Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

Hierarchical modelling and verification based on Petri net components with multiple import interfaces

U. Kuessel * **J. Padberg** ** **D. Abel** ***

* *Institute of Automatic Control, Technical University of Aachen, 52074 Aachen, Germany (Tel: 0241-8027479, e-mail: u.kuessel@irt.rwth-aachen.de)*
** *Institut für Softwaretechnik und Theoretische Informatik, Technische Universität Berlin, 10587 Berlin, Germany (Tel: 030-31424165, e-mail: padberg@cs.tu-berlin.de)*
*** *Institute of Automatic Control, Technical University of Aachen, 52074 Aachen, Germany*

Abstract: This paper introduces the modelling of discrete event based system and the verification of their properties using Petri net components. It is particularly interesting to apply a component based verification approach in order to hierarchically structure Petri nets and to verify their properties component-wise. Here, a new theoretical notion is exemplified that facilitates modelling. This extension allows the definition of multiple import interfaces. Multiple import interfaces allow the component to import more than one other component and so simplifies the modeller's task as it provides means for a "divide and conquer" strategy.

Keywords: Discrete event systems modelling and control, Petri Nets and other tools.

1. INTRODUCTION

Process modelling and analysis is a crucial step to controller design of technically controlled systems. A key role of the modelling procedure is to understand the process in more detail as well as to determine the needed accuracy of modelling. Nevertheless, a model is also established for the purpose of analysis and verification of system properties. In the field of discrete event based systems modelling using Petri nets is common practice due to their capability of dealing with concurrency and due to their well established analysis methods, see e.g. Abel and Bollig (2006). Large scale processes are still difficult to model without any structuring methods, it is especially error prone and uncomfortable for the modeller. Various concepts of hierarchical modelling with Petri nets exist, but usually the possibilities of analysis decrease or the structuring has only visual character and analysis has to be done for the unfolded net. A related approach is the work by Kindler and his group on component tools, see Kindler et al. (2006). This approach supports the definition of components with varied formal models and at different levels of abstraction. Moreover, it belongs to a tool level degree of abstraction and is less directed to modelling of discrete event based systems. For the field of control theory of discrete event based systems a component-based hierarchical modelling approach is suggested, applying Petri nets without the loss of important analysis and verification methods. The concept of component-based Petri nets relies on the component concepts of Continuous Software Engineering in Weber (1999) and Große-Rhode et al. (2000). According to the concept, the body of a component is extended by two interfaces, the import and the export. Hereby, the body net BOD describes the internal desired functionality of modelled subsystems, the import IMP states prerequisites of components integrated into the body net and the export EXP displays the behaviour of the body net in an abstract form. The import-export implication of the Petri net components expresses properties of the abstract export net that are guaranteed, provided the imported environment satisfies the import requirements. The import-export implication is expressed by a temporal logic formula. Therefore a suitable temporal logic calculus is needed as for example in Girault and Valk (2003). The underlying idea is that components guarantee specific properties for the export net if import assumptions are satisfied, see Padberg (2006). The hierarchical composition of components requires that the corresponding interfaces, namely EXP and IMP, coincide. Then, as shown in Padberg et al. (2007) and illustrated in Padberg and Kuessel (2007), compositional verification is given in the sense that the import-export implications can be constructed according to the composition of the components. In order to simplify this approach for the modeller multiple interfaces are developed that even may overlap. Multiple import interfaces are very useful as they allow using different components. Hence, the hierarchy is not merely a sequential but a tree-like structure. In order to preserve the advantages of the compositional verification as given by the new approach partial composition of components is introduced. This composition allows using only one of several import interfaces. This approach is investigated by considering a Petri net based sequence controller developed for a model plant in the field of manufacturing

engineering. The controller is modelled using the software Netlab as introduced in Orth et al. (2006). This tool is used for modelling, analysis and stepwise simulation of discrete event based systems applying Petri nets.

