel, check_credit, notify_cancel, archive*.

The EPC only describes executions where the two departments work together correctly: they either both accept the order or both reject it. Hence, the execution *check_credit, notify_cancel, pick, wrap, deliver, archive* is not described by the EPC.

The EPC does not determine "how" the accepted executions are achieved. It does not stipulate the order of the two possible choices. It, therefore, accepts executions where the two departments work in parallel as well as executions where the two departments work sequentially. In an early design phase, this abstraction is appreciated, because it relieves the designer thinking about efficiency aspects of the execution at this point in time.

We hope that, under the assumption of a non-operational semantic, everybody will agree that the EPCs from Figure 3 describe useful behavior and should therefore be considered correct.

In the next step of the proposed procedure, we will provide a formal semantic for EPCs by mapping them to Petri nets. The speciality about this transformation is, that the ambiguity of the EPC is preserved but made explicit. Correspondingly, the resulting Petri net will not satisfy traditional correctness measures. We will see that, deadlocks and/or spare tokens may occur. We, therefore, provide an adapted correctness criteria, which takes this into account.

2nd Step: Translation into WF Nets

We suggest to transform the EPCs into a formal specification based on Petri nets. As suitable Petri net type we chose Workflow nets (WF-nets), cf.[Aal98]. The transformation of EPCs into WF-nets was introduced in [DR01]. It takes place in several steps. First, elements of the EPC are mapped onto Petri net-modules. Events and functions are transformed into places and transitions respectively including in- and outgoing arcs. Routing constructs such as *AND split, AND join, XOR split, XOR join, OR split* and *OR join* are mapped to small net-pattern. The net-pattern describe the behavior of the routing constructs explicitly. This is primarily relevant for the OR, because its semantic has not been described consistently. We exemplarily show the translation of Figure2:a) containing the *OR join*. The EPC as well as the Petri net have the semantics that C can be reached if either A or B or both occur. In the EPC, all these different cases are described through one connector whereas in the net-pattern all possibilities are modeled explicitly via the transitions t_A, t_{AB} and t_B. The behavior of the EPC and the Petri net are equivalent, because both accept the same executions. Note that the case that C is reached twice if A and B occur sequentially has not been excluded.

After mapping the EPC-elements to net elements or net-pattern, the different parts are combined to form a complex process model. If the derived Petri net has more than one input and/or output place, one further step becomes necessary. In that case, the net is enhanced, such that a WF-net is obtained. The transformation is well-defined. Applying the proposed rules each EPC is assigned to exactly one WF-net. Figure 3 b) shows the application of the rules to the EPC from Figure 3.

Petri nets do have an operational semantic. A Petri net model also describes "how" an execution is reached. Where an EPC only describes a set of accepted executions,

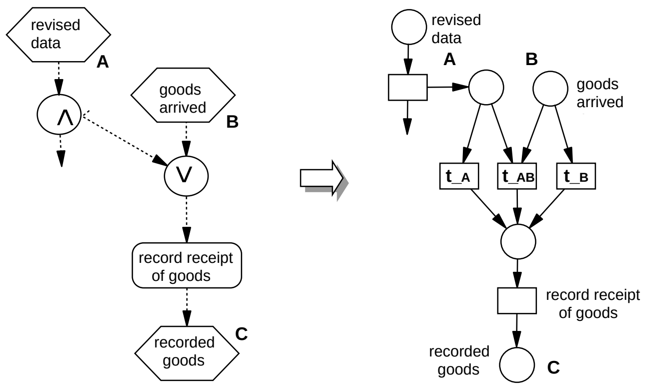

Figure 4: Translation of the OR-connector

a Petri net describes all possible behavior. This difference was neglected in previous attempts of mapping EPCs to Petri nets. As a result, all EPCs have been considered malicious if the corresponding Petri nets were deficient, cf. [Aal99, LSW98]. As a consequence, the modeling facilities of EPCs have often been restricted, such that only correct Petri nets have been gained through the transformation. Referring to the examples from Figure 3 and Figure 2, previous approaches would reject the process specification because of deficiencies of the corresponding Petri net, c.f. Figure 3. In the following, we will see that the Petri net is vulnerable to deadlocks.

