Quo Vadis Cyber-Physical Systems

Research Areas of Cyber-Physical Ecosystems

A Position Paper

Christian Bartelt
Clausthal University of Technology
Department of Informatics
38678 Clausthal-Zellerfeld, Germany
christian.bartelt@tu-clausthal.de

Andreas Rausch
Clausthal University of Technology
Department of Informatics
38678 Clausthal-Zellerfeld, Germany
andreas.rausch@tu-clausthal.de

Karina Rehfeldt
Clausthal University of Technology
Department of Informatics
38678 Clausthal-Zellerfeld, Germany
karina.rehfeldt@tu-clausthal.de

ABSTRACT

Many *technological innovations* from the research area of dynamic adaptive systems or IT ecosystems are already established in current software systems. Especially cyber-physical systems should benefit by this progress to provide smart applications in ambient environments of private and industrial space. But a proper and *methodical engineering* of cyber-physical ecosystems (CPES) is still an open and important issue. Traditional software and systems engineering facilities (system models, description languages, or process models) do not consider fundamental characteristics of these ecosystems as openness, uncertainty, or emergent constitution at runtime sufficiently. But especially these aspects let blur the line of system boundaries at design time. The diverse components of CPES have essential impacts on the engineering of CPES as well, concerning time synchronizing, execution control, and interaction structure. Self-balanced control in CPES promises new application possibilities, but also needs new engineering techniques concerning the overall engineering process, including requirements engineering and runtime verification. In this position paper we survey and summarize the dimensions of challenges in applying control theory for the engineering of cyber-physical ecosystems.

Categories and Subject Descriptors

D.2.11 [**Software Engineering**]: Software Architecture

General Terms

Design, Reliability, Security, Languages, Theory

Keywords

Cyber-physical systems, software ecosystems, systems engineering, control theory, self-balanced control, system-of-systems

1. INTRODUCTION

The term cyber-physical system (CPS) is widely used nowadays. One of the most common definition of cyber-physical systems is a system which connects physical and virtual processes, where bidirectional information flows are significant[9]. The physical processes are controlled and monitored by computations. Traditionally, this kind of systems is engineered based on the automation system pyramid by the Totally Integrated Automation concept[13]. This classic architecture spans from field level to management level by a hierarchical control scheme. The automation system pyramid is inflexible and not suited for adaptive systems. Adaptivity in this context means, that a system is able to change its structure autonomously to adapt to a new environment or certain situation. By decomposing each layer of the automation system pyramid into components and allowing communication between arbitrary components, CPS become more flexible. As a result, the engineering of CPS deals with several challenges regarding the self-organization and adaptivity.

Openness yields benefits for CPS but also requires new research. In this case, openness refers to a system with high interoperability and open interfaces. We introduce the term cyber-physical ecosystems (CPES) for adaptive and open cyber-physical systems. These systems combine cyber-physical systems with an approach for interacting with other systems. The control of such highly dynamic systems have been researched in the field of software ecosystems for several years[6, 11, 12]. But the transfer of self-balanced control mechanisms to realize CPES raises new ambitious challenges in engineering. We aim to name some of these challenges in this position paper. To do so, we first inspect CPS and software ecosystems in three different engineering challenges areas: the overall approach, the design of components and the runtime operation. The overall approach for designing and running systems covers the whole development cycle as well as the runtime environment. The design of components concentrates on design time while the last area covers the runtime of systems. CPES combine concepts from CPS and software ecosystems. Therefore, we derive challenges for CPES by combining challenges for both system classes.

2. STATE OF THE ART

By allowing more flexibility in the architecture of CPS, new CPS applications will become possible. In this section, we first present engineering challenges of CPS and afterwards of software ecosystems.

2.1 Cyber-Physical Systems

A cyber-physical system is a system-of-systems. In a CPS, information systems cooperate with control systems. Information systems are socio-technical systems for information processing whereas control systems control physical processes. By cooperating, the control systems compensate the missing sensors and actuators of information systems. In turn, the information systems provide data analysis and storage for the control systems.