The contribution is structured in the following way. In Section 2 the benchmark process, a model plant of a manufacturing process, is presented. Section 3 introduces the concept of component based hierarchical modelling and verification with a focus on multiple and even overlapping interfaces. There the Petri net model of a subsystem of the controlled benchmark process is given as an example of the introduced notions. Beginning with the flat net, components are extracted in order to structure the net hierarchically. In addition, multiple import interfaces and partial composition are motivated using the benchmark example as a technical application. The contribution ends with section 4 which summarises contents, draws conclusions and shows future work.

2. MODEL PLANT AS BENCHMARK PROCESS

The approach of a component-based modelling and verification is applied to a technical system to evaluate the possible benefits. As a benchmark process a model plant is used for pointing out these advantages especially for complex large-scale processes.

2.1 Manufacturing model plant

The manufacturing model plant is situated at the Institute of Automatic Control (IRT) at the RWTH Aachen Technical University. The discrete event based process of the model plant describes the packing procedure of a liquid product coming from an arbitrary process plant. The model plant can be divided into three partitions as depicted in Fig. 1 which shows the overview of the plant.

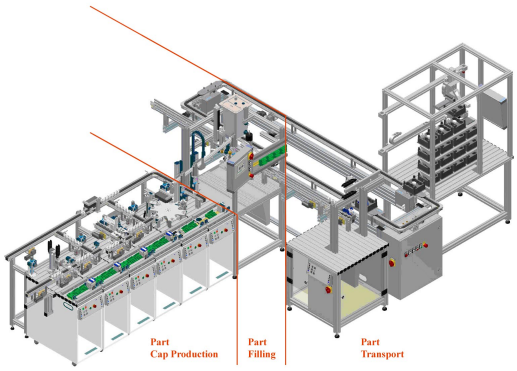

Fig. 1. Model plant for packing process

The left partition produces the closure heads for the glass bottles, in the following denoted as caps ("Cap Production"). The middle partition is responsible for the filling of the liquid product coming from a storage into provided glass bottles and the closure with caps ("Filling"). The right most partition ("Transport") groups six filled bottles into trays and transports them on a circular band conveyor to the high rack storage. The left partition "Cap Production" can be divided into five stations one of which will be explored in more detail in the following section. The stations are ordered linearly to avoid concurrency and resulting complex coding of sequence controller programs. The level of complexity is therefore reduced by hardware structuring.

2.2 Station "Compression"

A closer look to the station "Compression" is given in Fig. 2. It consists of a turntable which transports the caps to different internal positions where four different tasks are carried out.

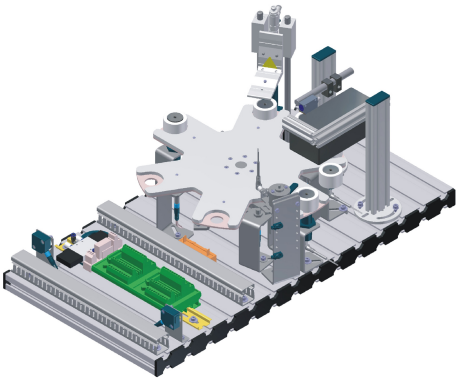

Fig. 2. Station "Compression"

The caps enter the station with a RFID chip (Radio frequency identification) placed loosely on the top. In a first step the chip is pressed into the cap by a fluidic muscle, followed by a measuring device that detects the quality of the pressing action (quality criterion). Thereafter, information is written on the data chip in order to provide this information for following process steps. Finally the cap is released by using a pneumatic actuator pushing it off the turntable. The fourth station "Compression" is now the object of a more detailed observation as it is used as an example for component derivation in Section 3.4.

3. COMPONENT-BASED HIERARCHICAL MODELLING AND VERIFICATION

3.1 Concept of the component based approach

Basically, a component consists of an import interface, an export interface and a body specification. Composition of components is achieved by a hierarchical operation that involves the import interface of the requiring component and the export of the providing component. Accordingly, a Petri net component COMP = (IMP,EXP,BOD) consists of three Petri nets: the import Petri net IMP, the export Petri net EXP and the body Petri net BOD.

