Service Self-Contextualization in Cyber-Physical Systems based on Context Modeling and Context Variation

Alexander Smirnov^{1,2}, Kurt Sandkuhl², Nikolay Shilov^{1,2}, Nikolay Teslya^{1,2}

¹ SPIIRAS, 14 Line 39, 199178 St. Petersburg, Russia ² ITMO University, Kronverkskiy pr. 49, 197101 St. Petersburg, Russia smir@iias.spb.su; kurt.sandkuhl@uni-rostock.de; {nick, teslya}@iias.spb.su

Abstract. Internet-of Things (IoT) and Cyber-Physical Systems (CPS) are considered as the key elements of the next industrial revolution. Operation and configuration of such systems require new approaches for managing the variability at design time and the dynamics at runtime, which is caused by changing application environments. The paper proposes to integrate concepts for variability management with context modelling and self-organization in intelligent systems. Self-contextualization is used to adapt behaviors of multiple services to the current situation and context variants for delimiting the extent of adaptation options. The main contributions are an analysis of variability challenges in IoT/CPS based on an industrial case, the concept of context variants as contribution to manage variability and an initial validation using a case study.

Keywords: Cyber-physical systems, self-organization, self-contextualization, context variation.

1 Introduction

In many industrial sectors, Internet-of Things (IoT) and Cyber-Physical Systems (CPS) are considered as the key elements of innovative solutions. CPS are expected to be essential for higher efficiency and flexibility [15]. IoT allows for data collection and new functions in smart connect products [20]. Operation and configuration of such systems require new approaches and techniques for managing the variability at design time and the dynamics at runtime, which is caused by a multitude of component types and changing application environments. This paper proposes to integrate concepts from product line engineering for systems. More concrete, we propose to explicitly model the "context" of IoT/CPS solutions, identify variants of the context and use these context variants for self-contextualization of IoT/CPS solutions.

A central concept of our work is "self-contextualization" which aims at autonomously adapting behaviors of multiple services to their current operational context. For this reason, the presented conceptual model enables context-awareness and context-adaptability of the service. Using on an application case in industrial production lines, the paper illustrates selected challenges as starting point for our conceptual contribution on integrating variability management and self-organization. For this purpose, a certain degree of formality is required in CPS models, which will also be subject of the paper and based on previous work on a reference model in self-contextualizing services [17].

The main contributions offered by this paper are an analysis of variability challenges in IoT/CPS solutions based on an industrial case, the concept of context variants as contribution to manage variability and an initial validation using a case study. The remaining part of the paper is structured as follows: Section 2 gives a brief overview to background for this work including variability management and context computing. Section 3 introduces the concept of self-organization including context wariants by considering a case study based on an assembly product line. Finally, section 5 summarizes the paper and discusses future work.

2 Background

This section summarizes the conceptual background for our work with focus on variability management (2.1) and context computing (2.2).

2.1 Variability Modelling

Capturing and representing variations in sub-systems, sensors or other elements of IoT/CPS solutions including the relationships or dependencies to other components is an essential task in context computing. The area of variability modeling offers concepts how to deal with variability in complex systems, which might be applicable for CPS and will be briefly presented in this section.

Variability modeling offers an important contribution to managing the variety of the variants of systems by capturing and visualizing commonalities and dependencies between features and between the components providing feature implementations. Since more many years, systematic management of variants is frequently used in the area of technical systems and in software product lines [5]. Feature models are one of the variability modeling approaches often used in product lines and product families. The purpose of a feature model is to extract, structure and visualize the commonality and variability of a set of products. Commonalities are the properties of products that are shared among all the products in a set, which places these products in the same category or family. Variability are the elements of the products that differentiate and show configuration options, variation points and choices that are possible between variants of the product and aim at satisfying different customer requirements. Feature diagrams are used to visualize the hierarchy and other properties of a feature model; they express the relation between features. The exact syntax of feature diagrams is explained in [5].

