# A Model-Based Service-Oriented Integration Strategy for Industrial CPS

Fang Li(✉), Ping Zhang, Hao Huang, and Guohao Chen

School of Computer Science and Engineering, South China University of Technology, Guangzhou, China
cslifang@scut.edu.cn

**Abstract.** In order to realize efficient and unambiguous development of reliable industrial CPS, we employ a model-based service-oriented integration approach, which adopts a model-centric way to automate the development course of the entire software life cycle. The structures and rules for iCPS modeling and hierarchical modeling elements are defined in the meta-model, including services, and function blocks of different abstraction level. The relationship between service and function blocks are also defined clearly in the meta-model. A UML-compliant graphical modeling environment is generated from the meta-model, with a suite of fully integrated tools. The approach is then used to develop the industrial assembly line system. It is an attempt to support iCPS design in an effective way, at the same time guarantee the system performance requirements.

**Keywords:** Industrial cyber-physical system · Model-based · Service-oriented · Integration

## 1 Introduction

The advances in information and computer technology have led to the improved integration of heterogeneous devices and systems in industrial automation. This has also led to a new type of industrialization named industry 4.0 in Germany [1, 2]. Industrialists, researchers and practitioners are focusing their eyes on the develop of a set of large process industry systems, in the form of industrial Cyber-Physical System(iCPS), in which a collection of devices are interconnected and communicating with each other by networks.

Designing and developing iCPS means to deal with several challenges for handling complexity of the system. From the functional point of view, one of the major challenges is focused, on one side, on managing the constantly increasing integrated functions and the vastly increased number of devices integrated from different manufacturers. Take the industrial assembly line as an example, not only various technologies are integrated in such system, such as, motion control, visual inspection, wireless sensing etc, but also several devices such as robotics, PLC, IPC, RFID etc. are integrated make it a large-scale control and monitoring system. On the other side, such heterogeneous systems combine mechanical components, embedded systems, networking, application software


J. Wan et al. (Eds.): Industrial IoT 2016, LNICST 173, pp. 222–230, 2016.
DOI: 10.1007/978-3-319-44350-8_22

and user interfaces to interact with each other. Crosscutting concerns from the multi-disciplinary domain make the integration of the system be a complexity task. From the non-functional point of view, rigorous performance demands are also required, for example, real-time performance and safety criticality, improved reliability and predictability, modularity, flexibility, reusability and reconfigurability, associated with design, test and verification of end products.

MBD(Model-based design) [3–6] and CBD(Component-based design) [7–10] have been proposed as efficient methods in embedded system development, which have also been widely applied in CPS development [11–13]. These approaches are effectively useful in performance guarantee and complex settlement, at the same time in allowing reuse of development efforts by components usage.

Meanwhile, SOA(Service-oriented architecture) [14–16] is being promoted based on CBD, in which, a collection of services along with an infrastructure to enable these services communicate with each other. Services are loosely-coupled and interact based on through internet, which makes SOA more flexible than CBD.

While, as the level of complexity of today's iCPS is continually increasing, problems still need to solve in large-scale complicated system design. We indicate here some key questions will need to be conducted on future industrial CPS systems, including:

1. The lack of group management for devices. As a set of individual devices are interconnected to form networks in iCPS, new ways of easily managing millions of devices in large-scale and complex systems need to be considered.
2. The lack of temporal semantics in service model. As networking technologies make predictable and reliable real-time performance difficult, synchronous and asynchronous communication in the system will be considered.
3. The function-service integration. As service is defined as coarsely-grained functional entity, an interesting question that need to be considered is how the functions will be mapped to services, for smooth integration.

We have previously proposed a model-based integration approach for CPS, including a CPS integration modeling language and a set of integrated development tools, based on IEC 61499 components [17]. As an extension of the previous work, this paper adopts model-based service-oriented integration approach in industrial CPS development. At first, the model-based service-oriented integration approach is proposed. Then, the hierarchical model definition and meta-model for service-oriented integration modeling are described. At last, a case study for an industrial assembly line system development is conducted to demonstrate the application of the proposed approach. Finally, the concluding section summarizes the features of our work and their implications.