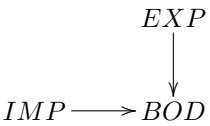

Fig. 3. Single body component with import and export

The interfaces are mapped to the body using suitable mappings of Petri nets as illustrated by the diagram in Fig. 3. In Section 3.4 these mappings are given in terms of the colourings of the places, where the import interface

(e.g. in Fig. 10) is illustrated by the red places and the transitions in between. The export interface (e.g. in Fig. 8) is adumbrated by the green places, but conforms to the corresponding import interface, in this case it corresponds to the import interface in the dashed box of Fig. 9.

Composition of components is the crucial operation for structuring a system into subsystems. Composition is achieved by mapping the import interface of the requiring component to the export interface of the providing component. Using the mapping the new component body is constructed gluing the bodies of the requiring and providing component together.

3.2 Component-Based Verification

Components are self-contained units with a well-defined syntax and semantics. In Ehrig et al. (2002) semantics of components is defined by considering each possible environment expressed by each possible transformation of the component's import. According to the transformation-based semantics the notion of import-export implications characterises the Petri nets component with respect to its environment. Based on a suitable temporal logic calculus which allows the derivation of formulas and import-export implications can be defined. In Padberg et al. (2007) the component concept is extended with import-export implications which are formulas given in temporal logic. The export statement given as part of the export interface is guaranteed independently of the component's environment, provided the import requirement is met. This approach to component verification helps to guarantee specific properties that are formalised in terms of a temporal logic. The underlying idea is that components guarantee specific export statements, provided that the import assumptions are satisfied. Hence, components are equipped with an additional import-export implication $\rho \Longrightarrow \gamma$ where ρ is a temporal logic formula concerning the component's import and γ is a temporal logic formula over the component's export. The component guarantees γ, provided that ρ holds. The satisfaction of the import-export implication by a component is formulated with respect to an arbitrary environment. The idea is graphically denoted in Fig. 4.

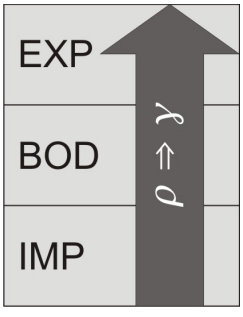

Fig. 4. Verification of a component with a single import

For the hierarchical composition of a requiring component and a providing component the export statement of the providing component has to imply the require assumptions of the requiring component's import. In case of hierarchical composition of two components, the import-export implications can be combined if the providing component meets the import requirements of the requiring component (for details see Padberg et al. (2007)). Then the result of the composition is a component that guarantees the original exports statements of the requiring component if the import assumptions of the providing component are met.

3.3 Multiple Interfaces

The new concept illustrates the use of multiple import interfaces. The need for multiple interfaces arises mainly for two reasons: First, the interfaces themselves become to large and need to be structured as well. And second, the requiring component requires different functionalities, that need to be separated. In both cases the import interface is split into several (sub-)import interfaces. These imports may have overlapping places and hence influence each other. Nevertheless, the concept of import-export implications is extended and realises means for the verification of Petri net components as introduced in Padberg et al. (2007) for the case of multiple import interfaces. Multiple import interfaces IMP_1 to IMP_n need not to be disjoint, they may overlap at some places. This set O of overlapping places is used for a construction that glues the import interfaces. This construction describes multiple imports IMP_i as shown in the diagram in Fig. 5.

For the formalization see Def. 2.4 in Padberg (2008). The formal description as given in Padberg (2008) uses different kinds of mappings and a few categorical constructions, mostly pushouts.