3rd Step: Correctness Check and Feedback

As WF-nets are a special type of Petri nets, there exist a lot of analysis techniques that can be applied. Beside the check of reasonable correctness criteria, such as boundedness we propose to check the resulting net for relaxed soundness as introduced in [DDGJ01]. Relaxed soundness is an alleviated criteria which has been derived from Soundness [Aal98]. It only checks for some minimum requirements the resulting WF-net should satisfy. Relaxed soundness only requires to find enough sound firing sequences, where enough means at least so many that each transition is contained in one of them. If the WF net is not relaxed sound, it is not sound either, as soundness implies relaxed soundness. In the context of modeling and transforming EPCs, relaxed soundness is more pragmatic than soundness. Relaxed Soundness provides a measure for good EPCs, as the corresponding WF net is checked to cover some reasonable behavior.

The results from the correctness check of the WF-net can directly be transferred to the primary model. If the result of the relaxed soundness check is positive, we can conclude that the EPC represents reasonable behavior. If the result is negative, the modeler gets a list of transitions which are not contained in any sound firing sequence. According to the proposed transformation the deficient transitions either corresponds to a task or to a connector within the EPC. This means that either the task or one of the possible choices described by a connector are not included in an execution that terminates properly. It can be concluded that the corresponding part in the EPC needs improvement. This way, a precise feedback is provided which will helps the modeler to improve the process description until the corresponding WF-net fits the property.

The check for relaxed soundness has been automated. The criterion can be checked with the help of existing Petri net tools, such as LoLA [Sch99] (Low Level Petri Net Analyzer) or Woflan [VA00].

Running Example

The resulting WF-net is relaxed sound. A set of sound firing sequences which contains all transitions is:

1. $AND\_split, pick, wrap, check\_credit, ok, deliver, arrange\_payment,$ $AND\_join\ (accept), archive$ and
2. $AND\_split, check\_credit, not\_ok, notify\_cancel, cancel,$ $AND\_join\ (cancel), archive.$

The resulting WF-net is not sound as there exist firing sequences that deadlock, e.g.:

- $AND\_split, pick, wrap, check\_credit, not\_ok, deliver, notify\_cancel.$

WF-nets Containing OR-Subnets

If the primary EPC contained an OR-connector it is transformed as described in Figure 4. The resulting WF-net will only be relaxed sound if all three transitions (t_A, t_{AB} and t_B) are contained in some sound firing sequence.

In most cases, the OR-connector was used to express the combination of one AND-connector (t_{AB}) and one XOR-connector (e.g.t_B), the second XOR-connector (t_A) often does not express a desired choice. However, the relaxed soundness check will detect the failure prone alternative and disclose it to the modeler. The feedback could be enhanced proposing to replace the OR connector through e.g. an AND connector and a XOR connector. After that replacement, the translation of the revised EPC will be relaxed sound.

With the second step, modeling and analysis is finished. The results are an EPC describing reasonable behavior and a relaxed sound WF net. But Workflow management comprehends more than modeling and analysis. Its final objective is the execution support of the process at run-time. In order to arrive at this destination, it is necessary to generate a process description that can be used as input format for a workflow management system guaranteeing a reliable execution at run-time.

The generation of such a process description is the objective of the remaining steps within the proposed process model.

4th Step: Sound Model Construction

For the process description which is used as base for the execution support, we again suggest the use of Petri nets. Properties that advice this choice are their precise formal foundation in combination with their operational semantic. Note, that it is not possible to just use the derived relaxed sound WF-net as input format for the workflow management system. Relaxed soundness only states that at least all intended behavior has been described correctly, but it does not guarantee that malicious executions do not exist, as the resulting WF-net must not be sound.

As the workflow management system shall guarantee a reliable execution, it must operate on a sound model. In the next step of the proposed procedure we therefore concern the construction of a sound model starting from the relaxed sound specification that has been reached so far. The required model is an extension of the relaxed sound WF-net restricting its behavior to sound firing sequences only. This is done by introducing, what we call, synchronization pattern. Through their incorporation a certain scheduling strategy becomes implemented. Thereby the efficiency of the execution becomes determined.