## 2 The Model-Based Service-Oriented Integration Approach

In order to realize efficient and unambiguous development of reliable industrial CPS, we employ a model-based service-oriented integration approach, which was evolved as an extension of CPS integration framework [18]. The approach adopts a model-centric way to automate the development course of the entire software life cycle, in an iterative

and interactive way. Also, the approach integrates a set of tools to support the complete development cycle, including requirement analysis, model specification, simulation & verification, code synthesis as well as implementation.

As illustrated in Fig. 1, a UML-based meta-model is used to define the structures and rules for iCPS modeling. Hierarchical modeling elements are defined in the meta-model, including services, and function blocks of different abstraction level. The interaction and connection between components, the relationship between service and function blocks are also defined clearly in the meta-model. Then, A UML-compliant graphical modeling environment is generated from the meta-model, with a suite of fully integrated tools, i.e. tool adaptors, code generators, et al.

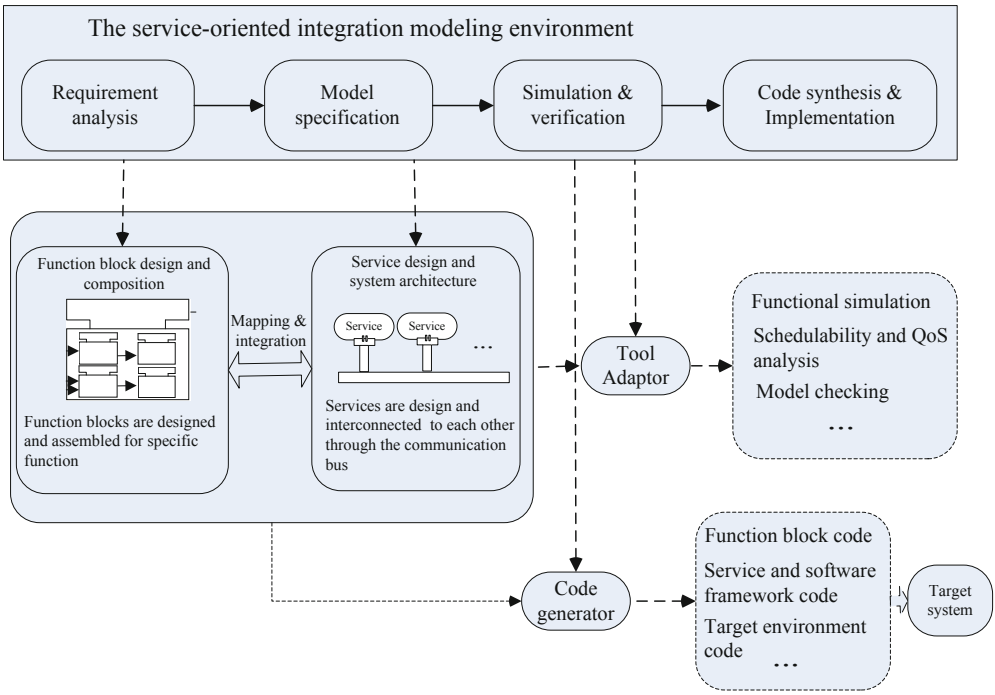


**Fig. 1.** The model-based service-oriented integration approach

In the application development process, models can be built in the service-oriented integration modeling environment after requirement analysis. It is a hierarchical component-based model with multiple-level of abstraction. In the system layer, services are designed, as a coarsely-grained functional entity that interacts with other services through a loosely-coupled communication model, for example, through a software communication bus. Then, a set of hierarchical function blocks are defined to implement the services, including BFB, CFB, SIFB and FU, partly according to IEC61499 standard. Integration can then be conducted through a mapping from function blocks to services.