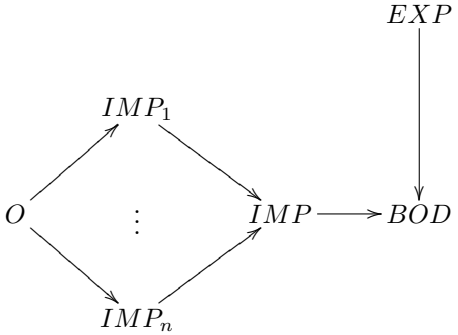

Fig. 5. Component with multiple imports

The partial composition allows the splitting of the import into several (at least two) import parts IMP_1 to IMP_n. The remaining unused import interfaces need to be connected adequately to the import of providing components (see Def. 2.5 in Padberg (2008)). The case of non-overlapping import interfaces is merely a special case of the split import. The partial composition allows connecting one of the import interfaces, e.g. import IMP_1, to an export interface of another component and results in a new component that again has a split import. This new split import consists of the import parts of the first component that have not been used (i.e. IMP_2 to IMP_n) and the import parts of the corresponding imported components which are glued adequately to those import parts that are not used (see Fact 2.6 in Padberg (2008)). According to Ehrig and Mahr (1990), the ordering of the partial compositions along different import parts is irrelevant. The main result for the successful application of this new

composition operation for components is that it allows component-based verification as well. Given several import interfaces IMP_n, an import assumption exists which consists of a conjunction of import requirements $\rho = \rho_1 \wedge \rho_2 \wedge \ldots \wedge \rho_n$ and for each import there is exactly one import requirement ρ_i. The idea is depicted in Fig. 6.

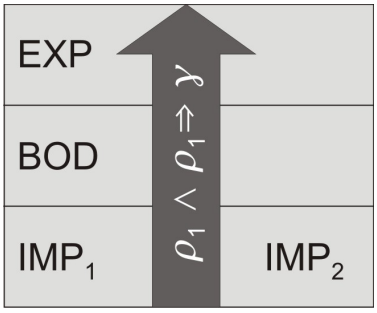

Fig. 6. Verification of a component with multiple imports

To achieve component based verification, it is shown in Fact 2.13 in Padberg (2008) that the partial composition of Petri net components again yields a component with guarantees, i.e. a new component that again satisfies its import-export implication.

A Petri net component is illustrated using colourings of the net elements, e.g. the net component in Section 3.5.

3.4 Hierarchical, Petri net based modelling of station "Compression"

The Petri nets graphically illustrated in this section show the body specification. The interfaces corresponding to the body are pointed out by using colourings and bold lines of the net elements. Thereby the export is denoted by light green nodes, the import is given by dark red nodes and overlapping interface nodes, i.e. places, are stressed explicitly by highlighting dotted circles (see Fig. 9). All other nodes of the body net are blank. The colouring is chosen to be light and dark grey in colourless print-outs, respectively.

The original Petri net model of station "Compression" is depicted in Fig. 7.

First of all, it is easy to see that the graphical representation suffers from the net size although only a small part of the plant is modelled. The controlled process is fragmented into 3 hierarchical layers (L0, L1, L2) which is denoted by dashed boxes in this figure. The bottom layer L2 consists of different nets which describe various operations at the internal positions of station "Compression". There exist four different tasks, namely pressing the loosely placed RFID chip into the cap, measuring the quality of pressing, writing the gathered quality information onto the chip and last but not least releasing the cap off the table. As an example for L2, the net describing the quality measurement is depicted in Fig. 8.

At the lowest layer there are no import interfaces and as a result no red (dark grey) nodes are given. In addition, the green (light grey) nodes indicate that this net is able to be incorporated into a higher layer in hierarchy using an appropriate abstract export representation of the body net. This representation can be found inside the dashed

