ENGINEERING TOOLS FOR THE INTEGRATION OF SERVICE-ORIENTED PRODUCTION SYSTEMS

Paulo Leitão*, J. Marco Mendes**, Axel Bepperling***, Daniel Cachapa*** and Armando W. Colombo***

* Polytechnic Institute of Bragança, Quinta Sta Apolónia, Apartado 1134, 5301-857 Bragança, Portugal, (pleitao@ipb.pt)
** Faculty of Engineering of University of Porto, Rua Dr. Roberto Frias, 4200-465 Porto, Portugal, (marco.mendes@fe.up.pt)
*** Schneider Electric Automation GmbH, Steinheimer Str. 117, D-63500 Seligenstadt, Germany, {daniel.cachapa, axel.bepperling, armando.colombo}@de.schneider-electric.com

Abstract: Engineering frameworks are currently required to support the easy, low-cost, modular and integrated development of production systems, addressing the emergent requirements of re-configurability, responsiveness and robustness. This paper discusses the integration of High-level Petri net-based service-oriented frameworks with 2D/3D engineering tools, allowing the digitally design, configuration, validation, simulation, control and monitoring of production systems, in an integrated manner. An experimental case study was implemented, based on the Petri nets Development toolKit (PnDK) development framework, to validate the proposed concepts.

1. INTRODUCTION

The global, dynamic and customized markets require the use of new paradigms and technologies to enable the development of low cost, modular, flexible and re-configurable production systems. Amongst other approaches, such as multi-agent systems, the use of Service-oriented Architectures (SoA) seems a suitable approach to face the referred requirements. In fact, SoA, well known in business and electronic commerce, uses the service-orientation principle that introduces important features like modularity, encapsulation (abstraction) and interoperability. The project SIRENA [Jammes and Smit, 05] was the first step in the introduction of this paradigm in industrial automation. Several open issues remain unanswered, namely the achievement of an engineering framework to support the easy, modular and integrated development of service-oriented applications.

The introduction of High-level Petri nets (HLPN) to power the process control in SoA production systems, allows the development of modular and re-configurable systems, and also permits the analysis, validation and simulation during the design phase, reducing the efforts and costs in the development of these systems. The integration of 2D/3D environment tools with HLPN-based frameworks allows the advanced engineering of modular flexible production systems. In other words, it allows virtually designing, validating and simulating production systems.

This paper describes the integration of a HLPN-based service-oriented framework with 2D/3D engineering tools, allowing an easy design and configuration of modular systems as well the analysis, validation and simulation of the designed virtual production system before to go to the operation of the production system. One important feature is the reduction of the effort and cost associated to the migration between the virtual/simulation environment to the real scenario, because the same control models are used and only the device services need to be changed.

The rest of the paper is organized as follows: first, Section 2 introduces the HLPN-based service-oriented approach for production systems and Section 3 describes the advanced topics and features of using 2D/3D modeling and simulation applications in the specified production systems. An experimental case scenario was used and explained in Section 4 to test the proposed approach. Finally, section 5 rounds up the paper with conclusions.

2. HIGH-LEVEL PETRI NETS-BASED SERVICE-ORIENTED PRODUCTION SYSTEMS

Aiming to face the challenge of developing re-configurable systems, a suitable solution is to design a control approach for service-oriented systems that are based on modular process description of intra- and inter-control activities, and able to support the modeling, analysis, validation, simulation and execution as an integrated methodology.

2.1 Service-oriented Automation Process Control

In service-oriented systems, the functionalities of automation devices can be abstracted and encapsulated as services. In these systems, the process control is achieved by the coordination of the services provided by these distributed devices, according to behavioral models.

Mendes et al. (2008) proposed a service-oriented control architecture built upon a set of distributed components, each one providing a set of services that encapsulates its internal functionalities. Fig. 1 shows the main elements of the architecture and their main functions. Several important classes of components are identified:
Mechatronic Components (MeC) comprise the physical device and its local control. For a more degree of autonomy, Smart Mechatronic Components (SMeC) have build-in decision capabilities and advanced control duties.

Process Control Components (PCC) coordinate the services offered by MeC and other components based on a process description model. These models can be defined in several languages, such as High-level Petri Nets (HLPN) [ISO/IEC, 08] and Web Services Business Process Execution Language (WS-BPEL) [OASIS, 07]. Processes can represent logical synchronization of services and forms of input/output data for device coordination, transport definitions, etc.

Intelligence Support Components (ISC) provide decision capabilities when the local solutions are not enough, e.g. for exceptions, conflict resolution and error recovery.

Manufacturing Control Components (MCC) are responsible for the production tasks and to control the production plan.