CTSE'15, August 31, 2015, Bergamo, Italy
ACM. 978-1-4503-3814-1/15/08...$15.00
http://dx.doi.org/10.1145/2804337.2804341

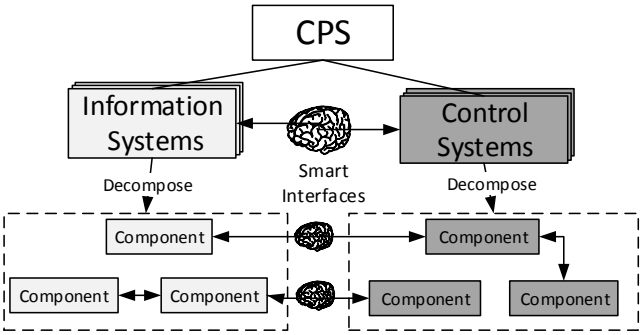

Figure 1: CPS (decomposed) with smart interfaces

There has to be an intelligent interface between the two kinds of systems (cf. Figure 1). On the one hand, control systems are often continuous and realtime-capable and interact with the physical world. On the other hand, information systems are discrete and have no natural concept of time but can operate on huge amounts of data. These different properties can only be joined by an intelligent interface.The special nature of CPS is adaptivity. Adaptivity in CPS is achieved by decomposing each information and control system in components and allowing data exchange between arbitrary components (cf. Figure 1). The interface between the components deals with the same different characteristics as before. But it can adapt to certain situations by composing the components in a different way. Those kinds of CPS will pose new challenges to science and research[2]:

How can an overall development and operation approach be established during the whole life-cycle regarding adaptivity of the systems, learning of functions, self-organization and more?

CPS consist of networked information systems and control systems. To manage complexity, information systems and control systems are decomposed in modular building blocks called components. Consequently, the intelligent interface between the information systems and the control systems are also decomposed to fine grained smart interfaces between the components of the information systems and the components of the control systems. Those local and small interfaces are smart enough to support adaptivity, learning of functions and self-organization.

How should design and development methods look like to consistently expand the concepts of system engineering in such a way that it can also be used for cyber-physical systems?

Heterogeneously networked structures like CPS require an integral systemic view and interdisciplinary cooperation between mechanical engineering, electrical engineering and computer science. Therefore an interdisciplinary multi-view and multi-level supporting modelling and development approach is required. It is necessary to prepare discipline-specific approaches for integration into CPS. Thus handling complexity and the realization of new functionalities through the adaptivity of the systems and the combination of functions will be at the forefront.

How should cyber-physical systems - that are more adaptive and open systems but still have high dependability requirements to be guaranteed - be deployed, operated, monitored and maintained?

CPS directly influence the physical world. Therefore an operation approach is required that is able to control and thereby prevent damage of the CPS and even more important its environment. The damage should not only be prevented in case of errors and failures but particularly in case of incorrect behavior and adaption of the CPS.

There are already different modelling approaches for CPS. Some of the challenges of modeling CPS are summarized in [4]. The work also lists some promising new approaches. The close integration of embedded systems and the physical world is often modelled with hybrid systems. In [8] hybrid automata are used for verification of cyber-physical systems. There are also approaches for component-based design of hybrid systems as well as methods for checking whether a hybrid systems satisfies a specification[3, 5].

2.2 Software Ecosystems

Complex software ecosystems are complex, compound system consisting of interacting individual adaptive systems, which are adaptive as a whole, based on engineered adaptability. The different life cycles of the individual adaptive systems must also be taken into consideration [12].

A complex software ecosystem (CSE) comprises of individual adaptive systems whose behavior and interactions change over time. These changes are usually not planned centrally, but arise from independent processes and decisions within and outside the CSE. For example, a slippery road warning system of a car requests for information about road conditions without any knowledge about relevant information providers in the ecosystem. The warning system could connect to totally different systems, like a weather forecast provider as well as an electronic accessible plan of the snow plowing service or even the ABS monitoring of cars in same area, to receive the relevant information.

