J Intell Manuf (2012) 23:2551–2563
DOI 10.1007/s10845-011-0591-8

# Integration of virtual and real environments for engineering service-oriented manufacturing systems

**Paulo Leitão · J. Marco Mendes · Axel Bepperling · Daniel Cachapa · Armando W. Colombo · Francisco Restivo**

Received: 29 March 2010 / Accepted: 8 September 2011 / Published online: 22 September 2011


**Abstract** Engineering frameworks are currently required to support the easy, low-cost, modular and integrated development of manufacturing systems addressing the emergent requirements of re-configurability, responsiveness and robustness. This paper discusses the integration of 2D/3D digital software tools with Petri net based service-oriented frameworks to allow the design, configuration, analysis, validation, simulation, monitoring and control of manufacturing systems in a virtual environment and its posterior smooth migration into the real "physical" environment. An experimental case study was implemented to validate the proposed concepts, using the Continuum platform to design, compose, analyze, validate and simulate the Petri nets based service-oriented manufacturing control system, and the Delmia Automation™ software suite to support the rapid prototyping and the easy simulation of the designed control solution. The experimental results prove several aspects of the proposed approach, notably the smooth migration between the design and the operation phases, one of the main objectives of the work.

**Keywords** Distributed production control systems · Re-configurable manufacturing systems · Service-oriented architectures · Petri nets

P. Leitão (✉)
Polytechnic Institute of Bragança, Quinta Sta Apolónia, Apartado 1134, 5301-857 Bragança, Portugal
e-mail: pleitao@ipb.pt

P. Leitão · F. Restivo
LIACC—Artificial Intelligence and Computer Science Laboratory, R. Campo Alegre 102, 4169-007 Porto, Portugal

J. M. Mendes · F. Restivo
Faculty of Engineering of University of Porto, Rua Dr. Roberto Frias, 4200-465 Porto, Portugal

A. Bepperling · D. Cachapa · A. W. Colombo
Schneider Electric Automation GmbH, Steinheimer Str. 117, 63500 Seligenstadt, Germany

A. W. Colombo
University of Applied Sciences Emden/Leer, Constantiaplatz 4, 26723 Emden, Germany

## Introduction

The current global and customized markets require the use of new paradigms and technologies to enable the development of low cost, modular, flexible, intelligent and re-configurable manufacturing systems. Service-oriented architectures (SoA) (Jammes and Smit 2005a) is a suitable paradigm to address the described challenges, based on the idea of encapsulating the resources functionalities as services that can be offered, searched and used by other entities (Melzer et al. 2007), without knowing their underlining implementation. For this purpose, service providers publish the services they want to offer in a service registry, and the service requesters search the services they want to use through the help of discovery methods. Such systems are mainly characterized by providing encapsulation, modularization, interoperability, composition and reusability. The SoA paradigm was originally applied in electronic commerce and business systems, using the Web services technology, but is being progressively adopted by other fields. The project SIRENA (Jammes and Smit 2005b) has contributed for the visibility of the SoA-based automation by providing an extension of the SoA paradigm into the realm of low-level embedded devices, such as sensors and actuators. Since then significant research is going on covering the engineering of such systems, including the modeling, semantic description and collaboration, namely the research project EU FP6 SOCRADES (http://www.socrades.org/).

In service-oriented systems, particular challenges are the description of the processes that regulate the system behavior and the synchronization and coordination of the execution of the services offered by distributed entities to achieve the desired behavior. A possible solution is the use of the Business Process Execution Language (BPEL) (OASIS 2007) but it was not designed for automation-specific problems, is too complex for embedded devices, does not include validation features and cannot be used for the internal logic of software components. Other alternative is based on IEC 61131-3 languages (IEC 61131-3 2003) with the objective of adapting industrial standards to SoA, but amongst others it also misses the formalization and validation issues. The use of the Petri nets formalism (Murata 1989; Zurawski and Zhou 1994; van der Aalst 1998) for the process behavior description and control in service-oriented manufacturing systems, allows a rigorous specification of the system behavior and also the formal analysis and validation, taking advantage of its powerful theoretical foundation.

The framework for reconfigurable manufacturing systems proposed by Mendes et al. (2008) and refined in Mendes et al. (2009b) is based on distributed control systems and service-oriented design paradigms, with the inclusion of a Petri net-based language for the process description of intra- and inter-coordination activities. The proposed integrated approach permits the modeling, analysis, validation, simulation and execution of the manufacturing control system, covering completely the system development life-cycle. However, the design and implementation of such collaborative and re-configurable manufacturing systems and their supervisory control systems are complex, time-consuming and very expensive tasks, with the design correctness only validated during the system operation. A challenge to overcome this problem is the consideration of virtual prototyping to support the easy and user-friendly design and validation of manufacturing systems, allowing the detection of misunderstandings and mistakes during the design phase, i.e. before to go into implementation. In fact, the use of virtual reality in simulating manufacturing environments gives to designers the opportunity to play a pro-active role in identifying flaws and optimizing the design (Ramaswamy and Yan 1998). Note that in spite of the potential benefits provided by the Petri nets formalism in the analysis, validation and simulation of the dynamic system behavior, the direct analysis of the Petri nets models and their token-games is not an easy task and requires a certain level of expertise.

The main objective of this work is to specify an engineering framework for the development of reconfigurable manufacturing systems supporting the smooth migration between virtual and physical manufacturing environments, by using SoA principles and the Petri nets formalism. For this purpose, the Petri nets based service-oriented manufacturing systems framework will be integrated with 2D/3D digital tools, allowing the advanced engineering of modular, flexible and re-configurable manufacturing systems, with the 2D/3D digital tools operating as front-end of the engines running the Petri nets behavior models. In other words, it allows the virtual design, validation and simulation of manufacturing systems before to go into operation. One important feature of this approach is the reduction of the effort and cost associated to the migration between the virtual/simulation environment to the real "physical" system, without any changes in the system configuration and using the same control models (only the device services need to be changed). Moreover, the paper presents an industrial proof-of-concept demonstrator that illustrates many of the desired features of the proposed approach: development of autonomous and cooperative smart devices, easy system configuration and development of the control structures, and straightforward virtual-to-real migration of the control model.

The reminder of the paper is organized as follows: first, section "A Service-oriented approach for manufacturing systems using Petri nets" introduces the Petri nets based service-oriented approach for reconfigurable manufacturing systems and section "The Continuum development tools" overviews the Continuum platform developed to support the design and implementation of such service-oriented applications. Section "Towards a virtual environment for the service-oriented manufacturing engineering" describes the advanced topics and features of using 2D/3D digital software tools to achieve virtual manufacturing systems and the mechanisms to support the smooth migration between virtual and real environments. An experimental demonstrator was used and explained in section "Experimental case study" to test the proposed approach. Finally, section "Conclusion" rounds up the paper with conclusions.