These components can be easily combined and aggregated to form more complex devices, as Russian dolls, reducing the “visible” complexity of the combined service (see the aggregated component in Fig. 1). To support the development and maintenance of the software-part of these systems, several additional engineering tools can be used (more on them later). Business and other higher levels, e.g. Enterprise Resource Planning (ERP) systems, can use the available services, such as monitoring and general production orders to integrate the automation system.

A common feature for all components is that they are service-enabled, i.e. the interaction is done by providing and requesting services. Thus, it is also possible to define and integrate other types of components that have to “talk” in the service language defined in the system. One solution for the technology side of service-orientation is to use Web services capabilities by the devices. The adopted solution is the inclusion of tools conformant with the Device Profile for Web Service (DPWS) [Microsoft, 06], namely the Service-Oriented Architecture for Devices (SOA4D), targeting services at hardware constrained devices.

2.2 High-level Petri nets for the Process Control

Each one of the components or modules belonging to the architecture has different roles and can be implemented using different technologies or techniques. As an example, the control engine embedded in SMeC or PCC can embed customized controllers, but in this approach, it is suggested to use the Petri nets formalism to power the logic control of these service-oriented components.

The Petri Nets formalism is a mathematical and graphical oriented language for design, specification, simulation and verification of systems, designed by Carl Adam Petri in 1962. It is well-suited for systems in which communication, synchronization and resource sharing are important. On one hand, as a graphical tool, Petri Nets can be used as a visual communication aid similar to flow charts, block diagrams and networks. On the other hand, as a mathematical tool, it is possible to set up state equations, algebraic equations, and other mathematical models governing the behavior of systems [Murata, 89].

High-level Petri nets are used to develop modular and flexible production systems, extending the ordinary Petri nets with some additional features in order to accommodate the requirements of service-oriented production systems. Namely, they use the concepts of stochastic/timed Petri nets, [Baccelli and Canales, 93], step-wise refinement by transition explosion (hierarchical Petri nets) and conflict detection and resolution. Moreover, High-level Petri nets models have “access” to their environment. This is done by associating transitions to input events and output actions, i.e. connecting service operations, device interfaces and other models. An important factor for flexibility is the modularization inherited from the Petri nets formalism: Petri nets models may be connected via specific ports, providing interoperability control between different models.
The use of HLPN formalism introduces the following main advantages in the development of flexible service-oriented production systems:

- Makes easier the system modeling and understanding by using a graphical and mathematical notation, and simplifies the validation, analysis and simulation of the system before to go to the operation phase.
- Drives and synchronizes the run-time behavior of the components, achieving a powerful and effective control mechanism, since Petri nets are represented and manipulated internally as a set of matrices.
- Adapts interfaces to the real physical I/Os and services via the description of transitions.
- Connects and aggregates sub- behavior models in a modular way, and makes easier the detection of conflicts and unexpected situations that require decision support.

As referred before, some architectural components may have advanced control capabilities. The following example considers a build-in Petri Net Kernel (PNK) accepting HLPN descriptions and able to run them. Fig. 2 illustrates the PCC with a PNK to coordinate/synchronize the services offered by distributed resources, according to a process behavioral model.

2.3 The Petri nets Development Framework

The Petri nets development toolKit (PndK) is an engineering framework for the specification, design, analysis, validation and control of re-configurable and modular service-oriented automation systems. This software tool, developed in C++ programming language, allows the edition, analysis (structural and behavioral) and simulation of High-level Petri net models, representing the control logic of elementary devices or the aggregate one, see Fig. 3.

![Fig. 2. Petri Net Kernel for the Process Control of Service-oriented Systems.](image)

The workplan intended in the model is to synchronize the operation between two conveyors and one robot in the middle of them that may pick and place objects from one conveyor to another or may execute a drill operation (represented as a conflict in the model). The services are provided by the MeC (2 times Transfer, Pick&Place and Drill) and can be requested by the PCC according to the HLPN model. At the end, only one service is provided by the PCC to the outside world (the Produce service) that is the aggregation of the individual ones according to the HLPN model.

![Fig. 3. Modeling and Validation in the PndK tool.](image)

The PndK framework offers the following analysis and validation modules:

- Structural and behavioral analysis, allowing 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.
- Place and transition invariants analysis, extracted from the incidence matrix, allowing confirming mutual exclusion relationships among places and functions, and the identification of work cycles.
- Performance evaluation, performed by means of the simulation of the temporized HLPN models, reflecting the temporal sequence of the system operation. It allows answering to pertinent questions, such as which states have been reached, which activities or functions have been performed and which is the history of the evolution of the system.

