Middleware to Support Cyber-Physical Systems

Nader Mohamed[1], Jameela Al-Jaroodi[2], Sanja Lazarova-Molnar[3], and Imad Jawhar[4]

[1]Middleware Technologies Lab., Bahrain
[2]Department of Engineering, Robert Morris University, Pennsylvania, USA
[3]Center for Energy Informatics, University of Southern Denmark, Denmark
[4]College of Information Technology, UAE University, UAE
nader@middleware-tech.net aljaroodi@rmu.edu, slmo@mmmi.sdu.dk, ijawhar@uaeu.ac.ae

***Abstract*—middleware can provide novel and practical approaches for enhancing Cyber-Physical Systems (CPS) application development processes and operations. This paper discusses the roles and advantages of using middleware to build CPS. In addition, the paper studies the required features needed in such middleware for development and operation of CPS.**

Keywords - cyber-physical systems, middleware

I. INTRODUCTION

There are many CPS applications that add enhancements and smart features to several types of physical systems and environments [1]. CPS attach different hardware components like sensors, actuators, microcontrollers, and other devices to physical systems or environments and use distributed software that implements smart algorithms to control the corresponding physical system. CPS can add smart mechanisms to fully automate manufacturing processes, manage and enhance the operations and safety of environments and infrastructures, enhance energy consumption in smart buildings, improve healthcare for patients, and many more. However, the development of such complex systems composed of many distributed and heterogeneous components interacting in various ways and capabilities is extremely difficult.

Middleware platforms can provide novel approaches for enabling the development and operation of CPS applications. While existing middleware platforms such as real-time CORBA platforms has shown promise as a platform for general distributed systems with some constraints, they lack the flexibility in dealing with the CPS challenges. This paper summarizes our initial investigation of the potential for middleware to support CPS applications. With this regard, we identify the common CPS applications challenges and the needed middleware support to address these challenges.

II. CPS APPLICATIONS CHALLANGES

We realize based on studying different CPS applications [2-14], there are common challenges facing developing these applications. The main challenges are:

1. Real-Time operations: Most CPS applications need to function in real-time to deliver usable information. This includes real-time sensing, communication, processing, decision making, and actions.
2. Heterogeneous Devices: CPS applications are built with multiple heterogeneous devices like sensors, actuators, microcontrollers, and communication devices. In addition, they operate in heterogeneous physical environments.
3. Limited Capability Devices: Some CPS need to use devices with limited capabilities and remote functions. This is mainly due to the current limitations on available devices or to reduce the cost of the CPS. These devices usually have limited communication, processing, and storage capabilities.
4. Distributed Processing: CPS applications require distributed processing and decision making for their operations. In addition, some applications need to use parallel processing for fast and timely decision making.
5. Security and Privacy: As most CPS support distributed critical applications, there are high security and privacy concerns. Therefore, the security and privacy of the information and software must be protected.
6. Reliability and Fault Tolerance: Many CPS applications are critical applications; therefore, they need to be reliable and highly available. They should be able to detect and resolve different types of faults without negatively impacting the physical systems or environments.
7. Special Communication: Special communication requirements are needed by some CPS applications among the devices and subsystems used. These may include real-time support, reliable communication, and efficient information collection and exchange. This will require supporting optimized communication mechanisms.
8. Mobility: Some CPS applications involve mobile devices that need to be efficiently and securely connected with the rest of the system. This may require actively managing the mobile devices as they change locations.
9. CPS Devices Locations: CPS applications that involve mobile devices, need to be aware of the mobile devices' locations to optimize their operations and correctly achieve the application's objectives.
10. Power Limitations: Some CPS components are used in remote locations where no constant power sources are available. Thus, the CPS design should focus on extending the life of the system components using efficient and power-aware software, hardware, and communication.
11. Integration with Other Systems: Some CPS applications require integration with other systems or other CPS. For example, integrating CPS applications in neighboring vehicles to enable collaborative safety applications or integrating CPS with a powerful cloud that offers processing power, storage, and data services.
12. Intelligent Decisions: Most CPS applications need to make intelligent decision to optimize their operations. This may

978-1-5090-5252-3/16/$31.00 ©2016 IEEE

involve including intelligent algorithms such as data mining algorithms in the CPS design.

13. Context Awareness: Some CPS cannot properly or efficiently function without knowing the context of their systems resources, physical environment and general domain. This will require the knowledge of specific information such as system status, and other external physical contexts utilized for the operations of CPS.

All the above mentioned challenges make developing CPS applications without using a suitable middleware framework very difficult.

III. Middleware to support CPS Applications

Middleware has become a necessary part of any distributed system such as wireless sensor networks (WSN) and cloud computing. It is practically impossible to develop large-scale or complex distributed applications without involving middleware [15]. Similarly, middleware can provide many advantages for CPS. However, one of the main differences between middleware for CPS and other distributed systems is that it should be designed and optimized to provide efficient solutions to the specific challenges facing CPS applications. Although WSN have some common relations with CPS, CPS have more heterogeneous devices and require closed loop and efficient controls which will raise many technical challenges that require new middleware frameworks specifically designed for CPS.

