Self-Coordination as Fundamental Concept for Cyber Physical Systems

Franz J. Rammig

Heinz Nixdorf Institut – Universität Paderborn, Paderborn, Germany

franz@upb.de

Abstract. *In this paper we discuss using self-coordination as a fundamental paradigm in designing Cyber Physical Systems (CPS). CPSs are characterized as a combination of connected embedded systems with the Cyber Space via cloud computing. As completely incompatible worlds have to be integrated, innovative solutions are required. Self-coordination is argued to be a promising approach. Following this approach implies new challenges. Some of them are discussed in the paper. Finally it is argued that taking inspirations from the Biosphere seems to be promising.*

1. Introducing Cyber Physical Systems

In a recent publication [acatech 2011] by acatech – the German national academy of technical sciences – Cyber Physical Systems (CPS) are characterized as the software-based integration of a (potentially worldwide distributed) controlling system ("Cybernetic System") with the physical processes to be controlled ("Physical System") together with the appropriate business processes. All these components together constitute a global, interconnected system ("Cyber Physical System") which in turn is seamlessly integrated into global networks ("Cyber Space").

Cyber-Physical Systems usually consist of numerous Embedded Systems (e.g. in devices, buildings, vehicles, traffic infrastructure, production plants, logistics processes, etc.) which, as any embedded system, directly sample physical data and cause physical actions by means of sensors und actuators. In addition to this local operation these embedded systems are connected to any forms of digital networks. This enables them using or providing worldwide available data and services. In some cases they may provide a multimodal man-machine interface which may be dedicated to devices or even general purpose, e.g. via a browser).

Due to this integrative nature of CPS (Embedded Systems and Cyber Space) their evolution has to be characterized from both perspectives. From the embedded systems perspective the following evolution can be observed:

Initially Embedded Systems have been local, isolated, and mono-functional control devices usually running on single Electronic Control Units (ECU). They sometimes then have been expanded to multi-functional systems, still not being interconnected. The first development towards CPS started when loosely interconnected systems came into existence. Usually they are implemented by means of multiple ECUs which are more or less loosely connected into networks. Such loosely coupled systems can evolve into networks of functionally closely interconnected systems. This results in multiple networks of ECUs which are also closely interconnected into networks. The entire

interconnection structure may be highly mission-critical. This close interconnection structure allows for extending this concept in a recursive manner, resulting in systems of systems. Such systems of systems may expand to globally interconnected systems, the so called Internet of Things.

Looking from the computing perspective the following evolution can be observed:

Initially computing started from global, centralized computing centers for business computing where centrally administered applications were executed on a single, isolated computer. Another direction went into multifunctional systems, being loosely coupled. Such systems consisted of local PCs located in relatively autonomous acting departments. The communication took place via restricted channels and was only partially integrated into applications. A combination of these two trends resulted in Client/Server systems. The processing still happens mostly local but there is an increasing use of globally offered services. A first important step towards CPS is done by bridging business computing and technical computing. This enables interconnecting administration and engineering. Production control now may administratively be monitored. In recent years an increasing virtualization of business computing can be observed, resulting in Grid Computing and later in Cloud Computing. Resources, platforms, services now are offered just as a service. This cloud finally is connected to the embedded systems which are used to control the physical world. This results in globally interlinked integrated systems (CPS), i.e. virtualized Cloud Computing combined with real-time capable, strictly deterministic components, governed by physical constraints.

To sum up: Cyber-Physical Systems are combining the area of embedded systems (real world awareness) with global interconnection (systems of systems) and globally available data and services (cloud).

2. Research Challenges of Cyber Physical Systems

Cyber-Physical Systems connect completely incompatible worlds: There is the area of embedded systems (real world awareness) which is characterized by aspects like real-time (including communication!), dependability, and strictly deterministic behavior. Such systems have to deal with global interconnection in order to build systems of systems. This interconnection has to happen using only partially available deterministic and real-time capable communication protocols. Otherwise just Service Level Agreements (SLA) are available. The situation becomes even more demanding if globally available data and services have to be integrated via the cloud. RaaS, PaaS, SaaS usually show highly probabilistic behavior with respect to timing. They are optimized with respect to average response time. Research in CPS means bridging two currently completely different technical and scientific communities.

3. Cyber Physical Systems as Self-coordinating Systems

For bridging the above mentioned gap towards really functional Cyber-Physical Systems a certain degree of self-organization together with cognitive capabilities seems to be crucial.

This approach starts by engineering local components as intelligent (cognitive) objects providing local real world awareness. Such systems locally have to be strictly real-time

capable. On the other hand, as they cannot continuously depend on global strategies and decisions they must be empowered to make local decisions and afterwards evaluate such decisions as soon as they receive feedback from the global system. I.e. these local systems need to have learning ability. This means the local systems are deterministic with respect to assumptions concerning global properties and objectives concluded from some prediction-making capabilities.