The PndK constitutes also the basis for a Petri-net based process control engine, by interpreting and running the process behavior in form of HLPN behavioral models. Being used in service-oriented systems, the services (encapsulating internal functionalities) are included in the Petri nets by associating them to the transitions. Internally, the PndK has an implementation of a PNK that uses the SOA4D implementation of the DPWS, providing standardized functions that allow the discovery, the binding and finally the exchange of control messages with the external partner services in a transparent manner.

A designed HLPN model can be configured to contain logic to manage (choreograph) the messaging of the composite service and its external partner services. The HLPN model includes properties to adopt the role of a control endpoint invoking the partner services and to adopt the role of a
service provider. The elements that are extended with additional DPWS properties are the transition elements. Transitions are configured with the type of message (request, response, event and node role), the type of action and the operation parameters that are processed/generated by the engine. The firing mode is used to emit messages, such as send service requests, responses to requests and send event notifications to subscribers. The enabling mode is used as guard for external events, such as incoming service requests, responses and event notifications.

3. 2D/3D TOOLS TO SUPPORT ADVANCED ENGINEERING

An important issue in the proposed High-level Petri nets – based service-oriented framework is to integrate 2D/3D engineering tools, e.g. CATIA™ or Delmia™, making easier the engineering development of modular and re-configurable production systems. The contribution of 2D/3D engineering tools can be done at different levels:

- Design and configuration of a production process, including its analysis and validation.
- Simulation of the virtual automation environment and control of a real production system.
- Integration and monitoring of virtual and real services.

In the next sections, these issues will be deeply analyzed.

3.1 Design and Configuration

An important piece of the puzzle is to have tools that automatically support the design and validation of automation systems, contributing for an easy and fast reconfiguration. The basis for the configuration of the system is the information about the physical positioning of the components which it comprises. For this purpose, the production system can be designed virtually in a 2D/3D engineering tool by arranging production components in the required layout (the user is able to pick and place 2D/3D models in a virtual shop floor). In this way, users can design the system in a graphically, intuitive and user-friendly manner.

Connecting 2D/3D models in the engineering tool can be used as input for an easy composing of component’s control models in the logic control platform, see Fig. 4. For this purpose, it is crucial the description of the connections among the different system’s components, which can be derived from the spatial position of components, their arrangement and their connection, using for example a XML-based document. When the layout information is available, describing the relative position and connection between components, it may contribute for the automatic synthesis of the whole control system.

In fact, the connection information is in any case input for a processor that is capable of doing automated composition of many behavioral models to one system model. The information about the connection of mechatronic components is provided to the logic coordination control layer to support the connection between mechatronic control models. Connecting the logic control models of different components according to this interface model, considering the port service concept [Mendes et al., 08], a complete logic model is created to represent the complete scenario (layout).

The example of Fig. 5 shows a FlexLink™ system that is composed of production components, such as conveyors units, cross units, lifters and mechanical frame elements without any production functionality except for determining the physical layout.

This approach is independent from the use of HLPN formalism, which means that the exported connection information can be input for other tools, such as reasoning engines that build up semantic models of the automation system. In our approach, the connection information coming from the 2D/3D design tool is basically used for composing HLPN control models and synthesizing the complete system step-by-step. In the proposed approach, illustrated in Fig. 4, a HLPN “compiler” loads HLPN models from a component’s HLPN control model repository for respective production components and makes synthesis of the production system model. The complete model constitutes a virtual production system that can be analyzed, validated and simulated, amongst other introspections of the system, to guarantee the correct behavior of the system. Using HLPN to represent the logic control models, formal analysis, validation and simulation of the complete system can be performed based on the functional analysis theory and linear algebra.
3.2 Simulation of Virtual Automation Environment

The second contribution of 2D/3D visualization tools in the engineering of flexible production system is to use them to test and validate the designed systems before installing them in the shop floor. Having the integration of 2D/3D tools with the HLPN-based service-oriented engineering framework, the complete development life-cycle is established, offering the possibility to experiment in a virtual 2D/3D environment the production system, helping to achieve maximum production efficiency, lower cost, better quality and short time to market. It also supports small change in the configuration and modification of the production systems, allowing evolving to the re-configurability on the fly. This means that the 2D/3D simulation will be driven by the HLPN-based logic control engine provided by PndK framework, as illustrated in Fig. 6.

Fig. 6. 2D/3D Tools to Support the Simulation and Virtual Automation Environment.

