Coordinating Planning Processes in AEC using an Adaptable Process Model

Rolf Katzenbach, Johannes Giere,
Institute of Geotechnics, Technische Universität Darmstadt, 64287 Darmstadt, Germany
sekretariat@geotechnik.tu-darmstadt.de

Summary

The contribution introduces an adaptable process model to meet the special requirements of the coordination of planning activities in AEC (Architecture, Engineering, Construction). The process model is based on the concept of Coloured Petri-Nets and uses metainformation to characterize process-relevant information and to enable process-control based on the actual results of the planning.

1 Introduction

Coordinating planning processes is one of the greatest challenges in the realization of projects in AEC (Architecture, Engineering, Construction). With the increasing number of specialists participating in the planning process the coordination of all activities becomes more and more crucial to the cost- and time-effective project-management.

Process-modelling as a means to analyze, model and coordinate different activities in a project has been in the focus of various research activities in the last years. The problem to adequately represent the a priori only partly known planning process and its changes in a process model has not been solved yet, however. To take into account the changes based on the results of the actual planning and to enable permanent adaption of the process model is within the focus of the presented approach. The paper is based on the results of the research activities, which are carried out in cooperation with the Institute of Numerical Methods and Informatics in Civil Engineering at the Technische Universität Darmstadt.

2 Process-Modelling with Petri-Nets

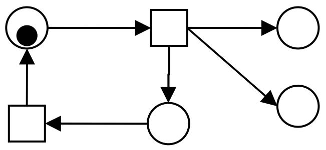
A variety of approaches to process-modelling have been developed to describe, analyze and control processes in the last years with different focuses. Methods such as UML, IDEF0 and Event-driven process-chains (EPCs) are frequently used as a means to visualize and better understand processes. In the presented approach however the method of Petri-Nets is used for process-modelling, as it has several advantages (see e.g. [v.d.Aalst 1996], [Rueppel et al. 2004]). The main advantages for this approach are the presence of a formal semantic, which enables the planning information to be passed through the net and to control the planning process based on the information itself. Also an abundant number of analysis and verification methods are available.

Petri-Nets were first formulated by C.A. Petri [Petri 1962]. They consist of the disjoint finite set of places P and transitions T as nodes of the graph, which are connected with directed arcs as a flow relation F. The state of the system is given by the marking M, which is represented by tokens in places. By firing a transition, the token is subtracted from the input place of the transition and added to the output place and the net is thus transformed into the next state

Formally the basic form of a Petri-Net is defined as a Tuple PN = (P, T, F, M_0), where:

P	is a finite set of places	$P = \{p_1, p_2, \ldots, p_m\}$
T	is a finite set of transitions	$T = \{t_1, t_2, \ldots, t_n\}$
F	is a set of arcs as flow relations	$F \subseteq (P \times T) \cup (T \times P)$
M_0	is the initial marking	$M_0: P \rightarrow \{0, 1, 2, 3, \ldots\}$
		$(P \cap T) = \emptyset$ and $(P \cup T) \neq 0$

For a good introduction to the Petri-Nets the reader is referred to [Reisig 1985]; a very good comprehensive report can be found in [Murata 1989]. The usual graphical representation of the elements of the Petri-net is shown in Fig. 1.



Petri-Net		Planning Process
○	Place $p \in P$	State
□	Transition $t \in T$	Activity
→	Arc $A \in F$	Dependency
●	Token	Information

Fig. 1: Elements of Petri-Nets and the corresponding representation in the planning process

With reference to the work by van der Aalst [v.d.Aalst 1997, v.d.Aalst 2002] the process is represented by activities as transitions and planning states as places. The whole planning process is represented as a set of planning activities and states, which are connected to each other by logical dependencies (arcs). Figure 2 shows the representation of routing primitives in Petri-nets, which underlie process modelling according to the WFMC [WFMC 1996].