Using tool adaptors in the environment, various simulation and verification toolsets (Matlab, UPPAAL et al.) can be integrated, for function and performance simulation

and verification to guarantee the function and performance requirements of the system. These tool adaptors extract specific model information and then transform it to different simulation and verification tools. For example, functional simulation, Schedulability and QoS analysis and model checking can be conducted using different tools. Also, It is an iterative process because the verification results can then feedback to tell the developer gives some modifications to model or not.

Engaging code generators, automatic code synthesis can be carried out, by inputting service-oriented integration model, and producing the platform specific configuration and the corresponding codes to build the final executable application. The object executive codes include function block code, services and software framework code and target platform code. Function block code deal with the function execution in the software, which come from pre-constructed function block library. The services and software framework code deal with network communication and connection in the system to ensure the integrity of events interaction as well as the accuracy of data interaction. Target platform code deal with the task implementation details and target environment information.

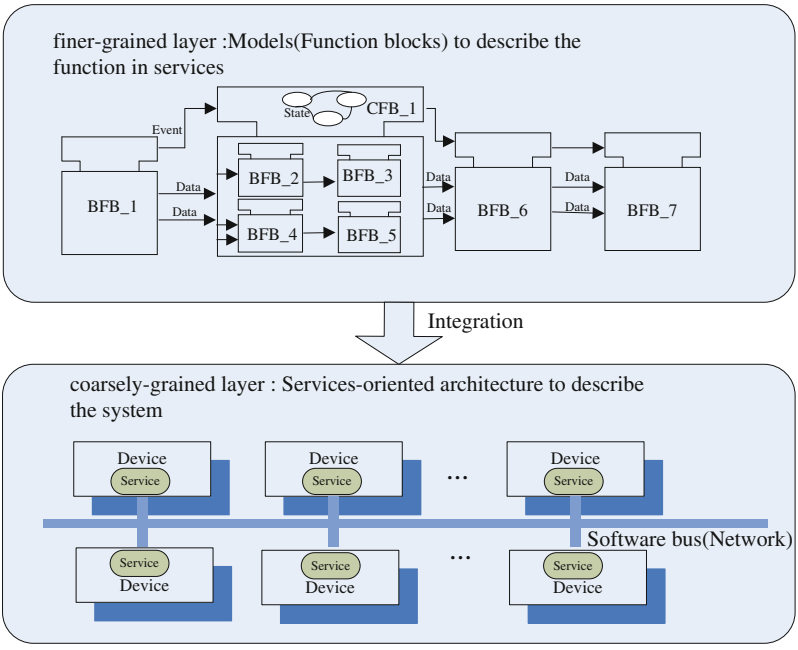


**Fig. 2.** The service-oriented integration model

## 3 The Service-Oriented Integration Model for iCPS

We propose a multi-layer integration model to specify iCPS. In the coarsely-grained layer, service oriented architecture is utilized through the application of a universal

communication framework, while, in the finer-grained layer defines various kinds of function blocks to describe the system function, as shown in Fig. 2.

### 3.1 Structural Model of the System

Structurally, a system is conceived as a network of communicating services, which interact transparently with each other by exchanging labeled messages through the communication bus. Collaborating devices are defined in system model as the providers of services.

**Service Model.** A service is defined as a well-defined, self-contained functional entity that does not depend on the context or state of other services. It is a coarsely-grained functional entity that interacts with other services through a loosely-coupled communication model. Service can be defined as a tuple

$$Service = (S_d, FU, SInf, QoS) \tag{1}$$

where, Sd is the basic description of service, including the basic information as to the service, for example, the version information, which can be described in XML. FU is the functional unit of service, which contains a set of hierarchically prefabricated function blocks to fulfill the required functionality of the service.

$$SInf = (S_i, S_o) \tag{2}$$

is a two-tuple unit to describe the interface of service, which responsible for the communication with the network. Si is the input from bus and so is the output to bus. QoS describe the non-functional characteristics of service, for example, the time constrains, etc.

**Device Mode.** Device can be defined as an independent physical entity capable of performing one or more specified functions. We also define device as supplier of services. A physical component in a larger distributed system. Device can be defined as a two tuple unit,

$$SInf = (S_i, S_o) \tag{3}$$

where, Rsc is a set of resources in a device. DInf is the physical interface of device.