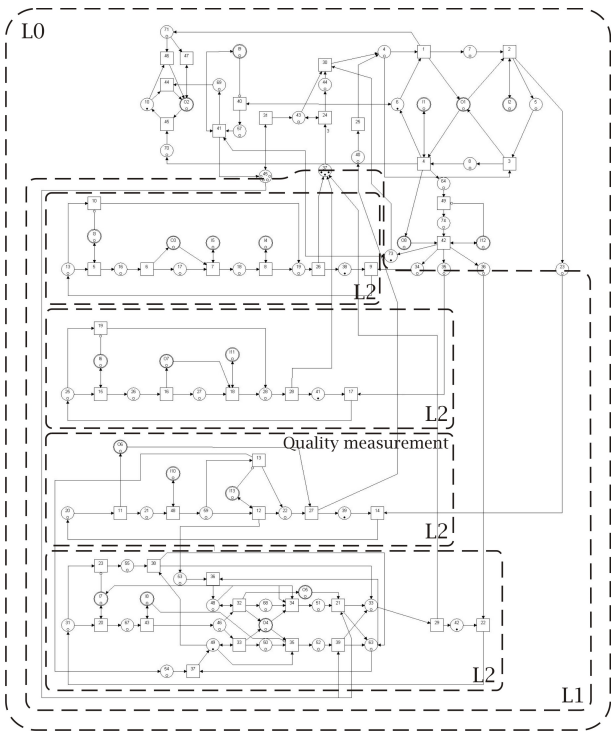

Fig. 7. Original Petri net of station "Compression" with layer classification

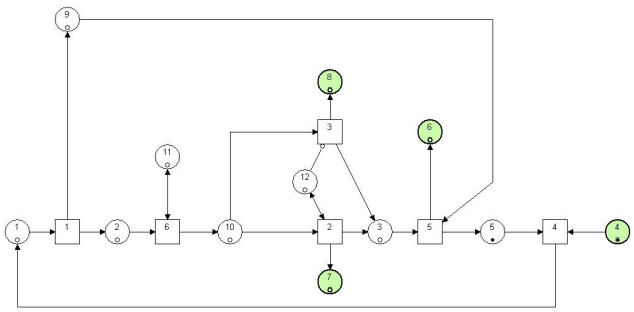

Fig. 8. Layer L2 body net of quality measurement

box in Fig. 9 which is L1 and where it is marked as import (red nodes). Remember that for a valid mapping the IMP and EXP of components need to coincide.

The net in Fig. 9 summarises all four functionalities by using valid abstract representation of the body nets in L2. It is important to notice at this point that the import interfaces (red nodes) of this component are identical to the corresponding export interfaces of the lower level components. The small dotted circles point to nodes which represent overlapping interfaces. Overlapping interfaces mean that places of different export interfaces of a lower level in hierarchy are glued together in places of the import interface. An overall import interface with an independent or parallel set of import components is possible, although it is not shown here. In addition, there are places which are part of the import and export at the same time. This is stressed with a fading in colour from green to red or light to dark grey, respectively. The information of these places is simply handed over from top layer to bottom layer. In Fig. 10 the upmost net (L0) of the hierarchical approach is displayed.

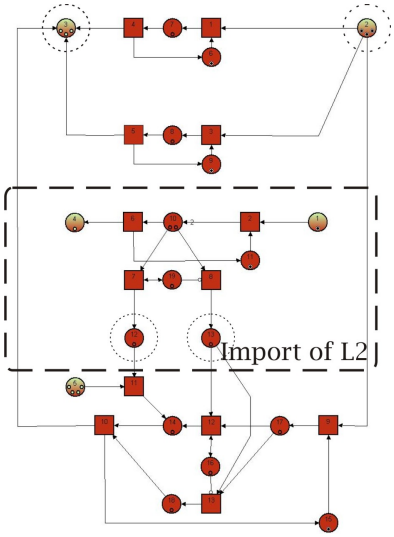

Fig. 9. Layer L1 body net of abstract actions

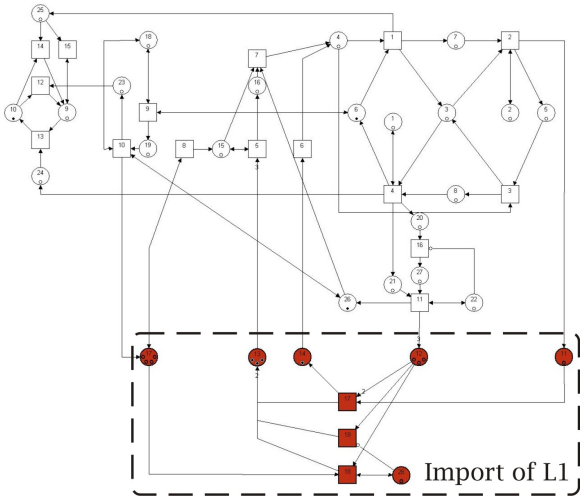

Fig. 10. Layer L0 body net of station "Compression"

Since this is the upper end of hierarchy, there are no nodes of export interfaces to display in terms of colouring. Nevertheless, the component of the level below is glued into the net using its export representation. Again the red nodes denote the import interface which must equal the export representation of the corresponding net (L1) in Fig. 9.