Synchronization pattern refer to additional places that restrict the behavior according to a desired strategy. For its computation we partially refer to the results of Petri net controller synthesis [Giu96, CDLX02, GRX02], where according algorithms were proposed. The used algorithm works on the reachability graph of the WF-net. It first computes the maximal permissive behavior. At this, it determines the set of forbidden state transitions[1]. After that, it computes additional places, which disable the forbidden state transitions. The first part of the algorithm was provided in [Deh02b], whereas the second part has been implemented in the tool Synet[2] [Cai97].

Decide Upon a Strategy

Possible are optimistic and pessimistic strategies. Following a pessimistic scheduling strategy, deadlocks are avoided by waiting for decisions to be taken in advance. This would be implemented synchronizing both processes at the decision points. Figure 5 a) shows a sound model implementing a pessimistic scheduling strategy for the running example. The process model has been enhanced by two places *S1, S2* and corresponding arcs. Through the introduced pattern the two transition *ok* and *pick*, and transition *not_ok* and *cancel* become synchronized. Using this sound model as input for the workflow management system the execution of the process becomes serialized. The customer check would always been executed before the ordering. Here all sound but parallel firing sequences of the relaxed sound WF-net have been eliminated.

In case the delivery process takes a long time and the customer check takes a long time, this would be very inefficient. This is especially undesirable if it is very rare that a customer check results in a *not_ok*. Then this way of pessimistic scheduling would be annoying. It would be more efficient to just start the delivery of the order to the customer hoping the customer check will be *ok*, i.e. following an optimistic approach. Only, in the rare case that the decision *not_ok* was taken, the order should be returned to stock and should be canceled after all.

To build an optimistic sound model an additional intermediate step: the specification of the recovery behavior, becomes necessary. This step can not be provided

[1]State transition which do not lead to state ”o”

[2]Synet: A Synthesizer of Distributable Bounded Petri-Nets from Finite Automata, Version 2.0b, http://www.irisa.fr/s4/tools/synet

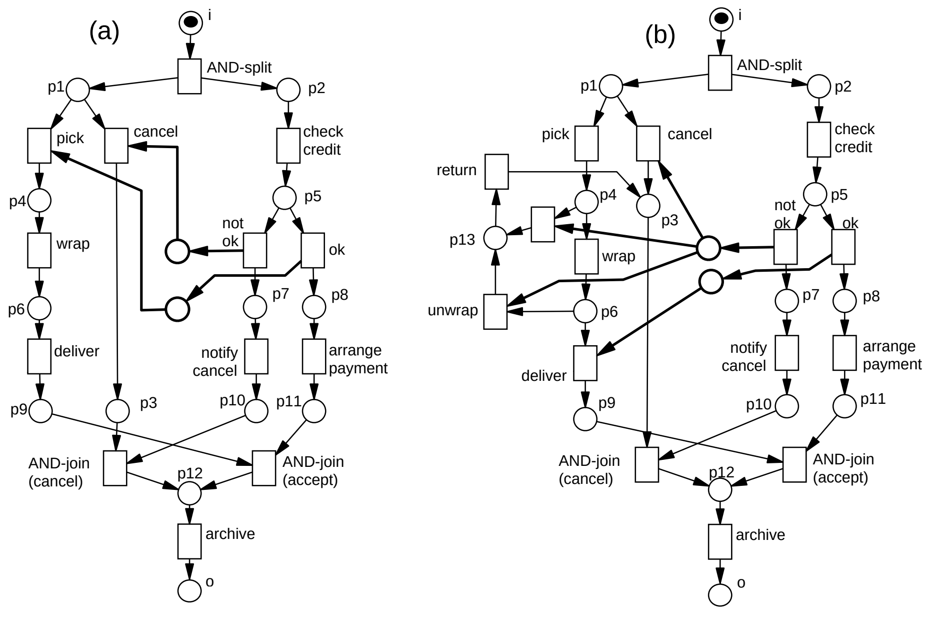

Figure 5: An implementation of a pessimistic (a) and an optimistic (b) scheduling strategy

automatically, as it requires background knowledge about the process and its context. The domain expert has to determine: from which states recovery is possible and not too expensive, what has to be done for the recovery and which are the points where the process can start again. These information can be gained from long-term observation of the process or by simulation of the relaxed sound process model. In the latter case, the model should be enhanced by some probability distributions at the decisions points and appraisals of activity duration and -costs.

