High-Level Petri Nets Control Modules for Service-Oriented Devices: A Case Study

J. Marco Mendes
Faculty of Engineering - University of Porto, Rua Dr. Roberto Frias s/n, 4200-465 Porto, Portugal
marco.mendes@fe.up.pt

Paulo Leitão
Polytechnic Institute of Bragança, Quinta Sta Apolónia, Apartado 134, 5301-857 Bragança, Portugal
pleitao@ipb.pt

Armando W. Colombo
Schneider Electric GmbH, Steinheimer Str. 117, D-63500 Seligenstadt, Germany
armando.colombo@de.schneider-electric.com

Francisco Restivo
Faculty of Engineering - University of Porto, Rua Dr. Roberto Frias s/n, 4200-465 Porto, Portugal
fjr@fe.up.pt

***Abstract*- This paper describes a solution for the control of service-oriented devices based on modular and special adapted High-Level Petri Nets process description of intra- and inter-control activities. The procedure is applied on a case study scenario, corresponding to a real transfer system made of several control devices represented as service-oriented components able to share information between them. The High-Level Petri Nets are adapted to associable models applicable to describe control processes and sufficient elastic for different control strategies. Valuable and flexible control features are obtained from its application, such as an integrated methodology for the modular control with decision support and validation.**

I. INTRODUCTION

Decentralized, autonomous and collaborative automation systems are becoming an emergent paradigm towards flexibility and automatic re-configurability. The reconfiguration of those systems and the emergence of decentralized control require the existence of distributed and modular control components that interact in order to accomplish distributed control activities.

The main tendency for the engineered control architecture used in this work is the service orientation. Service-oriented Architectures (SoA) is the abstract concept of a software architecture, which in the center stays the offer, search and use of services over the network [1]. In a service-oriented environment, distributed resources provide their functionalities in form of services that can be accessed externally by clients without knowing the underlining implementation [2]. Thus, the challenge of SoA is to reconcile the opposing principles of autonomy and interoperability [3]. Requesters of services only need to know the external visible interface (description of a service) and rules on how to access them; the internal structure and functionality represented by a service is hidden. Standard protocols should handle these issues and thus specify a set of interaction and technology rules that should be followed by all involved partners to successfully permit the conversation.

From the basic specification to the more complex engineering, coordination and aggregation methods are required in service-oriented control architectures of automation and production systems. The resulting executable processes should be able to integrate as part of the component's control and also applicable to their interaction. There are several possible specifications to be used for this purpose, but based on the authors experience and from a brief revision of the last published works, it is clear that High-Level Petri Nets (HLPN) are a strong candidate suitable for modeling, analysis, validation, synthesis and control supporting the design of service-oriented automation and production systems.

The presented solution for the control of service-oriented devices is based on modular and special adapted HLPN process description of intra- and inter-control activities. The modular, collaborative and event-based nature of these systems makes possible to form complex control architectures by tying together individual units. A case study scenario is used to illustrate the application of this control strategy and to raise its valuable features. Following the introduction, Section 2 introduces the component based control architecture and the HLPN solution. Section 3 describes the case study scenario and Section 4 presents the application of the HLPN-based control approach for service-oriented systems. Section 5 discusses some important features from the application. Afterwards, the paper ceases with the final conclusions.

II. HIGH-LEVEL PETRI NETS CONTROL ARCHITECTURE FOR SERVICE-ORIENTED SYSTEMS

This section resumes the architectural concepts and the control approach using HLPN, for service-oriented systems, which will be later applied to the case study scenario.

A. Basic Architectural Concepts and Control Components

The proposed architectural patterns for reconfigurable production/automation systems are enforced by the service-orientation, permitting the communication between the system's components. There are four different types of components involved in the control (Fig. 1) [4]: Mechatronic Components, Smart Mechatronic Components, Process Control Components and Intelligence Support Components. Others may also be integrated for diversified jobs.

The most common are the Mechatronic Components (MeC) that in their most basic form only provide atomic services, corresponding to the setting and reading of inputs and outputs for driving the mechanic and electronic parts. It can also be applied to machinery that is not open to be programmable or have unsupported control schemes, for example, providing

978-1-4244-1766-7/08/$25.00 ©2008 IEEE

aggregated services for the execution of a movement program executed by an industrial robot.