The global interconnection resulting in a system of systems takes care of global coordination and time synchronization. Even more important in such an approach is that at the global level a correction of potentially inconsistent behavior takes place which may have resulted due to local assumptions and predictions of the intended global behavior. So the global control may be characterized as adaptation of local actions instead of predetermination.

Globally available data and services are provided by means of the cloud with all its techniques like RaaS, PaaS, SaaS. By this, a highly adaptive behavior can be achieved, based on continuously receiving sensor information. Self-optimization may be implemented by means of an open market of alternative solutions. At the same time security should be guaranteed at this level while safety should be ensured on the level of local objects and their interconnection.

This approach makes use of basic principles of self-coordination (S.C.). Such systems have in common several characteristics:

Decentralization: S.C. systems are based on decentralized control and components are acting autonomously.

Volatility: S.C. systems are formed by a network of rapidly changing structure, communication behavior, and component behavior.

Scoped knowledge: The global system state is intrinsically not observable and thus we have to deal with components which have to act based on local knowledge only.

Selfishness: Optimization of own benefits is the driving force of a component`s cooperation or potential competition.

Adaptability: Adapting to and learning from environmental changes has to be a universal ability of all components.

Dependability: Independently from these characteristics, a high degree of security and safety has to be ensured. At the same time there are only limited resources available at component level.

4. Selected Approaches to Self-coordinating Cyber Physical systems

First of all, self-coordination means that we transfer many design activities from design time to run time. This makes design more challenging as no a generic system has to be designed instead of a predetermined one. Needed is a corridor-oriented design approach, i.e. the generic system has to be designed in such a way that despite of all variations at run-time it always stays within a predefined corridor of properties. To observe this at design time efficient “faster than real-time” simulation techniques are extremely helpful. New challenges are present in modeling. A communication-centric instead of object-centric modeling approach is needed. The systems to be modeled are highly concurrent

ones; they are loosely coupled, have self-adapting behavior and are embedded into a self-adapting communication infrastructure. Self-modifying higher-order Petri nets seem to be an appropriate approach as by nature Petri nets are communication-centric, highly concurrent, and loosely coupled. Higher-order extensions like Pr/T-nets include information processing. Additional extensions [Kleinjohann et al. 1996] deal with hierarchy and timing. Concerning standardization, they can easily be related to UML Activity Diagrams.

Interfaces are becoming central aspect as E. A. Lee pointed out [Lee 2008]. Interface specifications and composition is an intriguing aspect of model-based design of highly distributed CPS. He argues that it is a promising approach to develop and compose specialized "interface theories" [deAlfaro, Henzinger 2001] including e.g. causality properties [Zhou, Lee 2006] by abstracting temporal behavior, timing constraints [Henzinger, Matic 2006], or protocols [Kopetz, Suri 2003].

CPSs are (at least at a local level) real-time systems. Real-time does not mean that the system has to be fast, it means that it has to be predictable. On the other hand we have to create highly adaptive systems, which make predictability much harder to be achieved. Self-coordination requests adaptation due to unpredictable external demands, unpredictable peer behavior, unpredictable failures, and even an unpredictable market. In the above mentioned paper E. A. Lee [Lee 2008] raised the fundamental question whether the conceptual boundary between the OS and the programming language (established in the 1960's) is still the right one? What would replace them? As a possible compromise we propose to follow a RT-capable virtualization approach [Kerstan et al. 2010]. Self-coordination anyhow requests virtualization due to integration of legacy code, resource partitioning (Multi Cores!), isolation of applications (e.g. billing!), scalability, transparent use of MPSoCs, and cross-platform portability/migration.

CPSs in many cases are mission-critical systems with rather strict requirements concerning dependability. The adaptive nature of the proposed approach makes it necessary to add on-line monitoring and on-line verification to the traditional design-time quality assurance means. These on-line monitoring and verification techniques have to be executed under strict real-time constraints. In our case we developed an on-line Model Checking technique [Samara et al. 2010]. Conceptually it is bounded Model Checking with floating initial state. It is not intended to verify the entire component to be correct with respect to given properties but just that it respects these properties for the next few steps. This limited preview just has to be long enough to allow the system component to be transferred into some fail-operational state in case a property violation has been detected.