**Software Bus Model.** We define software bus as a service middleware, which is responsible for the process management and communication between services. Software bus model include several modules, i.e. communication module, resource management module, data storage module, Scheduling module, etc. The service orchestration can also be organized in software bus.

**Function Unit.** Function Unit is coarse-grain reusable component to describe the function in a service. It consists of a number of hierarchical connected function blocks, including BFB, CFB, and SIFB, partly according to IEC61499 standard, which is detailed described in [18]. BFB(Basic function block) is regarded as the fundamental

function element to construct the system, which cannot be further refined. CFB(Composite function block) is a composition of BFB or other CFB to accomplish a specific complex function. SIFB (Service interface function block) is a type of service interface responsible for communication between inner function block and outer environment.

### 3.2 The Meta-model for Service-Oriented Integration Modeling

The proposed meta-model defines the system structure, components, as well as relationships and constrains, which include the concepts of embedded control, real-time as well as service-oriented domains. The UML compliant graphic modeling environment allows the modeling of devices, services, function blocks and also their relationships and characteristics. Figure 3 depicts part of the proposed meta-model for service-integration modeling.

As the core modeling elements of the system, a Service has a uniquely defined ServiceName and ServiceID, which can be distinguished from other services. SerDescription and QoS are contained in a Service, represent the basic service description and quality of service. In the QoS, time constrains of a service are defined, for example, the period, priority, WCET(the worst-case execution time) etc. FU is defined as the function unit that contained in a Service to describe the function of service. Services can be connected to software bus through the input and output ports. The Orchestrator would help in orchestrating all services involved in a system.

As a service provider, the device model include a set of resources, in which, some platform attributes are defined, for example, the location, process frequency, process_type, etc.

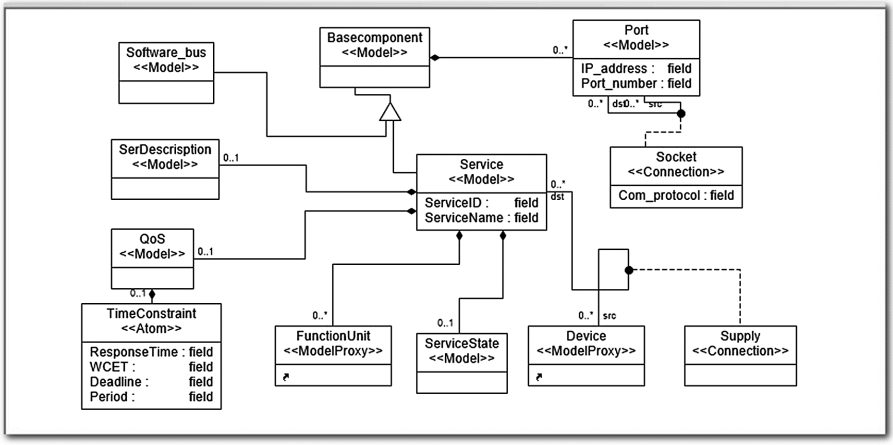


**Fig. 3.** Part of the meta-model for service-oriented integration modeling

# 4 A Case Study

## 4.1 The Assembly Line Description

To demonstrate our approach, we use the industrial assembly line development as a documented example. The industrial assembly line used in this demonstration is composed of feeding robots for automatic upper and lower material and split robot, equipped with IPC for information management and a conveyor controlled by PLC. Sample pieces in the feed box are all labeled with RFID.

In the work flow, the sample pieces are picked up to the conveyor, and information in the RFID label then can be read by sensors. The results are sent to the robot controller to command the robot to pick them from the conveyor and place them into a classified box. Devices in the system include robots, IPC, conveyor, RFID, sensors, et al. These devices are all connected by internet using TCP/IP.