In addition, CSE are mixed human-machine artifacts: human beings in the complex software ecosystem interact with the individual systems, and in this way they become an integral, active part of the CSE. Therefore, human requirements, goals, and behavior must be considered when designing a CSE, by modeling them as active system components. A number of ambitious challenges follow from the mentioned characteristics of software ecosystems. These can be divided in three areas – balancing evolution process between system (local) and ecosystem (global), distributed and independent engineering at design time, and technology support for system composition at runtime.

How can system development and system operation be integrated into a close linked and balanced approach for software ecosystem evolution?

The development and evolution of systems in software ecosystems is characterized by interplay of classical model-based engineering and constraint-based development to consider external restrictions from the ecosystem. As already mentioned, a CSE consists of a set of individual adaptive systems. To restrict the individual adaptive behavior of the adaptive systems, local constraints might be added and enforced locally under closed world assumption. In addition, institutional and improvement constraints from an ecosystem's communities or covering common ecosystem's objectives and guidelines are added—following an open-world semantic. All of these constraints can be used to validate the individual models of the adaptive systems but also to validate the union of models of all adaptive systems resp. the ecosystem's models. Therefore constraint validations of individual systems, but also of the interaction between these adaptive systems, can be identified during design time. However, as changes in these systems are not planned centrally, but arise from independent processes and decisions, adaptivity cannot be completely controlled during design time. Consequently, we also

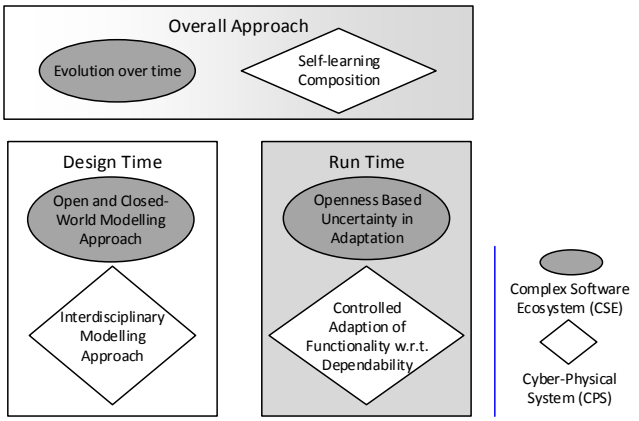

Figure 2: Combined challenges in engineering of CPES

have to take the runtime into account. Therefore, the constraints provide a knowledge transfer between design time and run time. Constraints are additionally monitored and enforced during run time.

How should distributed engineering of the constituent systems with different life-cycles and not centrally managed and coordinated be organized during design time?

As consequence of openness of software ecosystems as well as uncertainty of their system environments, and the independency and distribution of development processes follow several challenges. For example, established design methods using software modeling under the closed-world assumption (CWA) with description techniques based on the open-world assumption (OWA) have to be combined. The application of OWA during design is necessary because the demand for the specification of system parts while only incomplete information about the whole system at runtime is available[11]. Especially software interfaces of independently developed components have to be connected semantically dependable. Because of the distributed development of system parts by independent actors a syntactic connection of interfaces is undependable in general[7]. Furthermore, adaptability demands have to be engineered at design time.

How can self-adaptation of the software ecosystem evolution be balanced during runtime?

The systems composition in CSE moves from design to runtime. This is a profound changed paradigm which requires answers to several challenges. One central question is, how can technology support the dynamical adaptivity resp. the emergent composition of systems to facilitate a self-balanced ecosystem at runtime? The adaptability of a software ecosystems needs a common technological infrastructure to manage the joining components and their connections as well as additional services [10]. The technology has to provide an autonomous mechanism which can balance the concurrent needs of systems in a CSE. Systems of a software ecosystem compete for common shared resources. For the balancing of the competitive demands of systems in software ecosystems an appropriate mechanism has to be implemented by the infrastructure [1]. Furthermore the semantic correctness of the autonomous system integration and system requirements has to be validated during operation and design time.

3. RESEARCH AREAS IN ENGINEERING OF CPES

CSE are complex adaptive systems of adaptive systems and human beings. Thereby software ecosystems are open systems with respect to the independent evolution of the constituent adaptive systems, the dynamic self-adaption mechanisms of the constituent adaptive systems themselves, and the active human beings within the CSE. Hence software ecosystems come with a high degree of uncertainty due to their openness and adaptivity. To manage uncertainty additional knowledge is elaborated and used during design time and run time. In addition an overall evolutionary development approach is provided by software ecosystems.