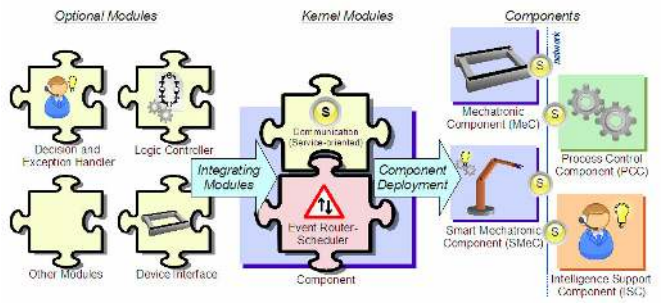

Fig. 1. Types of control components and modules to build them.

Smart Mechatronic Components (SMeC) are more complex with build-in logic control and can be supported by intelligent mechanisms, such as decision and exception handling. At the end, the exposable capabilities by these components must be distinguished and provided in form of services that can be used by other components (in general service requesters) to build the desire automation system. With the SMeC on place it is possible to build distributed systems due to the collaborative nature of them. But, from the other hand, it may be necessary to introduce centralization or hierarchy due to certain circumstances or implicit boundaries by the equipment.

Two other components are provided to take over some of the tasks that can be achieved by the social interaction of SMeC. A Process Control Component (PCC) is able to coordinate, in a centralized manner, a process model that describes the sequencing of several services and its operation to complete a specific goal. Finally, a dedicated Intelligence Support Component (ISC) incorporates real-time decision capabilities and exception handling to support the PCC. The distinctive attribution of process control based on predicted or desired behavior and the handling over exceptional events can also be merged in one entity, but they are here treated separately.

Each component of the architecture may be implemented independently and differently. The only requirement is that it should share its functions as services and obey to the protocols of communication and processes. In [4] a proposal is made for a modular and functional-guided design method for components, such as the SMeC. A component is structured by an agglomeration of different puzzle objects (modules), each one having its own role (see Fig. 1). Communication to other components is done via a specific module that provides service-oriented communication (so the component may interact with others). The central module is the Event Router-Scheduler, similar to the nervous system of animals and providing, among others, a way for passing impulses (events) between modules. Other modules are up to the requirement of each component: for example, a MeC would have a device interface module to control the inputs and outputs of the machine and a SMeC has additionally a logic control module and/or a decision-making module. This flexibility may contribute to the development for customized components for diversified tasks.

B. High-Level Petri Nets Control Approach

A kind of High-level Petri Nets (HLPN) adapted from [5-6] is used in this work as the kernel for modeling, analysis and execution of process control in service-oriented systems. These HLPN take advantage of their powerful mathematical foundation to represent and validate certain typical relationships, such as concurrency and parallelism, synchronization, resource sharing and mutual exclusion [5, 7], extending with features that covers some limitations of basic Petri nets formalism. Namely, they use the concepts of stochastic Petri nets, step-wise refinement by transition explosion and conflict detection and resolution (see Fig. 2). Beside these, to map transitions to "the real world", they are associated to input events and output actions, allowing to be connected to service operations, device interfaces and other models. Transitions may only be activated by enabling the associated rule and activating the input event. Transition firing corresponds to the firing rule and setting the output actions. A final appointment is about the modularization of HLPN, in the sense of the HLPN models may be connected via specific ports, providing interoperability control between modules.

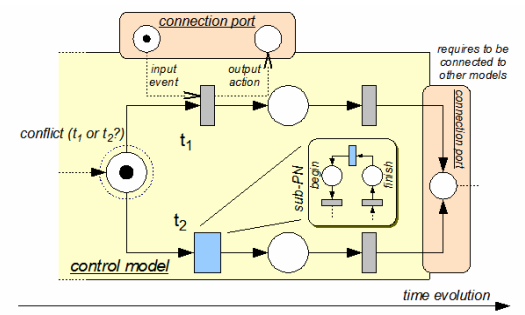

Fig. 2. Visual exhibition of the used extensions of HLPN (adapted from [8]).

The designed HLPN models are interpreted by the logic controller running inside the modular structure of SMeC and PCC. The service's behavior is basically a partially ordered set of operations, such as the complex processes between services. Therefore, it is straight-forward to map it into a HLPN, where operations (actions and events) are modeled by transitions and the places mean the state of service's coordination or the internal state of control. More information can be obtained by reading the document referenced by [8].