The advantages of using the simulation of the system by connecting virtual equipment to the control are mainly the following:

- A complete virtual factory automation system model can be built, including the logic control that can be simulated before to implement in the physical factory plant.
- The complete system or partial pieces of the system (e.g. lines, cells or equipments) can be debugged and validated without the need to use the (real) physical devices.
- Easy to reproduce abnormal conditions and to debug with conditions impossible or that cannot be easily created in real world (for example introducing “what-if” scenarios). Especially, dangerous tests in the real world can be done safety in this virtual world.
- Data can be reused for operator training and maintenance, and the simulations can be repeated as many times as necessary to the correct understanding and tuning of the system control or as a feasible study to reuse logic control to other systems.

The implementation of production control systems is usually carried out manually and not derived from a model-like description of the production system. Then, the correctness of the design can only be validated after the implementation phase, presenting high rates of misunderstanding and mistakes, and, as a consequence, it is very expensive [Leitão and Colombo, 05]. Using the proposed approach, the process ranging from the design to validation of the system may be iterative, till reaching the correctness of the control system behavior. In fact, if the control system can be validated and simulated during the design phase, the costs and number of mistakes detected in the operation phase are reduced significantly.

3.3 Integration and Monitoring of Virtual and Real Services

Finally, and after the complete verification of the correctness of the virtual production system behavior, the PndK platform can generate control engines to control the real production system and/or generation of code that will control and monitor the production system. The execution of the real production system also allows tracking the real-time production execution, tuning the system configuration, and introducing model changeovers and scheduling maintenance operations.

Fig. 7 illustrates an architecture consisting of real services (devices, machines, etc.) situated in a service-oriented production system and additionally an engineering environment (allowing 2D/3D and HLPN-based modeling, validation, analysis and simulation) making so-called virtual services also available to the production system. This is possible because the real and the corresponding virtual services (components) are exposing the same service interface and are sharing the same communication infrastructure.

Fig. 7. Integration of Real and Virtual Web Services.

Another feature at runtime is the monitoring of the real production system – in this case the engineering tool is used to monitor, using event-based mechanisms, the synchronization of the virtual production system with the real production system at runtime, showing actions that are performed with the real components also at the virtual component.

4. EXPERIMENTAL CASE STUDY

The experimental case study is a FlexLink™ production cell that is built by using autonomous DPWS-enabled smart devices [Cachapa et al., 07], representing conveyors, cross-
tables and end lifters arranged in a closed-loop configuration, featuring a decision point (identified by the question mark), as illustrated in Fig. 8. The decision point represents a fork in the paths of the work-piece, upon which it can either continue straight on to the end lifter, or turn in the direction of one of the two workstations.

![Fig. 8. Representation of the Experimental Production Cell.](image)

Once the devices are assembled in the model and properly configured, the system is ready for simulation. A virtual pallet representing the work-piece is added to the system so that the model can react to control commands and changes in the environment. As the simulation starts, each smart device will automatically initialize, launching and configuring an unique Web service interface and start listening on the network for control commands. At this point, services are available and any compliant application can search in the network and use them.

The HLPN based approach for composition and coordination of service-oriented production systems has been implemented and tested with virtual services that were hosted by Delmia Automation engineering tool. A synthesized behavioral model has been composed from the constituent component behavioral models according to the layout of the virtual production cell. The connection of the models is done through specific interfaces (ports), which, when connected, drive the correct collaboration between the two connected components. The configuration of the models with DPWS properties allows the embedded HLPN engine to discover and invoke the production services provided by the Delmia tool. When executing a HLPN inside the PnDK, the status of the HLPN model is made visible in the graphical Petri net editor, giving information about the current marking, the enabling- and the firing-modes of the executed Petri net.

Besides being an early experimentation and still in development, it already showed some of the intended features. Systems can be modeled using the combination of 2D/3D objects and their layout/connection information in parallel with the development of the control logic using HLPN models. The simulation and analysis can be done previously offline by using the 2D/3D model corresponding to the real one. It is also worth of mentioning that the transition to the real system is done in such a way that the same control models (previously analyzed) are maintained, thus not requiring re-engineering of the control system. Moreover, the use of Web services between applications and virtual/real equipment facilitates not only the communication, but also the integration of the virtual/real scenario.

5. CONCLUSIONS

This paper proposed the integration of High-level Petri nets - based service-oriented frameworks with 2D/3D engineering tools, aiming to achieve an integrated methodology for the easy design, configuration, validation, simulation, control and monitoring of modular and reconfigurable production systems. From the methodology to the application, several aspects could be proved, namely the use of High-Level Petri nets to drive the system's behavior (both virtual and real), easiness and less effort in the transition from the design and analysis phases to the operation and useful real-time monitoring capabilities demonstrated by the tools.

Future work includes more experimentation with the real hardware and enhances the transition efforts from the virtual scenario to the real one. Other aspects are being researched such as the definition of layout information of equipment, dynamic connection of devices/components, exception handling and reconfiguration mechanisms.

REFERENCES