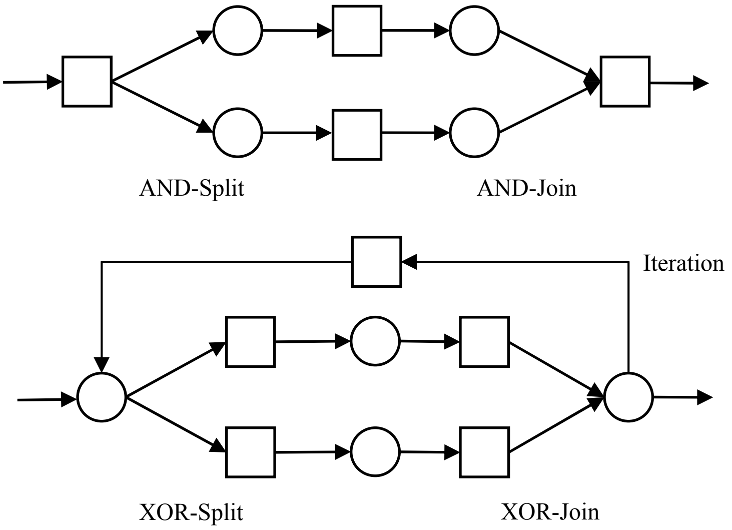


Fig. 2 Routing primitives

With respect to the requirements of process modelling of business processes, v.d. Aalst gives a formal definition of Petri-nets as workflow nets [v.d. Aalst 1998], defining as well a soundness criterion. According to it, a workflow net (wf-net) has one single input place *I* and one single output place *O*. When the wf-net is short-circuited by a transition t*, it is strongly connected. When the output place *O* is marked with a token, all other places in the wf-net must be empty, so no other activities can be carried out (no transitions are enabled to fire). This prevents, that a task has erroneously not been carried out, although the process should already be terminated. In this case the process has not been modelled correctly.

In AEC the set of tasks, i.e. the planning activities, is unknown at the beginning but evolves with the proceeding planning, as more and more information is gained and planning decisions are taken. Dynamic change of the underlying process-model is therefore required to enable control of the planning process.

Within the activities the planning information is generated or modified. Depending on the actual planning state the further planning activities have to be carried out. For example the number of basement floors which are designed by the architect imply the required construction for the retaining walls. In order to be able to take into account the exchanged information and the results of each planning activity in the planning process and its effects on the process-model, the Petri-Nets are extended with a formal semantic to individualize the tokens.

The theory of Coloured Petri-Nets [Jensen 1996] expands the formal semantics of the Petri-nets by adding colours to the tokens and a formal semantic to perform operations on the individual tokens. Thus it is possible to transport information through the Petri-Net and define operations based on this information. Guard functions on the transitions can be arbitrarily complex to formulate conditions to fire the transition, i.e. carry out the activity.

In the following section the definition of the individual token as information container for construction specific information is introduced.

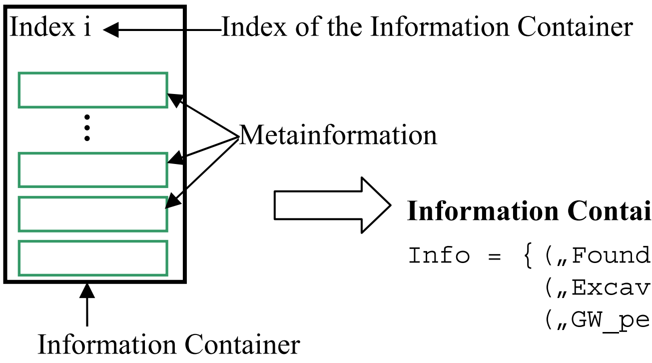
3 Concept of Metainformation

Information exchange in AEC is still mainly document based. From the abundance of information generated in a planning process, only a small part is relevant for the control of workflow. The idea of the metainformation is therefore to abstract the process-relevant information from the exchanged information [Katzenbach et al. 2004] and transform it into a form, which is accessible for the process-model. Table 1 shows the identified requirements from process-modelling in the left column and the developed implementation in the right column.

Requirements deriving from Process-Modelling	**Implementation in Petri-Nets with individual Tokens**
Metainformation Single, construction specific information • label (string) • value (integer/real/string) • version (integer)	Tuple (label, value, index) Example („foundation_type“, S(„shallow“), 1) („excavation_depth“, R(6.5), 1)
Information container All metainformation necessary for the control of the process	List of tuples (Tuple 1, Tuple 2, …, Tuple n)

Table 1: Implementation of metainformation in Petri-nets with individual tokens

The found solution of the implementation in Petri-Nets is a standardized and open container for all metainformation, which is needed to control the process. It consists of a list of tuples, which each represent one particular, construction specific information. Each tuple consists of a label to identify the information, a value, which may be textual or numerical, and an index of the version of the information. The list of tuples comprises all process-relevant information and is called the information container. In order to be able to identify different information containers, which may refer for example to different parts of the construction, an identifier is added to each container. Figure 3 shows a scheme of the information container as well as some examples of Metainformation in Standard ML-Code.