CPS merge together embedded software-intensive control systems and global networked internet-based information systems by modular, small, intelligent, and non-hierarchical interfaces between the components of the control system and the information system. Adaptivity in CPS is based on functional adaption of the control system with respect to high dependability issues. Interdisciplinary and holistic engineering approaches are applied to provide new functionalities through an intelligent new connection between existing control components and information components.

In CPES we integrate the openness and adaptivity of software ecosystems with the modular and intelligent component coupling of CPS. Thereby new, combined research areas arise. We deduce these research areas from the combination of the challenges in Section 2. The research areas cover A) the overall approach and evolution of the system, B) the design time development approach as well as C) the operation approach during runtime of the system (cf. Figure 2).

A) How should the controlled evolutionary development approach (CSE) be combined with the long-term self-adaption mechanism by the intelligent interfaces (CPS)?

In classical software engineering, the design starts with main requirements. These requirements are step-wise refined to hierarchical requirements. After design time the system is not intended to change. A new feature for the system will be a new project. This classical approach is not applicable for CPES.

In CPES integration and validation is done during runtime. Moreover the system is growing and changing dynamically over time. The system itself adapts to its environment. Once the system is no longer able to adapt itself, new data for machine learning based long-term self-optimization are provided by smart intelligent sensor data to fulfill the changing requirements and expectations after self-optimization. In addition new requirements might be derived for the system itself but also for components and physical processes realizing their own specific functionalities. These requirements may contradict each other – for example two machines with common resources which both aims at 100% occupancy rate. Therefore, global requirements exist as well. Each time a new component joins the system it has to be adjusted to fit to the global requirements.

Ensuing from this situation, technical challenges can be derived: The components and the host infrastructure of CPES need an advanced configuration service compared to CSE. This (distributed) service has to able to receive and value feedback from the physical environment. Further it has to balance competitive requirements (possibly by market mechanisms) and has to consider appropriate migration strategies.

B) How can an open and closed world modeling approach (CSE) be integrated with an interdisciplinary modeling approach (CPS)?

Traditional software development approaches offer various techniques to support software engineers. One of the most

fundamental is the use of models and modeling. Depending on what is considered relevant to a system under development at any given point, various modeling concepts and notations may be used to highlight one or more particular perspectives or views of that system. It is often necessary to convert between different views at an equivalent level of abstraction facilitated by model transformation, e.g. between a structural view and a behavioral view.

CPES combine the virtual world represented by information systems and the physical world influenced by control systems. Consequently our modeling approach must be able to represent the various engineering disciplines for software to electrical up to mechanical engineering. Furthermore, as CPES are evolving in a not centrally planned and managed manner we have to use open and closed world models combining models describing the system under construction as well as overall ecosystem constraints that have to be guaranteed by all parts of the CPES in an interdisciplinary integrated approach.

The model-based design methods in the well-established engineering disciplines of physical systems (electrical/mechanical engineering) assume traditionally a closed world paradigm. For a consideration of openness in CPES, these system description languages have to be enhance by a support of not only inter-disciplinary modeling but also by additional expressions which allow to specify semantics based on the open-world assumption.

C) How can the openness based uncertainty in system adaption (CSE) be balanced with the need to control system adaption with respect to dependability issues (CPS)?

After a system has been designed, verified and developed, it will to be deployed and executed within the ecosystem. In order to be useful over time, systems must be able to adapt themselves to changing needs, goals, requirements or environmental conditions as autonomously as possible. Three levels of adaptability, namely: engineered adaptability, emergent adaptability and evolutionary adaptability have to be supported. During runtime, various aspects have to be considered in order to enable and control those kinds of adaptability. Therefore the MAPE-K loop is a typical well-known architectural blueprint for such a system.