III. DESCRIPTION OF THE CASE STUDY SCENARIO

The case study scenario used to illustrate the application of HLPN based control for SoA is based on instances of the Unidirectional Transfer Unit and Cross Transfer Unit, modeled in similarity to the units from the FlexLink® Dynamic Assembly System (DAS) 30, depictured in Fig. 3.

The DAS 30 transfer system combines flow-oriented production control and modular automation with ergonomic manual assembly solutions, providing flexibility and

versatility. Only a part of the system is used, mainly the top assembly transport system, missing the bottom transfer units and the two lifters for connecting the top and the bottom parts.

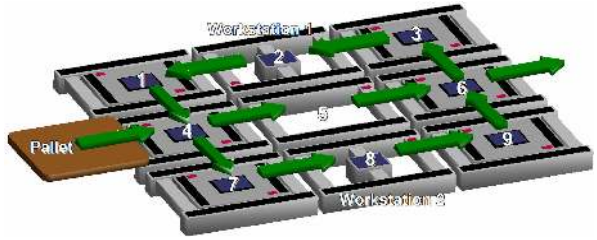

Fig. 3. Layout of the DAS 30 transfer system.

The transfer system is made of nine transfer units (conveyors) of the Unidirectional and Cross types, represented in Fig. 4. The Unidirectional Transfer Unit provides an input and an output port and the Cross Transfer Unit provides transfers not only in the longitudinal but also in transversal axis. Moreover, the Cross Transfer Unit may be seen as a composition of two devices, namely a Unidirectional Transfer Unit and a lifter with directional transfer capabilities.

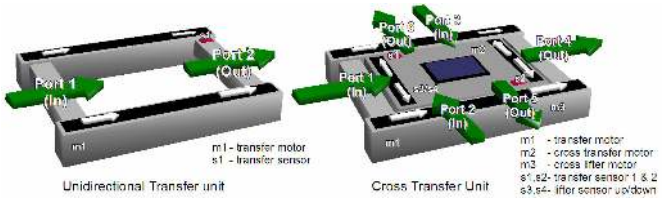

Fig. 4. Types of transfer units used in the demonstrator.

The pallets enter in the system through the transfer unit 4 and are conveyed using alternative paths to achieve the two workstations associated to the transfer units 2 and 8. These transfer units have the possibility to halt the pallet during the required amount of time for the execution of the operation. At the last the pallets are routed outside through the transfer unit 6. Each transfer unit has a RFID (Radio-Frequency IDentification) reader/writer for identifying the pallets and transmitting information to them. Table I summarizes the characteristics of each transfer unit belonging to the transfer system.

TABLE I
CHARACTERISTICS OF THE TRANSFER UNITS

Unit Id	*Type*	*RFID*	*Work Station*	*Multiple I/O*
1,3,7,9	cross	✔	✘	✘
2,8	unidirectional	✔	✔	✘
4,6	cross	✔	✘	✔
5	unidirectional	✘	✘	✘

Up to this point nothing has been said about the control of the scenario. It is up for the next section to explain the applied modular control approach based on High-Level Petri Nets.

IV. DEVELOPING AND INTEGRATING CONTROL STRUCTURES FOR THE CASE STUDY SCENARIO

The control system for the case study scenario is based on single HLPN models that can be set together to balance the overall operation of the transfer system. The proposed solution uses the logic controller of (S)MeC and PCC as an integrated interpreter that handles the special type of HLPN previously discussed. Simpler devices may embed already pre-compiled HLPN instead of having an interpreter, since the logic and communication itself doesn't require to be changed at runtime.

A. Identification of Components and Global Behavior Description

Before entering into details of the process control for the case study scenario, the identification of the components that model the system's behavior is crucial. Fig. 5 represents the case study scenario transfer units mapped into control components of the architecture, resulting in nine MeC or SMeC (with embedded control and/or intelligence support), depending on options for control.

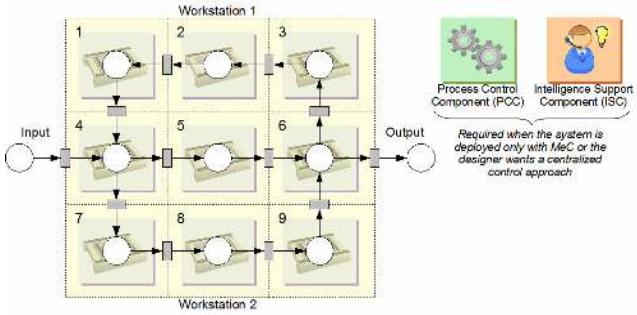

Fig. 5. Identification of control components and global behavior description.