Information Container in SML (Example)

```
Info = {(„Foundation_Type“, S(„shallow“), 1),
        („Excavation_Depth“, R(6.5),1),
        („GW_permission“, S(„required“),1)}
```

Fig. 3: Scheme of the Information Container in SML

The Process-model has been implemented in the Petri-Net tool Design/CPN [Jensen 1996]. All methods to operate on the Metainformation and all guards on the transitions have been developed and implemented in the functional programming language Standard ML (SML) [Milner et al. 1997].

4 Extending routing primitives to construction specific process patterns

Based on the introduction of metainformation to process-modelling the routing primitives (fig. 1) are extended. Figure 4 gives an example of two planning activities ordered in parallel (transitions t_2 and t_3). For a better understanding the declaration node, where the colors, variables and functions are defined, is provided in parts as well.

Figure 4 shows the following net inscriptions

- guards (in angular brackets) and
- arc inscriptions (in brackets)

For reasons of readability all default net inscriptions defining the type of places and the arc expressions are not included. By default the places are of type (Info,b) and arc expressions are of type (info,b), carrying the information container. Transitions shown in grey indicate a user interaction, an actual planning activity. The methods to integrate the planning participants into the process and their information exchange with the system are not shown here. Transitions shown in white are fired automatically.

```
color Int_Real_String = union I:Int + R:Real + S:String;
color Meta = Product String * Int_Real_String * Int;
color Info = list Meta;

...

var info, info1, info2: Info;
var b, b1, b2: Int;
...

(* list_union unifies two lists of Metainformation *)
fun list_union ([], list2) = list2
|list_union (hd::tl, list2) = exists_l hd (list_union(tl,list2));
```

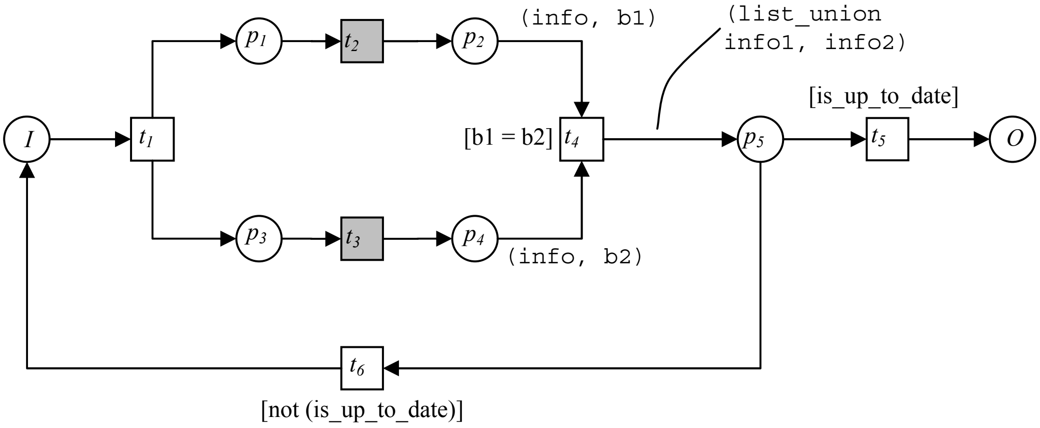


Fig. 4: Example of two parallel planning activities

The example net contains various elements, which have been introduced. There are the single input place *I* and the single output place *O*. The parallel planning activities in t_2 and t_3 are preluded by an AND-split in t_1. Both activities receive the identical Information. As a result of the activities, which may be carried out by two different planning participants, the metainformation in the information container is updated. New tuples are added to the list, the value of others already in the list may have been altered and added with a new index. In t_4 the information container has to be synchronized. The guard on transition t_4 [b1=b2] verifies, that both information-containers have the same identifier to be unified. On the outgoing arc of t4 the arc expression is the function list_union. This function synchronizes the two lists of tuples in both information-containers and checks, if metainformation, which had been passed in to the activities has been updated. In this case an iteration is initiated in place p_5 by transition t_6, as to make sure, that both activities have the same information basis.

The example depicts the essential process control mechanisms based on the information itself. Routing primitives have been extended by methods to take into account the planning results.