Recent work on variability modelling also addresses the field of services and service line engineering. A method for service line engineering is proposed in [6] that

bundles all variations of a Software-as-a-Service (SaaS) application based on a common core. The authors of [7] work in the same area and use variability models to derive customization and deployment information for individual SaaS tenants. Unlike conventional distributed agent-based systems, the resources of CPS interact in both cyber and physical space. For this reason, the mechanisms developed for agent-based systems in most cases are not efficient in CPS [8].

2.2 Context Computing and Context Modelling

Context-computing plays an important role to enable services adapting to situations in IoT/CPS solutions [2, 3]. The term "context" has been used and still is subject of research in various application areas and sectors of computer science. In the most general meaning, context describes what relates the entity under consideration to the environment surrounding this entity. What an "entity" is depends on the actual interpretation of context. In this paper, we use the term context according to Dey, who defines context as "any information that can be used to characterize the situation of an entity, where an entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and the application themselves." [4]

Context-computing is first introduced in 1994 by [22]. They consider the context as the information about located-object and the changes to object over time. With increasing mobility of users, increased performance and functionality of mobile devises and sensors, and increasing amount of information available, context computing also gains of importance in order to integrate circumstances and situations of the users, what is often referred to as human related context.

Although context-computing is widely used in computer science, there is no general representation and development procedure for context models. Many authors of context-based systems describe the way of developing the context model for their specific application, but do not provide a general view. [18] and [19] show examples for UML-context development in pervasive computing and OWL-based context for reasoning applications. Mena and colleagues [10] sketch a development process for context –aware systems and identify invariant characteristics of context as part of their work. These characteristics are (a) context relates always to some entity, (b) is used to solve a problem (c) depends on the domain and (d) is a dynamic process. [9] propose a method for context modelling in information systems.

3 Context Variants for Service Self-Contextualization

Our approach for increasing flexibility and controlling variability in IoT/CPS solutions consists of the principle of self-contextualization (section 4.1) and detailed specification of context variants (section 4.2).

3.1 Self-Contextualization in CPS

From a technical viewpoint CPS tightly integrate physical and IT (cyber) systems based on interactions between these systems in real time [16]. CPS rely on control infrastructures commonly consisting of several levels with different components, such as sensors, actuators, computational resources, communication services, etc. The cyber and physical spaces of CPS are represented by sets of resources. The resources have some functionality in result of which they provide services. In this work, the term service is used to describe a software or hardware functionality offered by the service provider to a service user – in our case resources - by a defined interface and including constraints and policies for the service usage. With this definition, we are in line with definitions from the area of service-oriented architectures (see, e.g. [25]).

The services provided by one resource are consumed by other resources. Since the resources are numerous, mobile, and with a changeable composition, the IoT/CPS solutions belong to the class of variable systems with dynamic structures. Restriction to only planned resource interactions in such systems is only a theoretical option, in practice this basically is just impossible. Resource self-organization is the most efficient way to organize interactions and communications between the resources making up IoT/CPS solutions. In order to achieve the dynamics of the self-organizing system, its components have to be creative, knowledgeable, active, and social. The resources that are parts of a system permanently change their joint environment what results in a synergetic collaboration and leads to achieving a certain level of collective intelligence.

In order for distributed systems to operate efficiently, they have to be provided with self-organization mechanisms. In IoT/CPS solutions such mechanisms concern self-organization the system's resources. The goal of the resource self-organization is support of humans in their decisions, activities, solution of the tasks, etc. At that, humans are the participants of the self-organization process, as well.

The process of self-organization of a network assumes creating and maintaining a logical network structure on top of a dynamically changing physical network topology. The autonomous and dynamic structuring of components, context information and resources is the essential work of self-organization [11]. The network is self-organized in the sense that it autonomously monitors available context in the network, provides the required context and any other necessary network service support to the requested services, and self-adapts when context changes.

Due to the nature of CPS, semantics is one of the necessary bases to ensure that several resources arrive at the same meaning regarding the situation and data / information / knowledge being communicated. Ontologies provide for a shared and common understanding of some domain that can be communicated across the multiple CPS' resources.