## A service-oriented approach for manufacturing systems using Petri nets

An engineering framework was designed to face the challenge of supporting modular and re-configurable manufacturing systems, based on distributed control systems and service-oriented design paradigms, with the inclusion of the Petri net-based language for the process description of intra- and inter-coordination activities. The proposed methodology permits the modeling, analysis, validation, simulation and execution of the system in an integrated manner. Additionally, besides the service-enabled embedded controllers running Petri nets orchestration engines, appropriate tools are required to configure the system. One of the possibilities that is being researched is the inclusion of virtual environments to design, configure and test manufacturing systems, with the migration to the "real" system performed without significant changes in the system configuration.

## Service-oriented smart embedded devices and system architecture

The service-oriented manufacturing system approach, proposed by Mendes et al. (2008) and refined in Mendes et al. (2009b), is built upon a set of distributed components, each one providing a set of services that encapsulates its internal functionalities. These components are designated as smart embedded devices and are responsible of several tasks in the system, namely coordinating the operation of industrial machinery, integrating shop-floor equipments into the service network, exchanging information with other devices and computing systems, taking decisions about different possibilities attending current criteria and synchronizing the inter-device operations. In combination with the machinery and other shop-floor equipments, the smart embedded devices, connected together via the network, form the distributed control system (DCS) of the shop-floor (see the lower part of Fig. 1).

The smart embedded devices follow an anatomical-like structure, comprising several functional modules, to provide a modular and event-driven design and deployment (see the bottom-right side of Fig. 1). The *communication module* is responsible to handle the interaction with the other components, requesting and providing services. The *orchestration engine* defines an interpreter for the coordination of services and other actions based on a workplan, and the *decision support module* provides decision support for conflict resolution and unexpected situations. The *device interface* permits the access to the physical device, such as setting outputs or reading inputs. These modules may be implemented by using different technologies. As an example, the inter-entity communication can be implemented using the SoA for Devices (SOA4D) implementation of Device Profile for Web Services (DPWS) (OASIS 2009) or using any other communication technology such as Common Object Requesting Broker Architecture (CORBA) or OPC (OLE for Process Control) Unified Architecture (OPC-UA).

Complementing the activity of smart embedded devices and especially for their configuration, engineering tools are needed as well. These elements are the basis to support an operational DCS and can be integrated into the Manufacturing Execution System (MES) level. A MES system is responsible to guide the product execution, providing information about how to execute the products based on the product and process models. Smart embedded devices can dynamically request information to MES to support the decision-making about production execution (e.g. to which machine a pallet of products should be transported). A more extended integration can be followed with the Enterprise Resource Planning (ERP) layer for the resource planning and usage, as well attending business strategies (see Qiu and Zhou 2004; Gibson et al. 1999).

In between, service-orientation is a deal when the cross- and intra-layer integration are done via SoA. The SoA approach permits not only the communication between the parts in the system, but also defines rules that makes possible the conjunct interoperability and distributed coordination of activities and tasks. A common service technology (e.g. Web services) and standard protocols are fundamental to achieve the vision of a SoA-enterprise, as well as engineering and orchestration solutions.

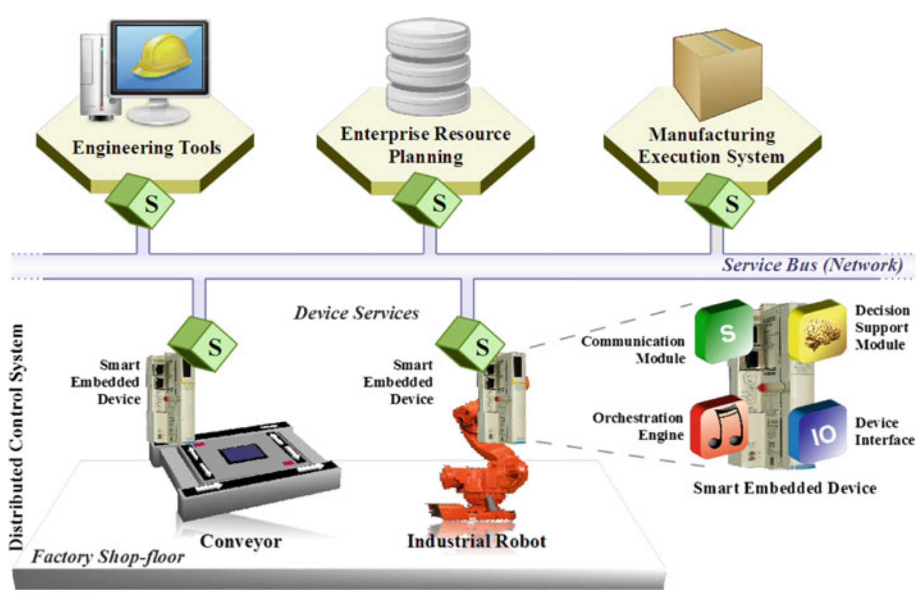



**Fig. 1** SoA for manufacturing systems using smart embedded devices

Service engineering and orchestration with Petri nets

The behavior description and orchestration in service-oriented systems can be performed by using several languages, such as WS-BPEL (Puttonen et al. 2008), IEC 61131-3 (Mathes et al. 2009) and Petri nets (Mendes et al. 2009b). If the technology and the application domain are avoided, these languages are quite similar (nothing more than a mathematical abstraction to design the digital logic, computer programs and work-plans). The complexity of the language may affect its implementation, and as such, a simple but powerful extensible basis fits the needs for this work. Therefore, Petri nets formalism is used not only as a language for the description of service workflows, but also for their analysis, composition and execution (Mendes et al. 2010).

Petri nets are a mathematical and graphical oriented language for the design, specification, simulation and verification of discrete systems. It is well-suited for systems in which communication, synchronization and resource sharing are important characteristics. Therefore, Petri nets and their extensions are widely used for modeling of discrete event dynamic systems including production systems and networks (Tsinarakis et al. 2005). 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, the system behavior is represented by means of algebraic equations and other mathematical models (Murata 1989).

The language used in this work for the process behavior description and execution is Petri net derived, concerning the introduction of service calls, responses and events. Petri nets models have "access" to their environment by associating transitions to input events and output actions, e.g. setting actuators or reading sensors. Moreover, time consideration (Baccelli and Canales 1993), parallel execution and coordination of Petri net branches and conflict management are also considered. An important factor for flexibility is the modularization inherited from the Petri nets formalism: Petri nets models may be connected via specific ports to permit their composition. Ports are not only used for the connection of models, but also defined as a set of service operations that can be used externally for means of interoperability control between different models. In fact, ports can also be associated to the equipment that is modeled (e.g. a lifter or a conveyor), representing the logical input/output for the physical system (e.g. input and output of pallets to/from the lifter are represented by ports in the models that can be connected to other models).