## 4.2 Service-Oriented Integration Model Definition

The industrial assembly line can be modeled in the UML compliant service-oriented integration environment, as depicted in Fig. 4. A set of services are defined in the model, i.e. Feeding_service, Conveyor_service, PLC_service, etc. These services can be connected through the Software_bus. Service interface and communication protocol can also be defined in the model. Then, these service can be refined by defining a set of communication function blocks to describe the functions.

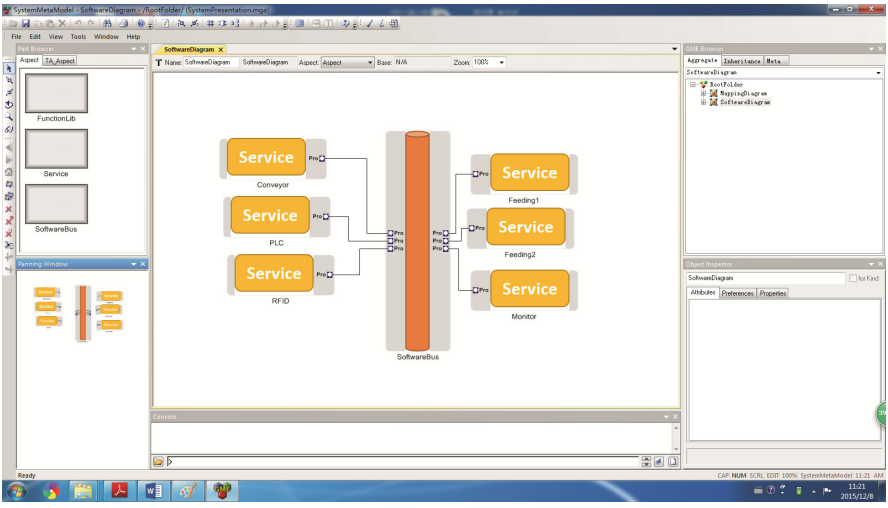


**Fig. 4.** The application model for industrial assembly line

### 4.3 The Implementation of the System

After the application model build, the verification and implementation can be conducted based on the integrated toolsets. The generated codes include function block code, services and software framework code and target platform code. Figure 5 depict the generated GUI for service management in the system.

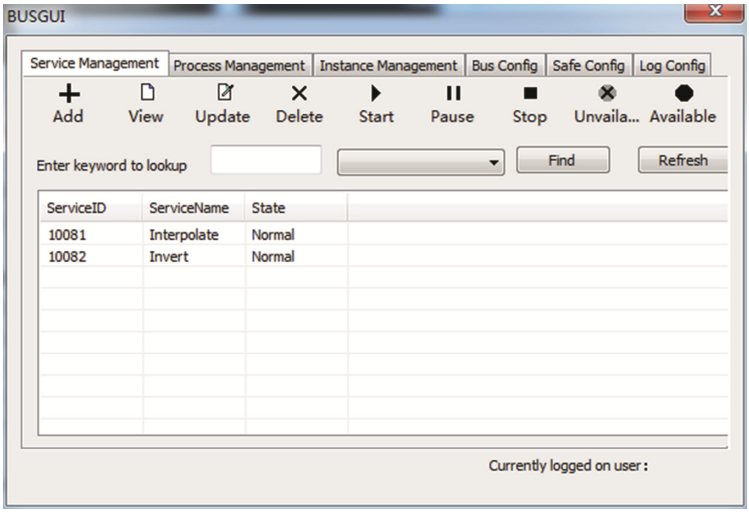


**Fig. 5.** The generated GUI for service management

## 5 Conclusion

A service-oriented integration approach for iCPS design and development is presented in the paper. It is a model-centric way to automate the development course of the entire software life cycle. By integrating a set of tools, the complete development cycle, including requirement analysis, model specification, simulation & verification, code synthesis as well as implementation can be conducted in the environment.

The structures and rules for iCPS modeling and hierarchical modeling elements are defined in the meta-model, including services, and function blocks of different abstraction level. The relationship between service and function blocks are also defined clearly in the meta-model. A UML-compliant graphical modeling environment is generated from the meta-model, with a suite of fully integrated tools. The approach is then used to develop the industrial assembly line system. It is an attempt to support iCPS design in an effective way, at the same time guarantee the system performance requirements.