The present research inherits the idea of ontology usage for modelling context in CPSs. According to [12], any information describing an entity's context falls into one of five categories for context information: individuality, activity, location, time, and relations. The individuality category contains properties and attributes describing the entity itself. The category activity covers all tasks this entity may be involved in. The context categories location and time provide the spatio-temporal coordinates of the

respective entity. Finally, the relations category represents information about any possible relation the entity may establish with another entity.

The context is purposed to represent only relevant information and knowledge from the large amount of those. Relevance of information and knowledge is evaluated on a basis how they are related to a modelling of an ad hoc problem. Resource's context is described by location, time, resource individuality, and event. Resources perform some activity according to the roles they fulfil in the current context and depending on the type of event. On the other hand, the type of activity that a resource performs defines the type of event. The context is updated depending on the information from the service's environment and as a result of its activity. The ability of a system (service) to describe, use and adapt its behavior to its context is referred to as self-contextualization [13].

3.2 Concept of Context Variants

As explained in section 4.1, self-organization depends on context information. As the same set of context information potentially can be used for different IoT/CPS solutions or for different configuration or resource combinations in the same IoT/CPS solution, we propose the concept of context variant (cf. [21]). Context variant is a predefined structured sub-set of all potential contexts of a component for defining constraints in behavior of the component for this sub-set. Parametric knowledge set is the sub-set of system-related knowledge relevant for a specific context variant.

In order to further specify context variants, this section presents a semi-formal definition. In this definition we also consider the fact, that variants can be composed of different alternating, optional or mandatory sub-variants. Decomposing variants in such a way will ease definition of dependencies between variants.

A context is a tuple *Cxt*:={*C*, *CT*, *PK*, *PKT*, *P*, *PT*, *CV*, *VS*, *type*, *map*, *specify*}, consisting of

- disjoint sets *C*, and *CT* whose elements are called context elements and context element types, respectively; and a function *type_C*: *C* → *CT*, that assigns a type *ct_i* ∈ *CT* to each *c_i* ∈ *C*.
- disjoint sets *PK* and *PKT* whose elements are called parametric knowledge elements and parametric knowledge element types respectively; and a function $type_p: PK \rightarrow PKT$, that assigns a type $pkt_i \in PKT$ to an element $pk_i \in PK$
- for each $ct_i \in CT$ a function map: $CT \rightarrow PKT$ that defines which parametric knowledge element type can be specified by which context element type and for each c_i of the type ct_i a function *specify*: $C \rightarrow PK$ which updates the parametric knowledge element corresponding to the context element
- disjoint sets *P*, and *PT* whose elements are called parametric knowledge set elements and parametric knowledge set element types, respectively, with $PT \subseteq PKT$ and a function $type_{PS}$: $P \rightarrow PT$, that assigns a type $pt_i \in PT$ to each $p_i \in P$.
- a set of context variants CV with $\forall cv_i \in CV: cv_i \in PK \cup P$.

Furthermore, we define a variability specification as a tuple VS:={CV, R, man, opt, alt, req, excl}, consisting of

• the variation set *CV* introduced above and a set *R* whose elements are called relations; *CV* and *R* are disjoined sets.

- A function *man*: R → 2^{CV×CV} that relates mandatory variants. With *man*(R) = (CV₁, CV₂) we define CV₂ as a mandatory sub-variant of CV₁.
 A function *opt*: R → 2^{CV×CV} that relates optional variants. With *opt*(R) = (CV₁,
- A function *opt*: $R \to 2^{CV \times CV}$ that relates optional variants. With *opt*(R) = (CV_1 , CV_2) we define CV_2 as an optional sub-variant of CV_1 .
- A function *alt*: $R \to 2^{CV \times CV}$ that relates alternative variants. With *alt*(R) = (CV_1 , CV_2) we define CV_2 as an alternative sub-variant of CV_1 .
- A function *req*: $R \rightarrow 2^{CV \times CV}$ that relates required features. With *req*(R) = (CV_1 , CV_2) we define CV_2 as a required variant for CV_1 .
- A function *excl*: $R \rightarrow 2^{CV \times CV}$ that relates mutual-exclusive features. With *excl*(R) = (CV_1 , CV_2) we define CV_2 is mutual-exclusive to CV_1 .