The use of the Petri nets formalism introduces several advantages in the development of flexible service-oriented manufacturing systems:

- Makes easier the system modeling and understanding by using a graphical and mathematical notation, and simplifies the analysis, validation and simulation of the system before its deployment into real controllers.
- Aggregates and composes behavior models in a modular way and makes easier the detection of conflicts and unexpected situations that require decision support.
- Drives and synchronizes the run-time behavior of the components, achieving a powerful and effective control mechanism.
- Adapts interfaces to the real physical I/Os and services via the description of transitions elements.

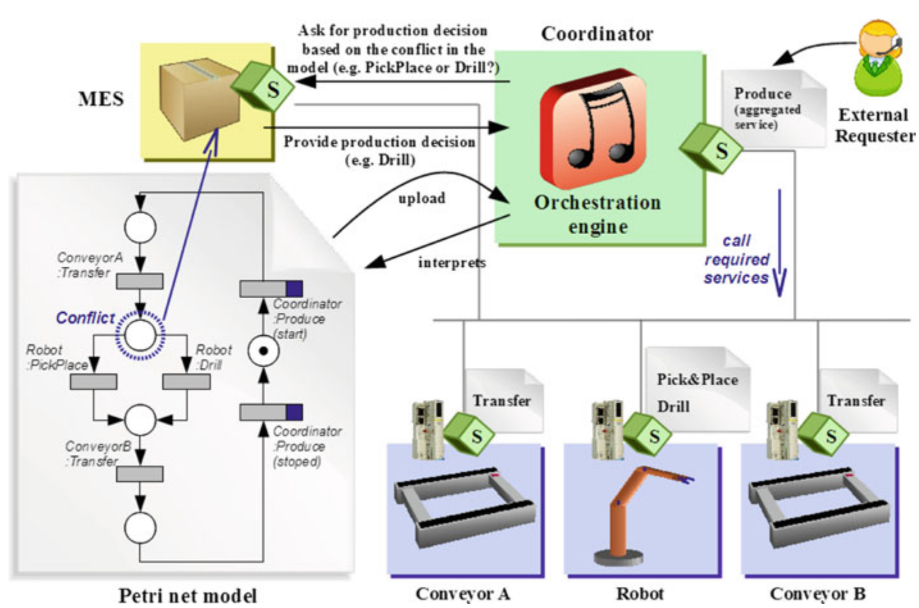



**Fig. 2** Petri nets -based orchestration for a service-oriented shop-floor system

The Petri net based orchestration engine for smart embedded devices was implemented to interpret this type of Petri net language. In fact, the architectural components, particularly smart embedded devices with orchestration engine, may have advanced control capabilities by considering Petri nets orchestration engines that interpret and run Petri nets behavioral models.

Figure 2 illustrates a smart embedded device with a Petri net orchestration engine (the coordinator) that coordinates/synchronizes the services offered by the distributed resources according to a Petri nets process behavioral model. Note that the orchestration engine was deliberately separated from the smart embedded devices in a new component (coordinator) to simplify the comprehension of the figure.

The objective here is to synchronize the operation of two conveyors and one centered robot 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 smart embedded devices representing the conveyors A and B (i.e. *Transfer* services), and by the robot (i.e. the *Pick&Place* and *Drill* services), and can be requested by the coordinator. The coordinator offers the aggregated service *Produce* that is the composition of the individual ones according to the Petri nets model. The conflict in the model can be solved with the support of the MES system, based on the production information.

For more detailed information on this language, please consult (Mendes et al. 2009b). The Petri net approach was also used in the SOCRADES project, not only for modeling, but also as an executable language for a real industrial system using embedded devices (Mendes et al. 2010).

## The Continuum development tools

The Continuum development tools (CDT) (Mendes et al. 2009a) is being developed with the objective to facilitate the design, analysis, validation and control of distributed service-oriented automation and manufacturing systems, by using Petri net structures for the process description and execution. It considers the need to integrate several software packages, especially for the configuration of smart embedded devices with distributed Petri net orchestration engines.

The main component of CDT is the Continuum Development Studio (CDS), illustrated in Fig. 3, which is based on an extensible document/view framework and allows the visual and formal design, validation and discrete simulation of Petri nets models, besides using them as a control mechanism for service-oriented systems. The CDS supports the edition of individual Petri net models representing the behavior of automation devices and/or system processes considering a hierarchical perspective, i.e. allowing the stepwise refinement of the control models to introduce detailed information.

A Petri net model can be configured to contain the logic to choreograph the messaging of provided and requested services, i.e. the model can be associated to service interactions following the logic represented by the Petri net. The elements that are extended with additional DPWS properties are the transition elements. For this, it is possible to import Web Service Description Language (WSDL) files and associate services to the transitions of the modeled Petri net. Transitions are configured with a type of message (i.e. request, response, event and node role), a type of action and operation parameters that are processed/generated by the orchestration

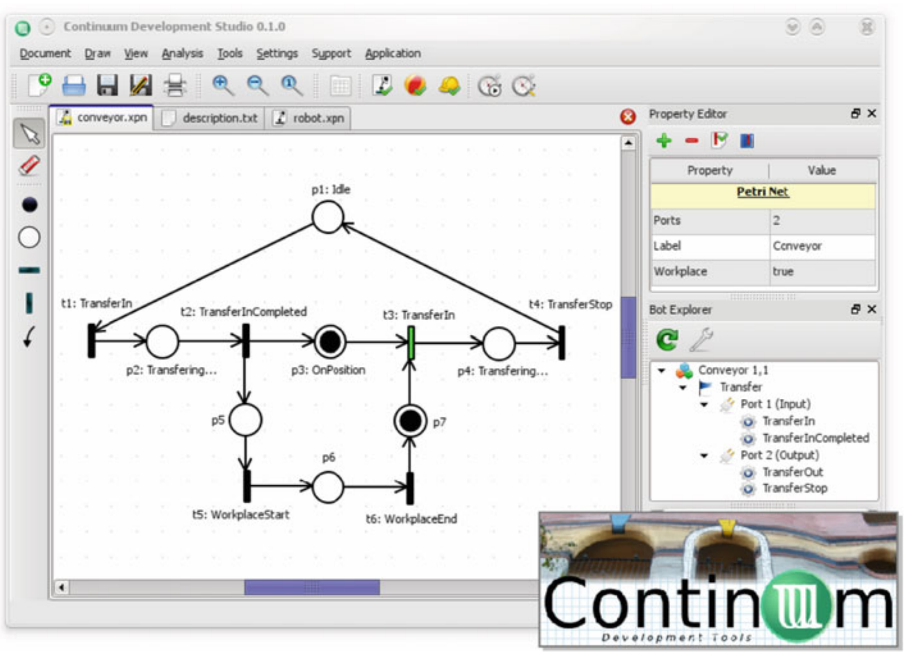



**Fig. 3** Modeling of Petri nets models using the Continuum Development Studio

engine. When the orchestration engine is running a Petri net model, the enabling mode of transitions is used as guard for external events, such as incoming service requests, responses and event notifications. The firing mode of transitions is used to emit messages, such as send service requests, responses to requests and send event notifications to subscribers.