Using MeC components, additional PCC and ISC components must be used as controlling masters. The overlaid HLPN model describes the global expected behavior. The transitions between the places reflect the need for synchronization between two components. They can be detailed to overview the associated operations that realize a transfer movement from one unit to another.

Besides the global control view, it can be easily seen that actually the behavior represented in Fig. 5 corresponds also to the paths that pallets can take. This is due to the close association of the process control to the functionality of the system, i.e. routing pallets to the workstations or working as a pass-through to other systems.

B. Control Models for the Unidirectional and Cross Transfer Units

The expected behavior of the Unidirectional Transfer Unit and the Cross Transfer Unit will be used to build the HLPN control models.

The Unidirectional Transfer Unit provides two ports (*In* and *Out*) to be connected to other devices (such as similar transfer units) and a port to set and read the inputs/outputs of the device. The logic that controls the three ports is done by a HLPN model represented in Fig. 6.

The expected behavior is basically related to set the motor according to the external requests (e.g. start transfer service) and the status of the sensor (which indicates that a pallet is available after a transfer in operation). The two transfer ports can also be used to synchronize the transfer in and out of pallets. The HLPN control model for units 2 and 8 considers a

special transition that can be used to represent the time execution during the workstation's operation.

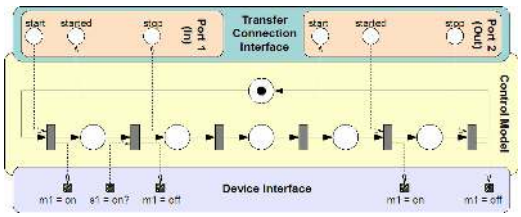

Fig. 6. HLPN control model for the unidirectional transfer unit.

The more complex Cross Transfer Unit allows pallets to be transferred not only in the transversal axis, having six different transfer ports (of type *In* and *Out*) and one device port. With the lifter unit down and using the motor *m1* it is possible to transfer from port 1 to port 4. When the lifter is up, the transfer from port 2 to port 6 is done using the motor *m2* and the transfer from port 3 to port 5 is done by setting the motor *m2* with reverse polarity. The movement of the lifter is done via the motor *m3*, using two sensors (s3 and s4) to indicate if the lifter is up or down.

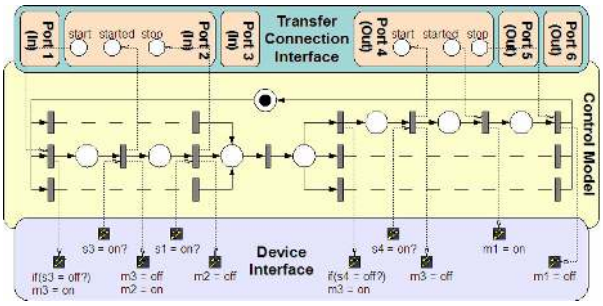

Fig. 7. HLPN control model for the cross transfer unit.

Transfer Units 1, 3, 4, 6, 7 and 9 are deployed using the control logic represented in Fig. 7. Most of them (1, 3, 7 and 9) require only one path of the several available, thus the others are deactivated. In the other side, units 4 and 6 are made of several routing possibilities and the control model has now different options to receive and route a pallet (for simplification, only the logic of port 2 and port 4 is present in Fig. 7, the others are done in a similar way). Decision is here required to choose one among different options that are described in the control model. For instance, a special decision support module (or in alternative an external ISC) may provide necessary information, e.g. based on the identifier of the pallet (given by the RFID device in the middle of the unit).

C. *Putting Control Models Together*

The case study scenario can be modeled by the combination of the nine control models that are the representation of the two types of transfer units. The task is now to define the whole emerged logic and resulting behavior based on the individual models. Since they are specially tailored for this requirement, the models can be easily used as building blocks, only by correctly connecting the external visible ports together.