CPS Middleware frameworks can provide important support for developing and operating applications (see Table 1). The availability of a suitable middleware platform that provides solutions for the challenges discussed earlier will not only enhance the CPS development process, but also enhance reusability and maintainability, and reduce risks and overall implementation cost. With a suitable middleware platform designed to support CPS applications, the developer can focus on developing the main functions of the CPS application rather than expending huge time and effort in implementing solutions and codes to solve the general and common CPS issues. The specialized middleware for CPS can be designed to include a set of services that provide solutions for various common CPS challenges. These services can be used to implement different CPS applications. The developers can use middleware APIs for these services to utilize their functions and features and integrate them with the required specific functions of the CPS. In addition, the middleware can enable the reuse of any previously implemented module in new CPS applications. This will also reduce the time and efforts needed to test new modules as the reusable modules have already been tested and approved for use. The middleware will also enable maintainability by allowing easy changes for any modules in the system. Any software module can be changed or replaced by better one if it uses the same interface. This will also enable easy changes of the hardware components in the CPS environment to upgrade or enhance them.

We classify middleware systems that can support CPS applications into three types. These three types are defined based on their abstraction levels and their support functions. Each of these types can support different programming models to develop and operate CPS applications. Here we discuss the main characteristics and functions of the three types:

A. Communication Middleware for CPS Applications

This type of middleware mainly enables and facilitates communication among heterogeneous and limited resources CPS components. This includes offering services for efficient unicast communication and efficient group communication for broadcasting, multicasting, and data collection. It can provide basic security mechanisms to be used by developers to protect the communications among CPS components. It also provides basic mechanisms to communicate with other systems through message passing and/or remote procedure calls. While this type of middleware enables communication among CPS components, the developers need to solve and write codes for many other challenges mainly due to its limited scope and functions. The developers also need to handle the details of distributed processing in CPS. One possible approach for a communication middleware is to use a customized massing passing model which can meet the CPS challenges.

B. Value-Added Services Middleware for CPS:

In this type, a number of services can be provided and used as a part of the CPS applications adding features and value to these applications. These services can be real-time support, monitoring, validation, reliability, fault tolerance, mobility support, location-based support, security attacks detection services, and service-level integration with other systems. A higher level of programming models for distributed and parallel processing can be used with this middleware to allow developers to implement CPS applications using the available services. This middleware includes a resource manager and scheduler to enable implementing the services and also to allow developers to define available resources and detailed policies guiding the use of the available services. One of the most suitable middleware architectures to use here is the service-oriented middleware [16].

C. Advanced Services Middleware for CPS:

Here advanced services and smart support components in addition to the services available in the Value-Added Services Middleware can be offered. These include autonomous resource discovery and management, context awareness support, multiple CPS collaboration support, and intelligent decision support. In addition, smart services such self-adaptive, self-resilient, and self-protected services can be offered. The developers in this case will resort to advanced high-level abstracted programming models to write the CPS applications instead of conventional programming languages. They can also specify high-level policies and global objectives for the whole system to operate on.

These three middleware types differ in the level of abstraction and how much time and effort is needed by the developers to implement new CPS applications. The developers need more effort to develop new applications with the communication middleware alone as they need to deal with individual components in the CPS, while they need less time and effort with the advanced services middleware as they will use high-level policies and global objectives and the middleware will map these into implemented services. However, the third type of middleware is very difficult to implement as it needs to self-handle most of the CPS common challenges.

TABLE 1. DIFFERENT MIDDLEWARE SUPPORT FEATURESS NEEDED BY DIFFERENT CPS APPLICATIONS IN THE GENERAL CASES.

	Real-Time Support	Heterogeneously Support	Limited Capabilities Support	Distributed Processing Support	Security and Privacy Support	Reliability & Fault Tolerance Support	Special Communication	Mobility Support	Location-Based Support	Power Efficiency	Integration with Other Systems	Intelligent Decision	Context Awareness
Medical Cyber-Physical Systems [2]	✓	✓	✓		✓	✓	✓	✓		✓	✓	✓	✓
Smart Buildings [3]	✓	✓			✓	✓					✓	✓	✓
Smart Grid [4]	✓	✓		✓	✓	✓	✓				✓	✓	✓
Gas & Oil Pipelines Monitoring & Control [5]	✓	✓	✓	✓	✓	✓	✓			✓	✓		✓
Smart Water Networks [6]	✓	✓	✓	✓	✓	✓	✓			✓	✓	✓	✓
Vehicular Safety Applications [7]	✓	✓			✓	✓		✓	✓			✓	✓
Collaborative Vehicular Safety Applications [8]	✓	✓		✓	✓	✓	✓	✓	✓		✓	✓	✓
Self-Driving Car [9]	✓	✓		✓	✓	✓		✓	✓		✓	✓	✓
Manufacturing Control and Monitoring [10]	✓	✓			✓	✓					✓		✓
Unmanned Autonomous Vehicle (UAV) [11]	✓	✓	✓		✓	✓		✓	✓	✓		✓	✓
Collaborative UAVs [12]	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Wind and Hydro Power Plants [13]	✓	✓		✓	✓	✓					✓		✓
Greenhouse Efficient Control [14]		✓	✓		✓	✓				✓		✓	✓