A main characteristic included in the CDS tool is the ability to analyze a Petri nets model, verifying its correctness. The following analysis and validation modules are available in the CDS tool:

- *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. This information constitutes valuable information to support decision-making services.
- *Performance evaluation*, performed by means of the simulation of the temporized Petri nets 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.

Another important feature regarding distributed systems is the ability to aggregate elementary devices, in this case connecting and synthesizing individual Petri net models. The Petri nets Composer (PNC) tool, included in the CDT tool, synthesizes automatically individual Petri nets models according to a layout configuration file that describes the connections among the system's components. The composition tool can be avoided if connections are known or the composition of models is unneeded.

Finally, smart embedded devices have to be configured with the information specified in the CDS tool. This can be done with the configuration and deployment tools provided by the Continuum platform. The configuration generator creates deployment files from the system models (i.e. the designed Petri net models with the layout information), the WSDL(s) files and the device description files, to create the direct connection between binding reference names used in the models and the real DPWS devices/services. The deployment manager loads the generated deployment files into a target device (e.g. a Telemecanique Advantys STB device), which must host a Petri nets orchestration engine. Once the configuration files are received, the orchestration device is ready to lookup for the required services (expressed in the Petri net model) and orchestrates them according to the logic described in the Petri net.

In spite of all the tools included in the CDT, a virtual environment representing the physical manufacturing system that supports the testing of the developed models is missing. Additionally, the information provided by the virtual environment can be used for the composition of models (e.g. machine 1 connects to machine 2 via a specific port) and to simplify the migration from the PC-engineering systems to the factory shop-floor.

## Towards a virtual environment for the service-oriented manufacturing engineering

The use of the Petri nets formalism is an important contribution to support the life-cycle development of service-oriented manufacturing systems. However, the understanding and visualization of the results from the formal Petri nets analysis, validation and simulation is not an easy task, especially for those unfamiliarly with the Petri nets concepts. In this way, and aiming to achieve a truly and complete virtual environment for the development of such applications, it is crucial the integration of 2D/3D digital software packages, such as ARENA$^{TM}$ or Delmia Automation$^{TM}$. The integration of such platforms contributes for the achievement of an engineering framework covering the complete life-cycle engineering of service-oriented manufacturing systems, namely allowing the design, configuration, analysis, validation, simulation, control and monitoring of the production system in a 3D environment.

This section analyzes the several contributions resulted from the integration of digital software platforms.

### Design and configuration

The engineering of manufacturing systems requires tools supporting their design and validation contributing for an easy and fast system reconfiguration. The basis for the properly configuration of the manufacturing system is the information about the physical positioning of the automation components and their inter-logic connections. Using a digital software tool, the users can design the system in a graphically, intuitive, virtual and user-friendly manner, by picking and placing 2D/3D models of the manufacturing devices, such as machines, robots and conveyors.

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 provided by the virtual environment, may use a XML-based document. As an example, the XML file for describing the connections between two conveyors, where the conveyor A is the client and the conveyor B is the server, looks as follows:

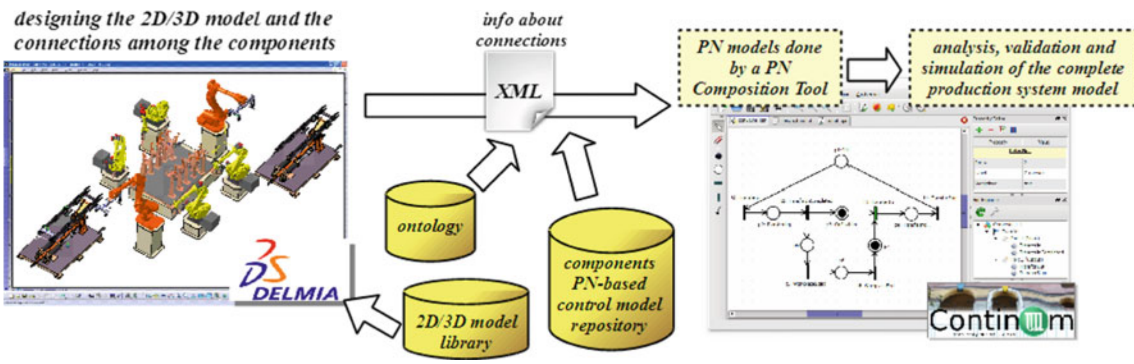



**Fig. 4** Virtual manufacturing environment using digital software tools and Petri nets development platforms

```
<?xml version = "1.0" encoding = "UTF − 8"? >
  < connections >
    < connection
      resource1 = "Conveyor A" port1 = "out"
      resource2 = "Conveyor B" port2 = "in"
        / >
   < /connections >
```

In the example, only one connection is included, but multiple connections can also be defined. For this purpose, a new <connection/> tag must be defined inside the <connections> tag. If the conveyor B would be connected to a conveyor C on the right, then this information is represented by one additional <connection/> tag.

In the proposed approach, the information about the system components, their layout and inter-logic connections, provided by the 2D/3D design tool through the XML files, is basically used for composing the Petri nets control models representing the behavior of individual automation components. The composition of Petri nets models can be done offline, generating a synthesized Petri net model that is the composition of several individual ones, or online, where individual Petri nets models are maintained and executed in their distributed devices and synchronized together on the fly via the network (Mendes et al. 2010).

The offline composition comprises the generation of a new Petri net model based on the connection of two or more individual ones, representing a new inter-logic model. The basic idea is to match two transitions from different models by connecting them via a place (and corresponding arcs). This approach, besides to simplify the development of bigger and more complex models, also facilitates the synchronization of models viewed as individual entities. The offline composition can be performed by using the PNC tool that allows the user to create simplified individual models and then connect them together into a global model. In the proposed approach, illustrated in Fig. 4, the PNC loads the Petri nets models from a component's Petri nets control model repository for the respective manufacturing components and synthesizes the manufacturing system model. Connecting the Petri nets control models of the different manufacturing devices according to the information provided by the digital software tool, and considering the port service concept to connect individual behavioral models (Mendes et al., 2008a), a complete logic Petri nets model is created to represent the complete system.

The complete model constitutes a virtual manufacturing system that can be analyzed, validated and simulated, amongst other introspections of the system, to guarantee the correct behavior of the system. Using Petri nets 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. However, this approach is independent from the use of the Petri nets formalism, which means that the exported connection information can be used by other tools, such as reasoning engines that build up semantic models of the manufacturing system.

Online composition means that each model runs separately in its own orchestration engine and they are synchronized via a connection logic, achieving a truly distributed manufacturing system through the distributed orchestration and the synchronization thereof. In this case, the connection logic represents a service-based communication act, where services and their operations are described in Petri net models. The online composition requires that Petri nets models have information on how to invoke and represent services to synchronize with the other models, which is done by describing transitions in the Petri net model. A transition willing of sending a request/response or an event must be enabled, and the action is done when it fires. In the other hand, a transition receiving a message from a request, response or event, will only fire if it is enabled and the message is there.