The synthesized control model of the scenario is represented in Fig. 8. It is made of the connection of the individual models from the transfer units that provide the individual control logic, following the connection rules described in [8]. Still open are two ports that were not connected, i.e. the left transfer in port of unit 4 and the right transfer out port of unit 6. Since they behave as the input and output to this system, they can be used to connect outwardly and thus provide the doors for an externally transparent and more autonomous aggregated component. By using these two ports, it is generally not required to know how it works inside (acting as a black box).

This transport system has different routing options for pallets that can populate it. The main options are for transporting pallets to the desired workstations and passing through the system to another one connected (when the pallet does not require any of the workstation's services). This involves the presence of conflicts in the control, primarily by supplying different paths for the pallets. The monitor states, presented in each individual model, are responsible for regulating the shared resources, forbidding any access by a pallet when the unit already has one. Other remark is the deactivation of some control branches of the cross transfer units in the scenario, namely the ones in the corners (1, 3, 7 and 9). The synthesized model does not define them and only exhibit the used ones.

D. *Service Deployment and Control Strategies*

The desired production process is achieved by putting these (S)MeC and other components working together, specially tying together the developed control models. The ports used in Fig. 8 to connect the control models together can be mapped to

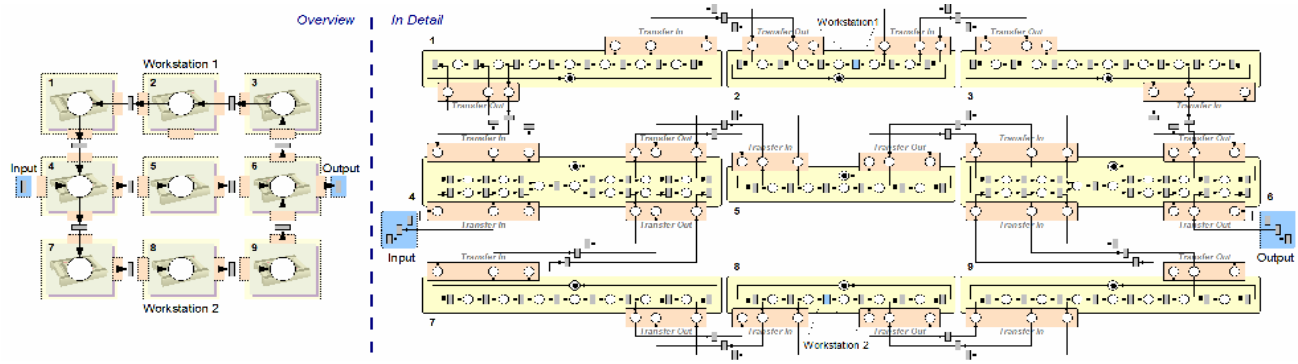

Fig. 8. Synthesis of the whole control for the scenario based on the models

service's ports, and consequently, the logic between two ports describes the interaction of two mutual offered services. This can be easily seen in Fig. 6 and Fig. 7 where the transfer connection interface is independently of the technology and therefore may correspond to a transfer service interface. The ports are specific gates to synchronize a service and must be known by the requester for successfully interaction. In practice they are also strictly related to the physical connectivity of the transfer units.

A connection between control models is done by simply matching the ports logic and binding them together. When these ports are deployed as their services counterparts, additional care may be required, such as discovering the right service, proposing its usage and synchronizing the message flow during the system operation. For the conversation, semantically-rich descriptions may introduce flexibility that allows using machine reasoning to perform automatic matchmaking of required and offered services using logical inference, rather than performing hard-coded one-to-one mappings [9].

Since these components comprehend the concepts of service-oriented systems, one technological solution is the use of Web Services (WS) to make them interoperable. WS have become simple, economical, widely available interaction means between information systems. In other hand, electronic devices are increasingly connected to standard networks (Ethernet and TCP/IP are widely available in many places) [3]. Device Profile for Web Services (DPWS) defines the extensions required for using Web Services in electronic devices taking in account their specific constraints [10].

Being able to define the services in the components, thoughts have to be carried about how and where the control models should run. As said before, when the transfer ports are deployed as services and the models can run inside the corresponding unit (represented as a SMeC in the architecture), parallel peer-to-peer communication may be sufficient to achieve the objectives of the system's behavior. In case of using a centralized control, i.e. a PCC component, the whole synthesized control based on the singular models runs externally and does not access directly the device's I/O. In this case the distribution can be handled by including an intermediate service (proxy), so that the client (where the control is in) can easily access the atomic operations provided by the server (a MeC). In both cases, the same control models are used but differently deployed.