Our fundamental paradigm for CPS implies that fault tolerance has to be provided (local miss-interpretation of the global goals have to be corrected afterwards). This fault-tolerance can make use of the adaptability potentials of the system components. This allows following active fault-tolerance techniques instead of just predetermined fault-tolerance means. It is our approach to provide Artificial Immune System techniques to implement active fault tolerance [Montealegre, Hagenkötter 2010]. Detection of sabotage or internal miss-behavior can be achieved efficiently be means of Formal Immune Networks. The repairing operation can be handled by reconfiguration either of

the underlying hardware (if reconfigurable hardware constitutes the hardware platform) or the software structure. Of course, fault-tolerance always needs redundancy. In our approach, however, static redundancy of the system itself is replaced by a redundancy of information.

5. Lessons learned from Nature

There is already a highly successful self-coordinating Cyber Physical System in existence: The Biosphere. So despite of all the differences which should be considered carefully, it might be helpful to observe fundamental "design patterns" that can be observed in biological systems.

Principle 1: Follow a cell-based approach. Nature invented life by inventing cells. Whatever species or colony of species we are looking at, they always are built bottom-up as a collection of cells. Cells include intelligent I/O (cell membrane), static code, a reproduction mechanism, a complete chemical plant (cell plasma), energy management, and a fundamental motion mechanism. I.e. cells are self-contained concerning information, energy, and material flow.

Principle 2: Follow a federation approach. Higher species are made by (large) collections of cells that may cooperate very closely. Cells may be differentiated into highly specialized ones but cells never loose their autonomy. I.e. biological systems are federated ones. The same concept is maintained in social insects („macro cells") and even higher up to human societies.

Principle 3: Elasticity. Higher species are made by different, highly specialized cells. But there stays some degree of elasticity: components dedicated to specific tasks can take over other tasks up to a certain degree. This elasticity is an extremely valuable principle to achieve robustness. It constitutes a good compromise between efficiency (division of labor) and avoidance of single point of failure

Principle 4: Broad variation of communication techniques. Nature invented the entire bandwidth of communication techniques including "broadcasting" (via hormones, cytokines, or pheromones) or multicast/unicast as seen in the nervous system. There is "wired" communication" in the nervous system or "wireless" one, e.g. via stigmergy. There is even "power line communication" in case of hormones or cytokines. All these techniques are based on basic capabilities of cells. Common to all techniques is the principle of delegation. Communication takes place strictly by sending messages. It is up to the receivers how to react (Publish/Subscribe).

Principle 5: Delegation. Large complex systems need a high degree of self-organization or even self-coordination. Pre-planned control and operation seems not to be adequate in such a context. The principle of delegation reduces dramatically the amount of information to be communicated. Having intelligent receivers allows sending just data. The receivers can decide, whether and how to react.

References

Actech (2011) "Cyber-Physical Systems: Innovationsmotor für Mobilität, Gesundheit, Energie und Produktion" (in German), Springer Verlag

Kleinjohann, Bernd; Kleinjohann, Lisa; Tacken, Jürgen (1996) “The SEA Language for System Engineering and Animation”, In: Applications and Theory of Petri Nets, LNCS 1091

E. A. Lee (2008) “Cyber Physical Systems: Design Challenges, In: Proc. International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing (ISORC), IEEE.

L. deAlfaro and T. A. Henzinger (2001) “Interface theories for component-based design”, In: First International Workshop on Embedded Software (EMSOFT), volume LNCS 2211, pp. 148–165, Springer-Verlag.

Y. Zhou and E. A. Lee (2006) “A causality interface for deadlock analysis in dataflow”, In: ACM & IEEE Conference on Embedded Software (EMSOFT), ACM.

T. A. Henzinger and S. Matic (2006) “An interface algebra for realtime components”, In: 12th Annual Real-Time and Embedded Technology and Applications Symposium (RTAS). IEEE

H. Kopetz and N. Suri (2003) “Compositional design of RT systems: A conceptual basis for specification of linking interfaces”, In: 6th IEEE International Symposium on Object- Oriented Real-Time Distributed Computing (ISORC), IEEE

Kerstan, Timo; Baldin, Daniel; Groesbrink, Stefan (2010) “Full Virtualization of Real-Time Systems by Temporal Partitioning”, In: Proc. of the 6th International Workshop on Operating Systems Platforms for Embedded Real-Time Applications , pp. 24-32, ArtistDesign Network of Excellence on Embedded Systems Design

Samara, Sufyan; Zhao, Yuhong; Rammig, Franz Josef (2010) “Integrate Online Model Checking into Distributed Reconfigurable System on Chip with Adaptable OS Services”, in: Distributed, Parallel and Biologically Inspired Systems, IFIP Advances in Information and Communication Technology, Vol. 329, pp. 102-113. Springer Boston

Montealegre, Norma; Hagenkötter, Sebastian (2010) “Process integrated wire-bond quality control by means of cytokine-Formal Immune Networks”, In: Journal of Intelligent Manufacturing