As results of this investigation, the modeler enhances the EPC model incorporating the recovery behavior. The enhanced model will be used then to build the sound model implementing an optimistic scheduling strategy.

For the running example, we assume that tasks *pick* and *wrap* can be reset without extraordinary charges, whereas task *deliver* is considered to be nonreversible. The compensation tasks are *return* and *unwrap*. After the item has been returned to stock, the instance should be canceled.

The enhanced EPC model incorporating the recovery behavior as well as the corresponding WF-net are shown in Figure 6.

The implementation of the optimistic scheduling strategy under these assumptions is shown in Figure 5 b).

Using this optimistic sound model as input for the workflow management system, the two departments can operate in parallel. The customer check of the accountancy can be executed concurrently to preparative tasks of the sales handling. Here, all sound

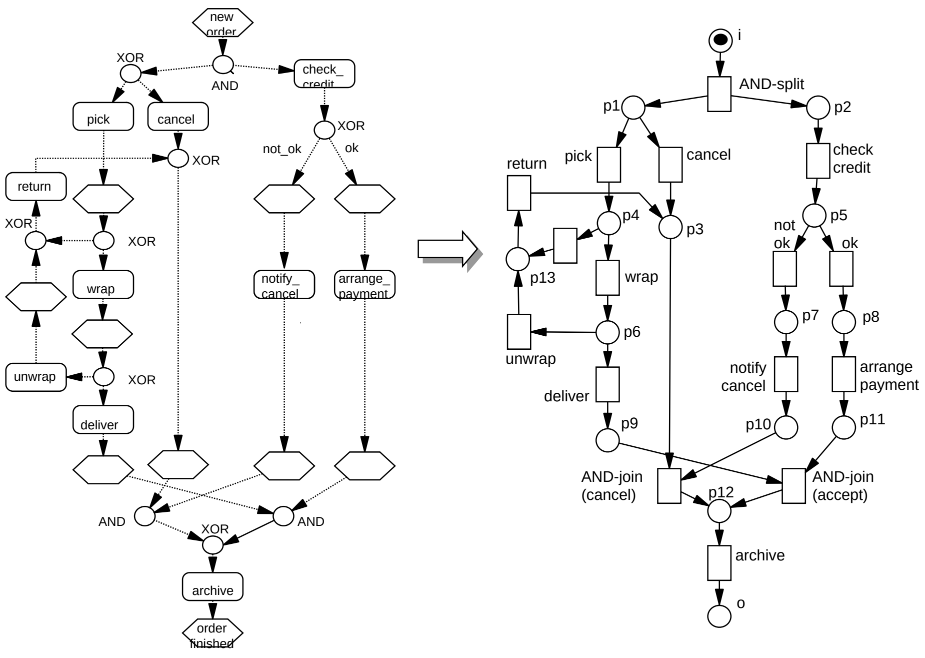

Figure 6: Enhanced process models incorporating possible recovery behavior

firing sequences of the relaxed sound WF-net are maintained. As consequence, some additional and not so effective executions are accepted as well.

Both models, the pessimistic as well as the optimistic, are sound. There exist no firing sequences that deadlock or do not terminate properly. Furthermore there exists no dead task. Using either of these models as base fot the internal process description of a workflow management system a reliable process execution at run-time can be guaranteed.

Feedback for the EPC Experts

The computation and integration of the synchronization patterns works on the base of a relaxed sound WF-net. Once, the synchronization pattern have been computed and added to the WF-net, it would be desirable to give the EPC expert feedback about the introduced changes. Therefore there should be a translation from WF-nets back to EPCs. This backward translation is an issue of current research. It is clear, that the rules provided for the EPC$\rightarrow$PN direction cannot simply be applied in reverse order. Difficulties arise with respect to the demarcation of net-pattern that correspond to connectors, as well as the translation of choices that are not free-choice. Their translation is not bijective.