3.5 Component-based Verification

So far the concept of a hierarchical modelling using Petri nets is applied to a technical system. Another aspect needs to be pointed out. The concept of component based hierarchical modelling also provides import/export implications in order to verify desired properties throughout the layers even if multiple interfaces with eventually overlapping places are used.

In terms of this technical benchmark the modeller is interested in verifying certain properties of the controlled system. Using a bottom up strategy one needs to guarantee that for a given initial marking of input and body places of lower levels in layer hierarchy a desired final marking for output and body places is reached without discrete cycles in-between. If the overall process is cyclic and therefore the lower layers are probably called several times, the initial marking of all body places in all layers needs to be preserved. As a result there is exactly one desired deadlock concerning the output places of each lower level in layer hierarchy which is to be guaranteed while a pseudo-reversibility of the corresponding body net is desired. For the upmost layer reversibility and deadlock freeness need to be guaranteed for the example process in this contribution. All these properties can be easily expressed in some temporal logic calculus and proven in general fashion as done in computer science. The desired quasi-reversibility is more or less the same as soundness in workflow nets, see van der Aalst (1997). By applying the bottom up strategy, the initial marking of the import nets become a requirement to the body nets in terms of a restriction to possibly allowed markings on the corresponding places. This requirement might be preserved by the modeller himself designing the net by hand and additionally apply some automatism implementing supervisory control to rearrange the design in way to meet the required marking specification, see Moody and Antsaklis (1998).

In other words, the concept of import/export implications allow to guarantee the export property formulated as γ, as long as the body and the imports fulfil the required property ρ. The prove of properties for the body nets can be done stepwise and in all layers separately. Thereafter, the overall system property can be guaranteed without proving the property of the reassembled complete net.

4. CONCLUSIONS AND OUTLOOK

In this contribution the concept of component based modelling and verification was introduced. A component consists of three nets, namely the body BOD, the import IMP and the export EXP. Using the import and export interfaces a component can be constructed hierarchically by gluing the corresponding interfaces. Desired properties of the flat net can be guaranteed for components that satisfy specific import/export implications. An essential result is that the concept still holds if multiple imports are used, even for overlapping places. A part of the benchmark process plant was presented and modelled by one Petri net for the Station "Compression". Using this net, a hierarchical structure with 3 layers was constructed. A bottom up strategy was used for developing the body nets from the original net. In a next step the export representation of these nets was derived. Only the body nets were depicted completely while the import and export interface nodes were denoted using colours, red and green respectively. It is easy to see that the hierarchical structure divides the complex flat net into smaller more manageable nets whose properties are verified more efficiently. Especially, the use of import/export implications in the sense of verification reduces the system to subsets which can be analysed separately and then lead to valid properties of the complete system. This results in a more efficient verification procedure due to smaller reachability space in subsystems.

This example therefore shows that the modeler can benefit from such a strategy for modelling and verifying large scale systems with Petri nets while using methods of computer science in the field of process engineering. Nevertheless,