Offline composition is used to limit the use of devices, network traffic, but introduces more complex models to be orchestrated (considering the limitations of embedded devices), while online composition focus in the distributed

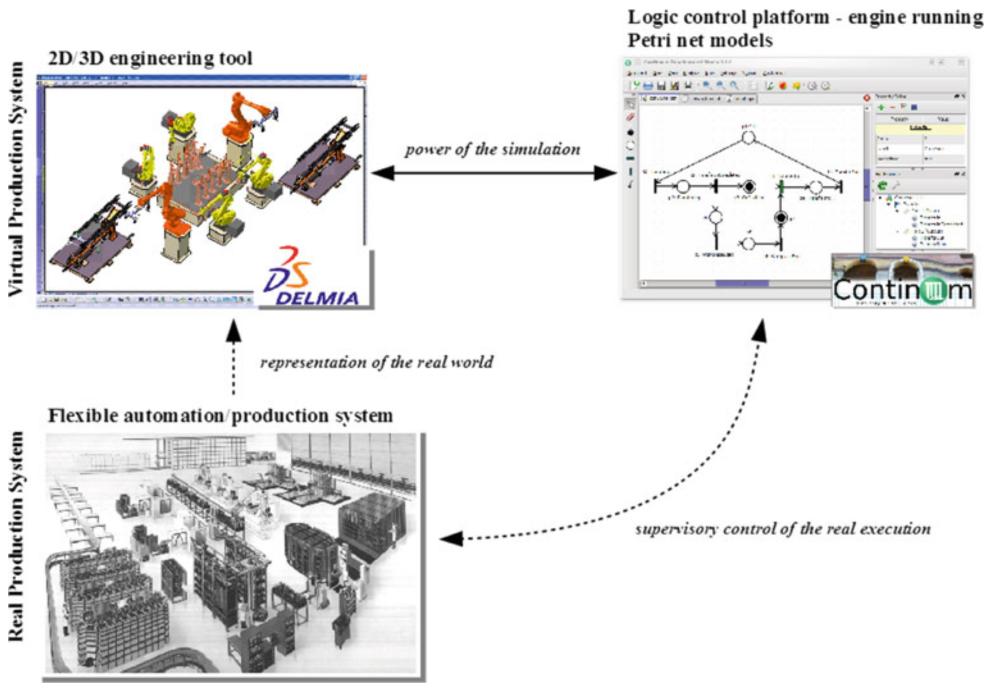



**Fig. 5** Simulation of service-oriented manufacturing systems using the virtual environment

orchestration. The correct division and use of the composition strategies depends always on the available resources, the optimization strategies and the layout of the system, but orchestration models can be individually developed without knowing this information.

### Analysis, validation and simulation

The analysis and validation of systems is crucial, not only to prove that they are free of errors, but also to test if they perform the desired specifications.

Simulation has been commonly used to study the behavior of real world manufacturing systems to gain better understanding of underlying problems and to provide recommendations to improve those systems (Ali et al. 2005). Being the observation of real systems very expensive and sometimes cumbersome, a simulation model is an easier way to identify bottlenecks and to enhance system performance in terms of productivity, queues, resources utilization, cycle times and lead times (Ali et al. 2005). The digital software tools can also be used in the engineering of flexible manufacturing systems to support the analysis, validation and simulation of the designed control systems before deploying them in the shop floor. Integrating the 2D/3D tools with the Petri nets based service-oriented engineering framework, the complete development life-cycle is established, offering the possibility to analyze and experiment the manufacturing system in a virtual environment, helping to achieve maximum production efficiency, lower cost, better quality and short time to market. Production strategies can be easily verified in the virtual manufacturing system environment, as also small changes in the configuration of the manufacturing systems, e.g. adding or removing devices or changing the connections between the devices.

In the proposed approach, the structural analysis of the control system models is performed in the Continuum platform, allowing verifying its structural and behavioral properties, extracting conclusions about the functionality of the system, using the features associated to the Petri nets formalism provided by the CDS described in the previous chapter. Namely, it allows the verification of the existence of deadlocks, the bounded capacity of resources, the conservativeness, the possible control sequences (through the extraction of the T-invariants) and the existence of structural and behavioral conflicts in the system. Similarly, the simulation of the system behavior is performed in the Continuum platform, through the execution of the token-game, allowing checking the system compliance with specified performance indexes, such as the manufacturing lead time, the throughput of the system, the percentual use of the resources and the manufactured parts per time units. However, the visualization of the simulation through the token-game may be not easily understandable and require the usage of more user-friendly digital environments.

In the proposed approach, as illustrated in Fig. 5, the 2D/3D simulation provided by the digital software tools is driven by a Petri nets based logic control engine provided by the Continuum framework. This is possible due to the inclusion of a Web service framework in the simulation software, so that the virtual equipments can expose services, similar to those exposed by the real equipments, to be used by the Petri nets orchestration engine. In this scenario, the virtual environment behaves like the real system offering valuable testing scenarios. For the control system it is indistinguishable to be connected to the virtual environment or to the real system.

The advantages of using the virtual environment to simulate the service-oriented control system 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.
- The reproduction of abnormal conditions or scenarios that are impossible or cannot be easily created in real world is an easy task. Especially, dangerous tests in the real world can be done safely in this virtual world.
- The control strategies can be validated/improved/optimized under the different "what-if" scenarios that can occur in the real world and before its practical realization.
- The 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.

Using the proposed approach, the development process ranging from the design to validation of the system may be iterative, till reaching the correctness of the control system behavior. In fact, errors and mistakes detected in the design phase can be easily corrected before the deployment into the real shop floor controllers.

### Control and monitoring

After the complete verification of the correctness of the manufacturing system behavior performed during the design phase and using the virtual environment, the service-oriented control solution should be deployed into the real controllers distributed in the shop floor. In the Continuum platform, the deployment and configuration into Petri net-enabled orchestration controllers is done by generating deployment files representing the information of services behavior, namely the Petri nets models and service interfaces. Once a controller is discovered in the network, the upload of this deployment information is possible.

The monitoring of the execution of the real production system allows tracking the real-time production execution, providing production indexes, supporting the tuning of the system configuration and the adjusting of control strategies, and introducing model changeovers and scheduling maintenance operations. In this case, the digital software tool is used to monitor in a graphical manner, using event-based mechanisms, the runtime synchronization of the virtual production system with the real production system. This allows showing the actions that are being performed by the real components at the virtual environment.

The proposed approach can simultaneously handle services provided by the real devices situated in a service-oriented production system and services provided by the 2D/3D engineering environment, making the so-called virtual services also available to the real production system. This is only possible because the real and the corresponding virtual services (components) are exposing the same service interface and are sharing the same communication infrastructure.