The adaptable approach to the process model requires furthermore, that planning activities can be integrated in the existing process [Katzenbach et al. 2002]. The approach follows the idea, that process models can be determined by each planning participant for his own domain. With the advancing planning process, more and more elements of the construction are determined. The process model and thus the planning activities are linked to the construction elements. For each construction element, one specific process model can be determined, which is called a construction specific process model.

In this context, construction elements are referred to as functional units and not necessarily physical entities. Construction elements may be ordered in hierarchies, as the following example of some geotechnical construction elements depicts:

- Building pit
  - Reataining Wall
    - Anchor
    - Piles
- Foundation
  - Slab

The process-model of the whole planning process is adapted by integrating the construction specific process elements into the existing process-model using the concept of hierarchical Petri-nets introduced in [Jensen 1996]. In this concept, transitions are replaced by subnets with the definition of so-called socket-places and port-places. To enable the substitution of planning activities, the construction specific process elements have to meet the requirements of workflow nets. Input and output places are then used as port-places. A detailed description of the aggregation mechanism is given in [Rueppel et al. 2004].

5 Conclusions

In the contribution an adaptable process model based on Petri-nets with individual tokens is introduced. It meets the special requirements of the dynamically changing planning process in AEC by the concept of construction specific process elements and their integration in the process model with port places and socket places. The concept of metainformation enables the control of the process model based on the results of the planning process. Soundness of the process model is provided by the concept of wf-nets.

6 References

[v.d.Aalst 1996]
van der Aalst, W. M. P.: Three Good Reasons for Using a Petri-net-based Workflow Management System. In: Proc. of the Internationsl Working Conference on Information an Process Integration in Enterprises (IPIC 96), 179-201, Navathe, S. and Wakayama, T. (Eds.), Cambridge, Mass., USA, 1996

[v.d.Aalst 1997]
van der Aalst, W. M. P.: Making Work Flow: On the Application of Petri Nets. Lecture notes in computer science 2360, 1-22, Springer, Berlin Heidelberg, 2002

[v.d.Aalst 1998]
van der Aalst, W. M. P.: The Application of Petri Nets to Workflow Management. Journal of Circuits, Systems, and Computers 8 (1), 21-66, World Scientific Publishing Company, Singapore u.a., (1998)

[v.d.Aalst 2002]
van der Aalst, W. M. P.: Verification of Workflow Nets. Lecture notes in computer science 1248, 407-426, Springer-Verlag, Berlin-Heidelberg, 1997

[Jensen 1996]
Jensen, K.: Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use. Vol. 1-3, Springer-Verlag, 1996

[Katzenbach et al. 2004]
Katzenbach, R.; Meissner, U.F.; Rueppel, U.; Savidis, S.; Giere, J.; Greb, S.; Mejstrik, M.: Abstraction of Process Relevant Information from Geotechnical Standards and Regulations. In: Proceedings of the Xth International Conference on Computing in Civil and Building Engineering, Bauhaus Universität Weimar, Weimar, Germany

[Katzenbach et al. 2002]
Katzenbach, R.; Meissner, U.F.; Rueppel, U.; Giere, J. und Greb, S. (2002): Process-Oriented Network-based Collaboration in Geotechnical Engineering. Proceedings of the 9th International Conference on Computing in Civil and Building Engineering, 1063-1068, National Taiwan University, Taipei, Taiwan

[Milner et al. 1997]
Milner, R.; Tofte, M.; Harper R.; MacQueen, D.: The Definition of Standard ML. The MIT Press, Cambridge, MA, USA, 1997

[Murata 1989]
Murata, T.: Petri Nets: Properties, Analysis and Applications. Proceedings of the IEEE 77 (4), 541-580, IEEE, New York, 1989

[Petri 1962]
Petri, C. A.: Kommunikation mit Automaten. Schriften des IIM 3, 1-128, Universität Bonn, Institut für Instrumentelle Mathematik, Bonn, 1962

[Reisig 1985]
Petri nets, An Introduction. In: Monographs on Theoretical Computer Science. W. Brauer, G. Rozenberg, A. Salooma (Eds.), Springer Verlag, Berlin, 1985

[Rueppel et al. 2004]
Rueppel, U.; Meissner, U.F.; Greb, S.: A Petri Net Based Method for Distributed Process Modelling in Structural Engineering. In: Proceedings of the Xth International Conference on Computing in Civil and Building Engineering, Bauhaus Universität Weimar, Weimar, Germany

[WFMC 1996]
Workflow Management Coalition Terminology and Glossary (WFMC-TC-1011), Technical report, Workflow Management Coalition, Brussels