There is some ongoing research to customize existing middleware platforms or design new middleware services to fit the CPS environments and address their applications challenges. Eexamples of these efforts are the different approaches to solve real-time issues in distributed CPS [17] and [18]. The aforementioned efforts provide solutions to specific issues in CPS rather than addressing the generic model that can support various features and apply to different CPS applications. More work is needed to address the general issues facing most, if not all, CPS applications and offer middleware platforms that can be adapted and used for several applications.

IV. CONCLUSION

The complex nature of CPS applications imposes many challenges involving the implementation, communication, security, heterogeneity and several other aspects of CPS applications. This calls for serious efforts to address these challenges effectively to facilitate better implementation, deployment and operations of CPS applications. In this paper, we propose the use of middleware frameworks to support these operations. Middleware has proven its worth and effectiveness in many other domains and can offer the same benefits for CPS applications. We also described three types of middleware frameworks that can be used. These are communication middleware, value-added middleware and advanced services middleware. Each type offers a different level of abstractions and features, which offers flexibility in how the CPS applications are developed. One of our prospects is to further investigate the three middleware types and define the specific features, capabilities and API needed to use them to build and operate CPS applications effectively.

REFERENCES

[1] Al-Jaroodi, J., et al. "Software Engineering Issues for Cyber-Physical Systems," In IEEE SMARTCOMP, 2016.

[2] Lee, I., et al. "Challenges and research directions in medical cyber–physical systems," In Proc. of the IEEE, 100(1), pp.75-90, 2012.

[3] Gurgen, L., et al. "Self-aware cyber-physical systems and applications in smart buildings and cities," In 2013 DATE, pp. 1149-1154, 2013.

[4] Karnouskos, S. "Cyber-physical systems in the smartgrid," In 9th IEEE INDIN, pp. 20-23, 2011.

[5] Ali, S., et al. "Network Challenges for Cyber Physical Systems with Tiny Wireless Devices: A Case Study on Reliable Pipeline Condition Monitoring," *Sensors*, *15*(4), pp.7172-7205, 2015.

[6] Kartakis, S., et al. "WaterBox: A Testbed for Monitoring and Controlling Smart Water Networks," In Proc. of the 1st ACM CySWater, 2015.

[7] Al Falasi, H. and Mohamed, N., "Similarity-Based Trust Management System for Detecting Fake Safety Messages in VANETs," In 2nd IoV, pp. 273-284, Springer, 2015.

[8] Fallah, Y.P., et al. "Design of cooperative vehicle safety systems based on tight coupling of communication, computing and physical vehicle dynamics," In 1st ACM/IEEE ICCPS'10, pp. 159-167, 2010.

[9] Berger, C. and Rumpe, B. "Autonomous Driving - 5 Years after the Urban Challenge: The Anticipatory Vehicle as a Cyber-Physical System," Proc. of the INFORMATIK, pp. 789–798, 2012.

[10] Lee, J., et al. "A cyber-physical systems architecture for industry 4.0-based manufacturing systems," *Manufacturing Letters*, *3*, 2015.

[11] Klesh, A.T., et al. "Cyber-physical challenges for space systems," In 3rd ACM/IEEE ICCPS, pp. 45-52, IEEE, 2012.

[12] Mohamed, N., et al. "A service-oriented middleware for building collaborative UAVs," Journal of Intelligent & Robotic Systems, 74(1-2), pp.309-321, 2014.

[13] Sridhar, S., et al. "Cyber–physical system security for the electric power grid," Proceedings of the IEEE, 100(1), pp.210-224, 2012.

[14] Gonda, L. and Cugnasca, C.E., "A proposal of greenhouse control using wireless sensor networks," In Computers in Agriculture and Natural Resources, p. 229, 2006.

[15] Al-Jaroodi, J. and Mohamed, N., "Middleware is STILL everywhere!!!.," In CCPE, 24(16), pp.1919-1926, 2012.

[16] Mohamed, N. and Al-Jaroodi, J. "Service-oriented middleware approaches for wireless sensor networks," In 44th HICSS, 2011.

[17] Zhang, Y., et al. "Reconfigurable real-time middleware for distributed cyber-physical systems with aperiodic events," In ICDCS'08, pp. 581-588, IEEE, 2008.

[18] Lin, K.J. and Panahi, M., "A real-time service-oriented framework to support sustainable cyber-physical systems," In 8th IEEE INDIN, pp. 15-21, IEEE, 2010.