Coming back to the process model, some adoptions remain in order to feed the resulting sound process description to a workflow management system. The last step on the way towards the final process description, which we will call *control model*, concerns the incorporation of requirements imposed through the reactive character of a workflow management system.

5th Step: Control Model Construction

The workflow management system is a reactive system, thus it runs in parallel with its environment and tries to enforce certain desirable effects in the environment. To be more precise the workflow management system monitors activities performed by actors (people or software). Monitoring comprises activating tasks by assigning them to certain actors and waiting for the tasks to complete (completion event). Which tasks are enabled at a certain point in time depends on the workflow model describing the process perspective.

WF-nets which have been used so far have an active behavior. This is introduced through the following three facts:

1. Tasks are modeled through transitions, which are the active parts within the Petri nets semantic.
2. Transitions fire instantaneous. This does not match with the requirement to model tasks performed by external actors as time consuming entities.
3. The MAY-firing rule, which introduces unintended non-determinism leaving open either to execute an enabled task or to defer its execution.

In the last step the WF-net is transformed into a *reactive* WF-net, which is a Petri net that can serve as input format for a workflow management system, specifying what the workflow engine should do. Therefore, transitions modeling tasks get refined, and the May-firing is changed towards Must-firing in the corresponding cases. The changes do not compromise the validity of the proven properties. Finally, a sound model with a reactive behavior is obtained. This process description can be used as input format for a workflow management system and will guarantee reliable execution at run-time.

Benefit of the Proposed Procedure

In this paper, a comprehensive process model for the control-flow perspective of workflow management was sketched. The benefit of the proposed approach is obvious. It supports the management of business processes through all its phases, as modeling, analysis and execution, bridging the gap between low modeling expertise on the modeler side and high requirements on the system design side.

The procedure starts with the modeling of processes using the widely accepted EPCs. In contrary to previous approaches it supports the whole spectrum of control-flow elements. Particularly, the use of the OR-connector is not forbidden although ambiguities may become introduced. It, furthermore, does not restrict the modeling to well-formed EPCs only. Following the EPC semantic, the modeler is not required to think about all possible behavior but only has to specify the accepted executions. Thereby, the modeler is relieved from thinking about efficiency aspects during the first phases of the development process.

The correctness means that are provided during the analysis phase have been adapted to these prerequisites. Furthermore, the feedback the application developer gets from the tests is precise. It exactly points at these parts of the model where deficiencies, that occur during the execution, stem from. The requirement posed through the modeling are not very strict. This becomes well-balanced again through the provided execution support. In the last steps, the present model, which possibly also describes undesired executions is enhanced automatically to a sound model. Only here, efficiency aspects get introduced. To postpone the efficiency aspect determination to the end of the procedure is especially desirable as corresponding informations, e.g. the occurrence probability of a certain failure, costs of failure compensation, or priorities often only get available or even change during run-time. Their late incorporation extends the possibilities to reuse modeling results under changing priorities.

The resulting description can be used as base for the input format of a Workflow management system. Giving it a reactive semantic the workflow engine can operate on the derived process description, guaranteeing a reliable and prompt execution at run-time.

References

[Aal98] W.M.P. van der Aalst. The Application of Petri Nets to Workflow Management. *The Journal of Circuits, Systems and Computers*, 8(1):21–66, 1998.

[Aal99] W.M.P. van der Aalst. Formalization and Verification of Event-driven Process Chains. *Information and Software Technology*, 41(10):639–650, 1999.

[Cai97] B. Caillaud. Synet : A tool for the synthesis of bounded petri-nets, applications. Technical Report RR-3155, Inria, Institut National de Recherche en Informatique et en Automatique, 1997.

[CDLX02] B. Caillaud, P. Darondeau, L. Lavagno, and X. Xie, editors. *Synthesis and Control of Discrete Event Systems*. Kluwer Academic Press, 2002.

[DDGJ01] W. Derks, J. Dehnert, P. Grefen, and W. Jonker. Customized atomicity specification for transactional workflow. In H. Lu and S. Spaccapietra, editors, *Proceedings of the Third International Symposium on Cooperative Database Systems and Applications (CODAS'01)*, pages 155–164. IEEE Computer Society, April 2001.