E. Decision Support and Exception Handling

The coordination of processes and services, depending on the flexibility that the system reveals, requires the decision-making and conflict resolution at runtime, because a system model does not describe a fixed sequence of actions, but rather all possible combinations thereof. On the other hand, it may also be necessary to choose from different available services that result from a filtered discovery.

Flexible parts can be identified in the behavioral HLPN-based models of the production system. Fig. 9 illustrates a conflict in the HLPN model representing a pallet centered on the transfer unit 4 that must be routed out, either to unit 7 or to unit 5. In this system the control models run all together in the PCC component that access and respond to the available MeC (representing each transfer unit).

When the conflict is detected (i.e. to choose between t1 and t2) a decision-making is necessary. Since the structure of the model and the information associated to their components are not sufficient for solving such conflicts, a decision support is requested to the ISC component. Having the problem description and additional information (such as the id of the pallet and the production process), it can now take a decision over the incoming problem. The choice is then transmitted to the PCC that activates the corresponding transition (for example t2), commanding the MeC of unit 4 and consequent conveying the pallet to the unit 5 (if it is not occupied).

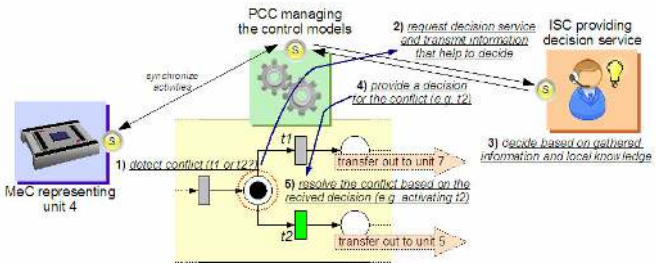

Fig. 9. Decision support for routing a pallet

The degree of complexity associated to the decision-making instance can range from simple algorithms to complex cognitive systems, such as multi-agent systems, neural networks and genetic algorithms. An approach is to have a society of intelligent components that will provide services to help the decision-making and/or the conflict resolution during the coordination process. Their tasks can be extended into the reconfiguration of the domain in any case where the models do not exhibit the actual possibilities.

V. Analysis and Validation of Control Models

The automation control components are described through modeling their individual behavior using the HLPN formalism. The analysis of HLPN control models, both quantitative and qualitative, helps to validate the specifications of the system's behavior, verifying the correctness of the models and verifying if the models fulfill the specifications of the control system.

Based on the theoretical foundations of High-level Petri Nets, namely the functional analysis theory and linear algebra, HLPN control models can be analyzed and validated to give a complete and correct description of the behavior of a component. The edition and analysis of the developed HLPN control models for the case study have been done using the Petri nets Development Kit (PnDK) software tool [2].

The qualitative analysis, based on the structural analysis of the matrix representation of the graph model [7], allows the verification of the structural and behavioral properties of the HLPN model, extracting conclusions about the operation of the system, such as the existence of deadlocks, the bounded

capacity of resources, and the existence of structural and behavioral conflicts in the system [5]. Fig. 10 illustrates the structural analysis of the synthesized HLPN control model, provided by the PnDK software tool using linear algebra methods.

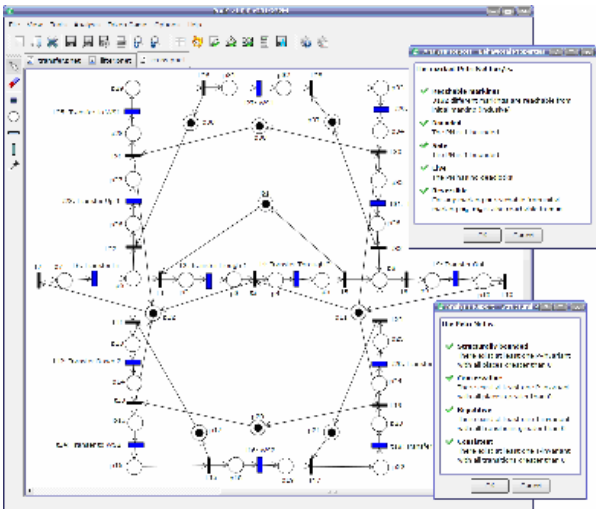

Fig. 10. Structural analysis of the synthesized control model with PnDK

This structural analysis shows that the HLPN control model is live (satisfying the necessary condition), bounded, repetitive and conservative.