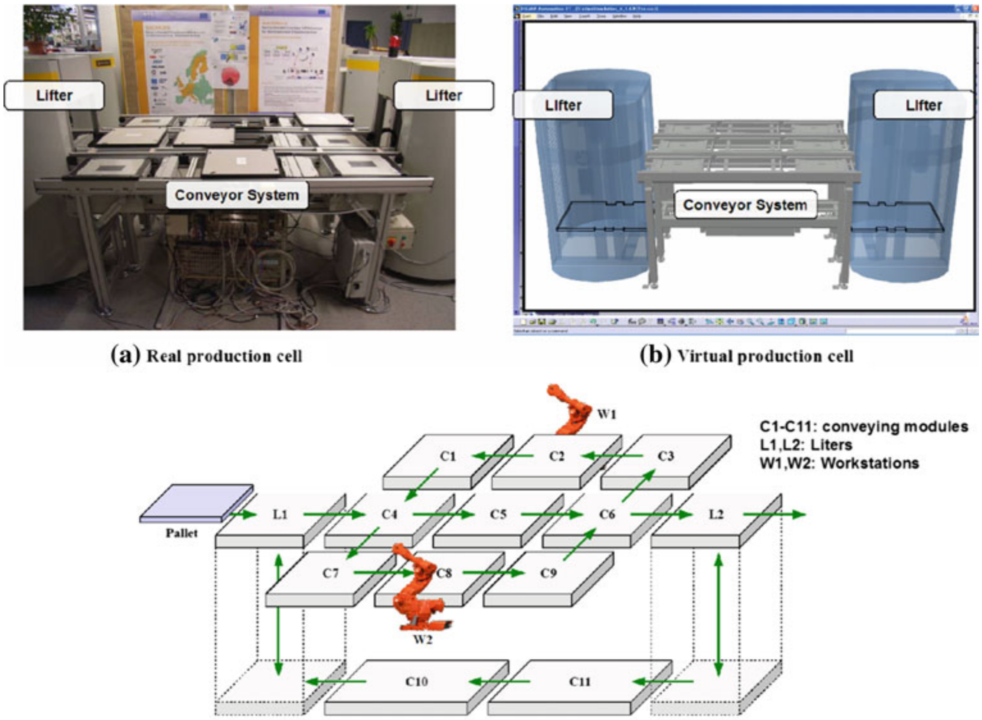
## Experimental case study

The experimental case study is a material handling system based on a flexible electromechanical assembly cell provided by FlexLink$^{TM}$, originally implemented as an industrial demonstrator of the results obtained in the EU IST FP6 SOCRADES project. It comprises a system of virtual and real autonomous DPWS-enabled smart devices (Cachapa et al. 2007), representing conveyors, cross-tables and end lifters arranged in a closed-loop configuration, as illustrated in Fig. 6a. The virtual system was specified in the Delmia Automation$^{TM}$ tool, as illustrated in Fig. 6.b and the layout of the system with the numerated modules and the possible paths for the pallets is shown in Fig. 6c.

The objective was to develop a service-oriented control application for the experimental system using Petri nets for the process behavior description, composition and coordination. The virtual representation in Delmia Automation$^{TM}$ tool permits the simulation and definition of the system's layout, which can be exported and used for the connection of individual Petri net models.

### Build up and simulation of the virtual manufacturing system

The devices in this manufacturing system share a common goal: to transport a pallet from one place to the other. These similarities allow us to abstract the individual characteristics of each device into a common set of high-level operations that enable the user to simply order transfer operations from one device to the next one with the same interface, regardless of the devices being accessed. For this purpose, the *Transfer Service* is used, describing the operations required to move




(a) Real production cell (b) Virtual production cell

(c) Layout of the production cell

**Fig. 6** Representation of the experimental manufacturing system

**Table 1** Operations and events implemented by the transfer service

| Operations | Events |
|---|---|
| TransferIn(port) | TransferCompleted |
| TransferOut(port) | Failure |
| TransferStop | |
| GetStatus | |

the pallets between devices and the events necessary for the synchronization. Table 1 illustrates the operations and events implemented by the *Transfer Service*.

These operations, implemented as a self-contained Web service interface, are connected to the device's underlying logic control, which interacts with the device's hardware in order to achieve the expected functionality. This layered architecture hides the complexity associated with the implementation of the machine code from the consumer of the service. It also harmonizes the interfaces for all the devices of the same functional nature, i.e. all of the devices capable of transporting work-pieces implement the *Transfer Service*, regardless of their physical and functional differences.

The service-oriented control system was developed using the Continuum platform, with the behavioral models of each device, i.e. lifters and unidirectional and cross conveyors, being described using Petri nets. The behavior of each equipment module is represented by an individual Petri net, which uses the transfer service of a module to synchronize its activity. This approach simplifies the design, because models may be reused for similar equipment and also the engineer does not need to think about the full system at the beginning.

A synthesized Petri nets behavioral model has been composed from the individual behavioral models according to the layout of the virtual production cell, using the information about the device connections provided by the Delmia Automation™ tool. The connection of the models is done through specific interfaces (ports), which, when connected, drive the correct collaboration between the two connected components. For this, each model has connection information, so that the PNC tool can compose individual Petri nets control models to reach the whole control model. Another option is to use Web services for the synchronization of different models. The configuration of the models with DPWS properties allows the embedded Petri nets engine to discover and invoke services that are available in both virtual and real environment.

The final models can be uploaded into a Petri nets control application running in a PC or embedded in an industrial controller. Once the services are available, the Petri nets controller can be used to coordinate the activity of the services according to the predefined composed models. Figure 7 represents the controller managing the services of modules C1–C3 and others, depending on how the composition was done and how many controllers are used. Since the same service

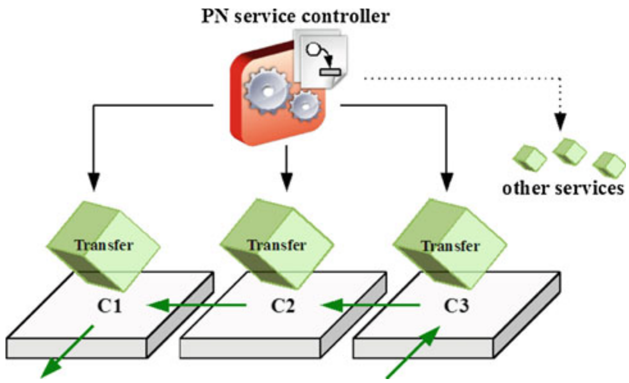



**Fig. 7** Petri net controller for services

layout is presented in both real and virtual systems, the control approach of Fig. 7 can be used in the real-time operation of the equipment as well as in the simulation performed in the Delmia Automation$^{TM}$ tool.

The achieved virtual system offering virtual services combined with the Petri nets control models constitutes the virtual manufacturing system that can be simulated. When the simulation starts, each virtual device will automatically initialize, launching and configuring an unique Web service interface and start waiting for control commands. At this point, services are available and any compliant application can search in the network and use them. The execution of the simulation is performed by running the Petri nets models, i.e. running the token game. A pallet representing the work-piece is added to the system so that the model can react to control commands and changes in the environment.