[Deh02a] J. Dehnert. Four steps towards sound business process model. In G. Rozenberg and H Ehrig, editors, *PNT-Volume*, LNCS, Advances in Petri Nets. Springer Verlag, 2002. to appear.

[Deh02b] J. Dehnert. Non-controllable choice robustness: Expressing the controllability of workflow processes. In J. Esparza and C. Lakos, editors, *ICATPN 2002, 23rd Int. Conf. on Application and Theory of Petri Nets*, volume 2360 of *LNCS*, pages 121–141. Springer Verlag, 2002.

[DR01] J. Dehnert and P. Rittgen. Relaxed Soundness of Business Processes. In K.L. Dittrich, A. Geppert, and M.C. Norrie, editors, *Advanced Information System Engineering, CAISE 2001*, volume 2068 of *LNCS*, pages 157–170. Springer Verlag, 2001.

[Giu96] A. Giua. Petri net techniques for supervisory control of discrete event systems. In *First Int. Work. on Manufacturing and Petri Nets*, pages 1–3, June 1996.

[GRX02] A. Ghaffari, N. Rezg, and X. Xie. Net transformation and theory of regions for optimal control of petri nets. In *15th Triennial World Congress of the International Federation of Automatic Control*, Barcelona, Spain, 2002.

[JBS97] S. Jablonski, M. Böhm, and W. Schulze. *Workflow-Management; Entwicklung von Anwendungen und Systemen*. dpunkt.verlag, Heidelberg, 1997.

[KNS92] G. Keller, M. Nüttgens, and A. W. Scheer. Semantische Processmodellierung auf der Grundlage Ereignisgesteuerter Prozessketten (EPK). Veröffentlichungen des Instituts für Wirtschaftsinformatik, Heft 89 (in German), University of Saarland, Saarbrücken, 1992.

[KT97] G. Keller and R. Teufel. SAP R/3 prozessorientiert anwenden – iteratives Prozess-Prototyping zur Bildung von Wertschpfungsketten, 1997.

[LR99] F. Leymann and D. Roller. *Production Workflow: Concepts and Techniques*. Prentice Hall PTR (ECS Professional), Upper Saddle River, 1999.

[LSW98] P. Langner, C. Schneider, and J. Wehler. Petri net based certification of event driven process chains. In J. Desel and M. Silva, editors, *Application and Theory of Petri nets*, volume 1420 of *LNCS*, pages 286–305. Springer Verlag, Berlin, 1998.

[Mar00] C. Marshall. *Enterprise Modeling With UML: Designing Successfull Software Through Business Analysis*. Addison Wesley, Reading, Massachusetts, 2000.

[MR00] D. Moldt and J. Rodenhagen. Ereignisgesteuerte Prozessketten und Petrinetze zur Modellierung von Workflows. In *Visuelle Verhaltensmodellierung verteilter und nebenläufiger Software-Systeme*, volume 24/00-I of *Fachberichte Informatik*, pages 57–63, 2000.

[NR02] M. Nüttgens and F. J. Rump. Syntax und Semantik Ereignisgesteuerter Prozessketten (EPK). In J. Desel and M. Weske, editors, *Prozessorientierte Methoden und Werkzeuge für die Entwicklung von Informationssystemen*, LN in Informatics. GI-Edition, 2002.

[Rum99] F. J. Rump. *Geschäftsprozessmanagement auf der Basis ereignisgesteuerter Prozessketten. Formalisierung, Analyse und Ausführung von EPKs (in German).* Teubner, Stuttgart, 1999.

[Sch94] A. W. Scheer. *Business Process Engineering, ARIS-Navigator for Reference Models for Industrial Enterprises*. Springer-Verlag, Berlin, 1994.

[Sch99] K. Schmidt. Lola: A low level analyser. In *Proc. Int. Conf. Application and Theory of Petri net*, volume 1825 of *LNCS*, pages 465–474, 1999.

[VA00] H.M.W. Verbeek and W.M.P. van der Aalst. Woflan 2.0: A petri-net-based workflow diagnosis tool. In M. Nielsen and D. Simpson, editors, *Application and Theory of Petri Nets*, volume 1825 of *LNCS*, pages 475–484. Springer, 2000.