there are some obstacles to overcome in order to apply this method in a powerful manner to the field of process engineering. As described in this contribution, an existing net is divided in hierarchical layers in order to achieve the named benefits. The idea is to motivate a new approach for modelling and verification applying a top down or bottom up strategy. Both strategies are worth an investigation, whereas the engineers may prefer the bottom up idea using knowledge from the real process plant and its possibilities of controls and in contrast, computer scientists may approach from the top using their knowledge in terms of refining the abstract representation as long as needed to handle the process. In the case of bottom up modelling, the modeller could be supported by an algorithmic procedure for the design of the abstract export representation of the body net since this reduction of the system is a crucial step in the design process. Such an algorithmic procedure would also be useful for verifying nets designed in a top down approach. Another scope for research is the fact that Petri nets used for control are so called interpreted Petri nets. The matter of input-/output-coupling or controlability/observability, respectively, needs further investigation. For the example presented in the contribution there is no impact on the validity of the methods because of the absence of concurrency in the model. Again, this is a result of hardware structuring. Furthermore, all input places are only for synchronisation to the process or are meant to be decision variables for alternative paths in the Petri net. Petri nets deal with non-determinism (i.e. alternatives) in a well defined fashion. Hence, it is possible to exclude the influence of controlabillity for this kind of processes. For technically applying this concept it still has to be implemented.

REFERENCES

D. Abel and A. Bollig, editors. *Rapid Control Prototyping.* Springer, Berlin, 2006.

H. Ehrig and B. Mahr. *Fundamentals of Algebraic Specification 2: Module Specifications and Constraints*, volume 21 of *EATCS Monographs on Theoretical Computer Science.* Springer, Berlin, 1990.

H. Ehrig, F. Orejas, B. Braatz, M. Klein, and M. Piirainen. A Generic Component Concept for System Modeling. In *Proc. FASE 2002: Formal Aspects of Software Engineering*, volume 2306 of *Lecture Notes in Computer Science*, pages 32–48. Springer, 2002.

C. Girault and R. Valk, editors. *Systems Engineering: a Guide to Modelling, Verification and Applications.* Springer, 2003.

M. Große-Rhode, R.-D. Kutsche, and F. Bübl. Concepts for the evolution of component-based software systems. Technical Report 2000/11, TU Berlin, 2000.

E. Kindler, V. Rubin, and R. Wagner. Component tools: Integrating Petri nets with other formal methods. volume 4024 of *Lecture Notes in Computer Science*, pages 37–56. Springer, 2006.

J.O. Moody and P.J. Antsaklis, editors. *Supervisory Control of Discrete Event Systems Using Petri Nets.* Kluwer Academic Publishers, Norwell, 1998.

Ph. Orth, U. Kuessel, and D. Abel. Netlab und Netlab Toolbox für Matlab/Simulink - Ein Werkzeug zum Rapid Control Prototyping von Steuerungen mittels Petrinetzen. In *Tagungsband Entwurf komplexer Automatisierungssysteme EKA 2006*, pages 343–353. Institut für Regelungs- und Automatisierungstechnik TU Braunschweig, 2006.

J. Padberg. Regelbasierte Verfeinerung von Petri-Netz Modulen. In *Tagungsband Entwurf komplexer Automatisierungssysteme EKA 2006*, pages 57–67. Institut für Regelungs- und Automatisierungstechnik TU Braunschweig, 2006.

J. Padberg. Partial composition of components: Formal foundation and component verification. Technical Report 2008-06, Technische Universität Berlin,Fakultät IV, 2008. URL `http://iv.tu-berlin.de/TechnBerichte/tb2008.html`.

J. Padberg and U. Kuessel. A component-based verification approach based on Petri net components. In *Proc. FORMS/FORMAT 2007 - "Formal Methods for Automation and Safety in Railway and Automotive Systems"*, pages 40–50. GZBV, 2007.

J. Padberg, H. Ehrig, and F. Orejas. Towards component verification in the generic component framework. In J. Kuester-Filipe, I. Poernorno, and R. Reussner, editors, *Proc. Formal Foundations of Embedded Software and Component-Based Software Architectures (FESCA 07), Satellite Event of the European Joint Conferences on Theory and Practice of Software (ETAPS)*, Electronic Notes in Theoretical Computer Science, Amsterdam, 2007. Elsevier Science. to appear.

W.M.P. van der Aalst. Verification of workflow nets. volume 1248 of *Lecture Notes in Computer Science*, pages 407 – 426. Springer, 1997.

H. Weber. Continuous engineering of information and communication infrastructures. volume 1577 of *Lecture Notes in Computer Science*, pages 22–29. Springer, 1999.