**Acknowledgments.** This paper is supported by Science and Technology Planning Project of Guangdong Province, China (2014B090921007), Science and Technology Program of Guangzhou, China (20150810068), Science and Technology program of Haizhu District, China (2014-cg-02).

## References


1. Wang, S., Wan, J., Li, D., Zhang, C.: Implementing smart factory of industrie 4.0: an outlook. Int. J. Distrib. Sensor Netw. **2016**, 10 p., (2016). doi:10.1155/2016/3159805
2. Wang, S., Wan, J., Zhang, D., Li, D., Zhang, C.: Towards the smart factory for industrie 4.0: a self-organized multi-agent system assisted with big data based feedback and coordination. Comput. Netw. (2016). doi:10.1016/j.comnet.2015.12.017. Elsevier
3. Object Management Group. Model Driven Architecture - A Technical Perspective [EB/OL], 9 July 2001. http://www.omg.org/mda/
4. Ptolemy Project. University of California Berkeley [EB/OL]. http://ptolemy.eecs.berkeley.edu
5. Vyatkin, V., Dubinin, V.: Sequential axiomatic model for execution of basic function blocks in IEC 61499. In: 5th IEEE International Conference on Industrial Informatics (INDIN), Vienna, 2007, pp. 1183–1188 (2007)
6. Porter, J., Lattmann, Z., Hemingway, G., Mahadevan, N., Neema, S.: The ESMoL modeling language and tools for synthesizing and simulating real-time embedded systems. In: 15th IEEE Real-Time and Embedded Technology and Applications Symposium 2009, pp. 112–130 (2009)
7. International Electro-technical Commission (IEC). International Standard IEC 61499, Function Blocks, Part 1–Part 4[S]. IEC Jan 2005. http://www.iec.ch/
8. Function Blocks Development Kit. Holobloc Inc [EB/OL], July 2005. http://www.holobloc.com
9. CORFU Project. University of Patras [EB/OL]. http://seg.ee.upatras.gr/corfu/dev/index.htm
10. Lee, E., Neuendorffer, S., Wirthlin, M.: Actor-oriented design of embedded hardware and software systems. J. Circuits Syst. Comput. **12**(3), 231–260 (2003)
11. Chen, M., Wan, J., Li, F.: Machine-to-machine communications: architectures, standards, and applications. KSII Trans. Internet Inf. Syst. **6**(2), 480–497 (2012)
12. Zou, C., Wan, J., Chen, M., Li, D.: Simulation modeling of cyber-physical systems exemplified by unmanned vehicles with WSNs navigation. In: Proceedings of the 7th International Conference on Embedded and Multimedia Computing Technology and Service, Gwangju, Korea, pp. 269–275, September 2012
13. Wan, J., Yan, H., Liu, Q., Zhou, K., Lu, R., Li, D.: Enabling cyber-physical systems with machine-to-machine technologies. Int. J. Ad Hoc Ubiquitous Comput. **13**(3/4), 187–196 (2013)
14. Cucinotta, T., et al.: A real-time service-oriented architecture for industrial automation. IEEE Trans. Indus. Inf. **5**(3), 267–277 (2009)
15. Spiess, P., Karnouskos, S., Guinard, D., Savio, D., Baecker, O., Sa de Souza, L.M., Trifa, V.: SOA-based integration of the internet of things in enterprise services (2009). http://www.socrades.eu/Documents/objects/file1259604734.51
16. Troger, P., Rasche, A.: SOA meets robots - a service-based software infrastructure for remote laboratories. Int. J. Online Eng. **4**(2), 24–30 (2008)
17. Li, D., Li, F., Huang, X.: A model based integration framework for computer numerical control system development. Robot. Comput. Integr. Manuf. **26**(4), 333–343 (2010)
18. Li, F., Wan, J., Zhang, P., Li, D.: A multi-view integration language for cyber-physical robotic system. In: ICMLC 2013, pp. 387–392 (2013)