As CPES manipulate and influence the physical world we have to guarantee the CPES does not damage or hurt its environment. CPS often realize safety-critical applications. Therefore, they underlie strict requirements like safety, security, privacy or realtime controlling. In classical applications the system behavior can be assured at design time. For CPES the existing approaches have to be enhanced to detect all possible dependability problems in advance and reorganize the system to prevent the environment from possible damage. Prediction techniques have to be elaborated to guide and guarantee the required dependability issues of the CPES during run time. Moreover the dependability issues might change over time and thus the guarantees have to be adapted dynamically during run time. To ensure the validity of dependability/safety conditions, the technical platform of CPES has to provide mechanisms to predict physical effects (simulation/test techniques). This is necessary for an estimation and valuation of the future behavior of systems before possibly irreversible or safety-critical effects are implemented in the physical environment. Considering that the available configuration mechanisms of CSE has to be enhance to ensure a safe operation of autonomous composed CPS.

4. CONCLUSION

In this paper we presented a new point of view on CPS. CPS will be more open and complex in the future. This will require transfer of self-balanced control mechanisms to CPES. We presented the current state of the art in both CSE and CPS. These different works may be considered when facing the challenges of CPES we presented.

5. REFERENCES

[1] Bartelt, C., Fischer, B. and Rausch, A. 2013. Towards a Decentralized Middleware for Composition of Resource-Limited Components to Realize Distributed Applications.

[2] Cyber-Physical Systems. Driving force for innovation in mobility, health, energy and production: http://www.acatech.de/de/publikationen/publikationssuche/detail/artikel/cyber-physical-systems-innovationsmotor-fuer-mobilitaet-gesundheit-energie-und-produktion.html. Accessed: 2015-06-15.

[3] Damm, W., Möhlmann, E. and Rakow, A. 2014. Component Based Design of Hybrid Systems: A Case Study on Concurrency and Coupling. Proceedings of the 17th International Conference on Hybrid Systems: Computation and Control (New York, NY, USA, 2014), 145–150.

[4] Derler, P., Lee, E.A. and Sangiovanni-vincentelli, A.L. 2011. Addressing Modeling Challenges in Cyber-Physical Systems.

[5] Henzinger, T.A. and Otop, J. 2014. Model Measuring for Hybrid Systems. Proceedings of the 17th International Conference on Hybrid Systems: Computation and Control (New York, NY, USA, 2014), 213–222.

[6] Herold, S., Klus, H., Niebuhr, D. and Rausch, A. 2008. Engineering of IT Ecosystems: Design of Ultra-large-scale Software-intensive Systems. Proceedings of the 2Nd International Workshop on Ultra-large-scale Software-intensive Systems (New York, NY, USA, 2008), 49–52.

[7] Kolatzki, S., Goltz, U., Hagner, M., Rausch, A. and Schindler, B. 2012. Automated Verification of Functional Interface Compatibility. 01 (2012).

[8] Krishna, S.N. and Trivedi, A. 2013. Hybrid Automata for Formal Modeling and Verification of Cyber-Physical Systems. Journal of the Indian Institute of Science. 93, 3 (Sep. 2013), 419–440.

[9] Lee, E.A. 2010. CPS Foundations. Proceedings of the 47th Design Automation Conference (New York, NY, USA, 2010), 737–742.

[10] Niebuhr, D., Klus, H., Anastasopoulos, M., Koch, J., Weiß, O. and Rausch, A. 2007. DAiSI -- Dynamic Adaptive System Infrastructure. Fraunhofer Institut für Experimentelles Software Engineering.

[11] Rausch, A., Bartelt, C., Herold, S., Klus, H. and Niebuhr, D. 2013. From Software Systems to Complex Software Ecosystems: Model- and Constraint-Based Engineering of Ecosystems. Perspectives on the Future of Software Engineering. J.M. Schmid, ed. Springer. 61–80.

[12] Rausch, A., Muller, J.P., Niebuhr, D., Herold, S. and Goltz, U. 2012. IT ecosystems: A new paradigm for engineering complex adaptive software systems. (Jun. 2012), 1–6.

[13] Totally Integrated Automation - Totally Integrated Automation - Siemens: http://www.industry.siemens.com/topics/global/en/tia/Pages/default.aspx. Accessed: 2015-06-08.