Figure 8 shows the system in operation using the different engineering tools. The Petri nets 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$^{TM}$ engineering tool (Fig. 8a) and the "real services" by the production cell (Fig. 8b). When executing a Petri nets model inside the Continuum platform, its status is made visible in the graphical Petri nets editor, giving information about the current marking, the enabling- and the firing-modes of the executed Petri nets model (Fig. 8c). The Delmia Automation$^{TM}$ engineering tool can also be used for monitoring the activity of the system by capturing events that are sent and thus updating the 3D representation of the system (Fig. 8a). Due the inclusion of the service framework into Delmia Automation$^{TM}$, it can also be independently tested and simulated with the Petri net controller. The Materna Service Explorer (Fig. 8d) can be used to see the discovered services in the system and also to call their operations by the user.

At this stage, alternative tests can be performed in the system by changing its parameters, introducing the degradation of the conveyors and including more pallets that leads to the increase of the traffic congestion. Besides the execution

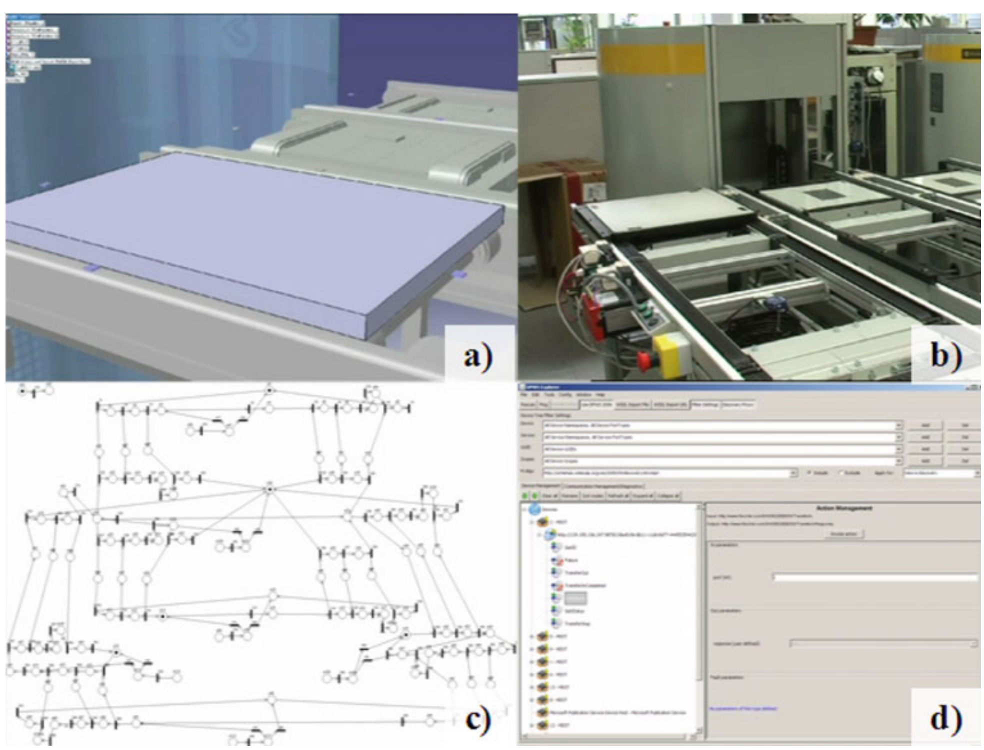



**Fig. 8** Operation of the system using the different tools: **a** Delmia Automation$^{TM}$, **b** physical system, **c** Continuum, **d** Materna Service Explorer

in the virtual environment, through the representation in the Delmia Automation$^{TM}$, the control solution was also applied and executed in the real physical system.

### Analysis and discussion

In spite of being an early prototype experimentation, some important concepts were observed, namely, that complex systems can be easily modeled, analyzed and simulated in a truly virtual environment, that combines 2D/3D simulation software tools and Petri nets computational environments. A demonstration video including this approach can be seen at www.youtube.com under "IP SOCRADES Demonstration of Service-Oriented Architecture integrating Real and Virtual Devices in the Electronic Assembly Scenario" and "SOCRADES Introducing a Service Oriented Infrastructure for Industry".

The availability of a functionally identical virtual model of the real production system enables the design phase to be conducted almost entirely under a virtual production environment. The plug-and-play nature of the virtual devices, when integrated with the functionality offered by the virtual engineering tool contributes greatly towards simplifying the work necessary to move from a pool of disconnected autonomous devices to a functional manufacturing system. This technique allows the fast prototyping the system's layout and validating the process control on the virtual manufacturing system without the timely and costly efforts required when dealing with production hardware. As the design phase reaches its end, the virtual devices can start being replaced for real ones as necessary, until the production controller is running the complete physical production cell.

The preliminary results achieved in this work have been analyzed in Mendes (2010) and Taisch et al. (2010) from a quality perspective. In spite of a formal performance analysis of the system has not been performed at this point, the real-world experiments show no visible penalties between the virtual and real worlds, or even as between the proposed approach and a more traditional one. These results have, however, presented some limitations of the current proposal. First of all, it is currently unclear how scalable is the SoA-based manufacturing system. It is expected that the decoupled nature of smart devices may allow an easy scalability, but it is currently untested how the more demanding network requirements of the SoA platform may affect a large-scale production network, since they are mitigated by a lower message ratio due to the substitution of polling for events. Moreover, smart devices are meant to be reusable, but currently they need to be completely developed from scratch. Even if this is true once for each device type, it still means an extended ramp-up time before the initial simulation is possible.

Another limitation is related to the simulation environment itself. Due to the way the Delmia Automation platform works, physical interactions that are not pre-programmed, are not taken into account during the simulation operation (e.g., a pallet does not move over a conveyor due to the rotation of the conveyor band if it is not pre-programmed to do so). This situation results in an increase of the complexity in the development of the virtual smart devices, as well as unintuitive situations in the presence of logic or simulation bugs.

In spite of the described limitations, the establishment of an engineering framework covering the complete development life-cycle of service-oriented manufacturing systems, by the integration of digital software tools, allows the achievement of important improvements, namely in terms of smooth migration, re-usability, reduction of development efforts, time and costs, and reduction of mistakes and errors of developed control solution. Especially important to this work is the reference that the migration from the virtual to the real system is done in such a way that the control models, previously analyzed and tested in the virtual environment, remain the same, being not necessary the re-engineering of the control system. Moreover, the use of Web services between the control applications and the virtual/real equipments facilitates not only the communication, but also the integration of these virtual/real scenarios. Any control approach using the service-oriented principles would trigger similar behavior in the virtual and real system. In addition, Petri nets are well known for the analysis, and if the same control approach is used in virtual environment, similar conclusions can be made for the control of the physical system. This allows testing a control logic in the simulation before applying it to the real system. These observations allow concluding, in a preliminary stage, that the control solution is smoothly migrated from the virtual environment to the real environment.