Additional analysis can be performed by obtaining the T- and P-invariants. The analysis of P-invariants constitution allows confirming mutual exclusion relationships among places and functions and resources involved in the model structure. The analysis of the T-invariants allows the identification of work cycles. The qualitative analysis of the incidence matrix and T- and P-invariants allows validating, e.g., the following specifications:

- Mutual exclusion is presented in each control model of the transfer units, in the sense of only one pallet may occupy it.
- The T-invariants of the synthesize model for the transfer system describe possible sets of operations. Translated into the topology it may refer to all possible routing sequences of pallets along this system.
- It is possible to have deadlock situations due to the presence of circular paths if all transfer units of these paths are occupied by pallets. The overlapping conflicts of these paths support the resolution of these issues by activating alternative ways to route some pallets.

In a similar way, a quantitative analysis can be performed by means of the simulation of the temporized HLPN models. The information extracted from the timed evolution of the HLPN control model reflects the temporal sequence of the system operation. Moreover, cyclic evolution, existence of bottlenecks, mutual exclusion activities, etc., can be easily discovered and optimization strategies can be then proposed and verified online.

VI. Conclusion

This paper applies the High-Level Petri nets approach for the modular control of service-oriented automation and production systems into a concrete transport system scenario, made of different control components. Different HLPN control models were specified for the two main types of transport units and afterwards synthesized together to form the complete control system.

The methodology is flexible enough to permit different control strategies based on the same models and adapted to the proposed service-oriented control architecture. Also the re-configuration of the automation system is simplified due to the modularity and adaptability provided. Special attention was devoted to the features of HLPN and its analysis, contributing to achieve the demanding for modularity, flexibility, re-configurability and validity.

Future work is related to the improvement of the engineering methodology and application to the control of the whole physical scenario. Other topics are the supporting features such as decision support systems and the integration into the IT-enterprise architecture based on service-orientation.

Acknowledgment

The authors would like to thank the partners of the Innovative Production Machines and Systems (I*PROMS) Network of Excellence (http://www.iproms.org) and the SOCRADES project (http://www.socrades.eu), for their support.

References

[1] I. Melzer et al., “Service-orientierte Architecturen mit Web Services”, 2. Aufl., Elsevier, Spektrum Akademischer Verlag.

[2] J. M. Mendes, P. Leitão, A. W. Colombo and F. Restivo, “Petri net-based Approach for Web Service Automation Resource Coordination”, Proceedings of the 8th International Conference and Exhibition on Laser Metrology, Machine Tool, CMM & Robotic Performance, 2007.

[3] F. Jammes, H. Smit, J. Lastra and I. Delamer, “Orchestration of service-oriented manufacturing processes”, Proceedings of the 10th IEEE Conference on Emerging Technologies and Factory Automation, vol. 1, pp. 819-22, Sept 2005.

[4] J. M. Mendes, P. Leitão, A. W. Colombo, F. Restivo, “Service-oriented control architecture for reconfigurable production systems”, submitted to the 6th IEEE International Conference on Industrial Informatics, 2008.

[5] T. Murata, "Petri nets: Properties, analysis and applications", Proceedings of the IEEE, vol. 77, pp. 541-580, April 1989.

[6] A. W. Colombo, "Development and implementation of hierarchical control structures of flexible production systems using high-level Petri nets". Fertigungstechnik, Erlangen, Nuernberg, 1998.

[7] K, Feldmann, C. Schnur and A. Colombo, “Modularised, distributed real-time control of flexible production cells, using Petri nets”, Int. Journal of IFAC Control Engineering Practice, pp. 1067-1078, August 1996.

[8] J. M. Mendes, P. Leitão, A. W. Colombo, F. Restivo, “Service-oriented process control using modular High-Level Petri Net”, submitted to the 6th IEEE International Conference on Industrial Informatics, 2008.

[9] I. Delamer and J. M. Lastra, “Ontology Modeling of Assembly Processes and Systems using Semantic Web Services”, Proceedings of the IEEE International Conference on Industrial Informatics, pp. 611-617, Aug. 2006.

[10] F. Jammes, and H. Smit, “Service-oriented paradigms in industrial automation”, IEEE Transactions on Industrial Informatics, vol. 1, n. 1, pp. 62-70, Feb. 2005.