## Conclusions

The paper proposes the integration of 2D/3D digital engineering tools with a Petri nets based service-oriented framework, to establish an engineering framework for the easy design, configuration, validation, simulation, control and monitoring of modular and reconfigurable manufacturing systems. Systems can be modeled using the combination of 2D/3D objects and their layout/connection information, while the development of the control logic using Petri nets models can take place in parallel with modeling. The simulation and analysis of the final control solution can be done in a virtual environment by using the developed 2D/3D model, before its deployment into real devices. The proposed approach allows a smooth migration from the virtual environment to the real "physical" system, without any changes in the manufacturing control system configuration, representing a solid contribution for the development of reconfigurable manufacturing systems.

An experimental demonstrator was setup to prove several aspects of the proposed approach, from the methodology to the application, namely the use of Petri nets to drive the system's behavior (both virtual and real systems/devices), the easiness and less effort in the migration from the design and analysis phases to the operation and useful real-time monitoring capabilities demonstrated by the tools. Of special importance is the fact that the transition to the real system is done in such a way that the same control models (previously designed and analyzed) are used both in virtual and real systems, thus not requiring re-engineering of the control system and reaching a smoothly migration. The experimental tests proved the applicability and correctness of the proposed approach, and showed that it contributes for the reduction of design and configuration phases with the possibility of "off-line" and "online" analysis and evaluation, based on the same service layout and control logic for both virtual and physical systems.

Future work includes the development of more user-friendly engineering tools to improve the connection between the digital engineering software tools and the Petri nets engineering platforms, and to enhance an even smoothly migration from the virtual scenario to the real system. Namely, important issues are related to importing information related to the system layout and devices connections, simulating the system behavior and monitoring the real-time system operation. Additionally, special attention should be devoted to the performance analysis of the composed models or distributed system, as well the analysis of some system properties, such as scalability and reusability.

## References

Ali, S. A., Seifoddini, H., & Sun, H. (2005). Intelligent modeling and simulation of flexible assembly systems. In: *Proceedings of the 37th conference on winter simulation conference* (pp. 1350–1358).

Baccelli, F., & Canales, M. (1993). Parallel simulation of stochastic Petri nets using recurrence equations. *Transactions on Modeling and Computer Simulation, 3*(1), 20–41.

Cachapa, D., Colombo, A., Feike, M., & Bepperling, A. (2007). An approach for integrating real and virtual production automation devices applying the service-oriented architecture paradigm. In: *Proceedings of the 12th IEEE international conference on emergent technologies for factory automation* (pp. 309–314).

Gibson, N., Holland, C. P., & Light, B. (1999). Enterprise resource planning: A business approach to systems development. In: *Proceedings of the 32nd annual Hawaii international conference on system sciences, Maui, Hawaii* (pp. 785–787).

IEC 61131-3. (2003). Programmable controllers—Part 3: Programming Languages.

Jammes, F., & Smit, H. (2005a). Service-oriented paradigms in industrial automation. *IEEE Transactions on Industrial Informatics, 1*(1), 62–70.

Jammes, F., & Smit, H. (2005b). Service-oriented architectures for devices—The SIRENA view. In: *Proceedings of the 3rd IEEE international conference on industrial informatics* (pp. 140–147).

Mathes, M., Stoidner, C., Heinzl, S., & Freisleben, B. (2009). SOAP4PLC: Web services for programmable logic controllers. In: *Proceedings of the 17th Euromicro international conference on parallel, distributed, and network-based processing (PDP)* (pp. 210–219).

Melzer, I. et al. (2007). "Service-orientierte Architecturen mit Web Services", 2. Aufl., Elsevier, Spektrum Akademischer Verlag

Mendes, J.M. (2010). Engineering framework for service-oriented automation systems (Doctoral dissertation submitted to the Faculty of Engineering of the University of Porto, September 2010).

Mendes, J. M., Leitão, P., Restivo, F., & Colombo, A. W. (2010). Composition of Petri nets models in service-oriented industrial automation. In: *Proceedings of the 8th IEEE international conference on industrial informatics (INDIN'10)* (pp. 578–583).

Mendes, J. M., Bepperling, A., Pinto, J., Leitão, P., Restivo, F., & Colombo, A. W. (2009a). Software methodologies for the engineering of service-oriented industrial automation: The Continuum project. In: *Proceedings of the 33rd IEEE international conference on computer software and applications (COMPSAC'09)* (pp. 452–459).

Mendes, J. M., Restivo, F., Leitão, P., & Colombo, A. W. (2009b). Customizable service-oriented Petri net controllers. In: *Proceedings of the 35th annual conference of the IEEE industrial electronics society (IECON'09)* (pp. 4341–4346).

Mendes, J. M., Leitão, P., Colombo, A. W., & Restivo, F. (2008). Service-oriented process control using high-level Petri nets. In: *Proceedings of the 6th IEEE international conference on industrial informatics* (pp. 750–755).

Murata, T. (1989). Petri nets: Properties, analysis and applications. *IEEE, 77*, 541–580.

OASIS—Organization for the advancement of structured information standards. (2007). Web services business process execution language version 2.0, OASIS Standard

OASIS—Organization for the Advancement of Structured Information Standards. (2009). Devices profile for web services (DPWS), version 1.1, OASIS Standard

Puttonen, J., Lobov, A., & Martinez Lastra, J. L. (2008). An application of BPEL for service orchestration in an industrial environment. In: *Proceedings of the IEEE international conference on emerging technologies and factory automation (ETFA 2008)* (pp. 530–537).

Qiu, R. G., & Zhou, M. (2004). Mighty MESs: State-of-the-art and future manufacturing execution systems. *IEEE Robotics & Automation Magazine, 11*(1), 19–25.

Ramaswamy, S., & Yan, Y. (1998). Interactive modeling and simulation of virtual manufacturing assemblies: An agent-based approach. *Journal of Intelligent Manufacturing, 10*, 503–518.

Taisch, M., Colombo, A. W., Karnouskous, S., & Cannatta, A. (2010). SOCRADES roadmap, the future of SOA-based factory automation. Available at www.socrades.eu, 2010.

Tsinarakis, G. J., Valavanis, K. P., & Tsourveloudis, N. C. (2005). Modular Petri net based modeling, analysis, synthesis and performance evaluation of random topology dedicated production systems. *Journal of Intelligent Manufacturing, 6*, 79–92.

van der Aalst, W. (1998). The application of Petri nets to workflow management. *The Journal of Circuits, Systems and Computers, 8*(1), 21–66.

Zurawski, R., & Zhou, M. (1994). Petri nets and industrial applications: A tutorial. *IEEE Transactions on Industrial Electronics, 41*(6), 567–583.