4 Context Variation Usage for Automated Production Line

This section illustrates the usage of the context variation in an Industrie 4.0 environment. Integration of the Internet of Things concepts in the industrial environment makes it possible to significantly increase the level of automation and flexibility enabling self-adaptation of the industrial equipment to the changing situation. In the case study, we consider a production line responsible for assembling optic devices consisting of a lenses and frames (a detailed description can be found in [23]). The production line consists of three handling units. Handling unit 1 (Fig. 1, R1) gets a tray from the storage facility (Fig. 1, Storage Facility) and loads it on the on a self-controlled carrier (Fig. 1, location S1). There are two types of trays in the system: trays containing frames and trays containing lenses ([24]), and R1 handles them by turns: one carrier is loaded with a tray with frames, and the next one is loaded with the tray with lenses. The loaded carriers move to the location S2 equipped with a controller and bar code sensor, where handling unit R2 unloads trays from carriers and puts them to locations S3 and S4 for trays with lenses and frames respectively. Then, handling unit R3 performs an assembly through putting lenses into frames (Fig. 2).

When the assembly is finished, the tray with assembled parts is transferred by R2 back to the carrier, which brings them further along the production line for gluing and other technological operations up to the final product assembly are done.



Fig. 1. Automated production line ([23])



Fig. 2. Handling unit R3

In the test implementation of the described above case study, the interaction process between handling units R1 and R2 was considered. Below, the description of the scenario in an algorithmic way is presented:

- 1) Self-controlled carrier transfers the first tray with lenses to S2.
- 2) The controller installed at S2 detects (through reading the bar code) the presence of the carrier with a tray with lenses at location S2.
- 3) The tray with lenses is handled by R2 from the carrier to the assembly table, location S3.
- 4) The next carrier having a tray with frames arrives to S2.
- 5) The controller of S2 detects the carrier with a tray with frames at location S2.

- 6) The tray with frames is moved by R2 from the second carrier to the assembly table, location S4.
- 7) R3 takes lenses from the tray, moves them to S4 and installs into the frames one by one.
- 8) If a complete set of lenses and frames is assembled, the system goes to step 11.
- 9) If a lens is missing but the assembly process is not completed, the tray with lenses is loaded by R2 back to the carrier and transferred to the storage facility (S1) to get the missing component. After that, the system goes to step 1.
- 10) If a frame is missing but the assembly process is not completed, the tray with frames is loaded by R2 back to the carrier and transferred to the storage facility (S1) to get the missing component. After that, the system goes to step 1.
- 11) The tray with frames with installed lenses is loaded by R2 to the carrier, which transfers it to the next assembly stages.
- 12) The system goes to the initial state.

The detailed description can be found in [23].

The test implementation showed that establishing complex communications and interactions between production units significantly increases the complexity and decreases the reliability of the entire system. Introduction of the context variation can simplify the implementation.

In accordance with the notation given in sec. 4.2, the context elements related to the case study can be described as follows (fig. 3):



Fig. 3. Context elements

 $C = \{location\}$ $CT = \{enum\}$ $PK = \{location2; location3; location4\}$ $PKT = \{S2, S3, S4\}$ $P = \{nothing: carrier with lenses; carri$

 $P = \{nothing; carrier with lenses; carrier with frames; carrier with assemblies; carrier with empty tray; carrier with lenses (missing lens); carrier with frames (missing frame); tray with frames; tray with assemblies; tray with lenses; empty tray <math>PT = \{22, 23, 24\}$

 $PT = \{S2, S3, S4\}$ $CV = P \cup PK$

R2 and R3 can be formulated as follows (only mandatory variants are considered):

<u>R3</u> (fig. 4) location3 = tray with frames (mandatory) location4 = tray with lenses (mandatory) Run "insert lenses into frames" program



Fig. 4. Activity diagram for handling unit R3

 $\frac{R2}{Context}$ (fig. 5) Context variation 1: location2 = carrier with lenses (mandatory) location4 = nothing (mandatory) Run "move tray from S2 to S4" program

Context variation 2: location2 = carrier with frames (mandatory) location3 = nothing (mandatory) Run "move tray from S2 to S3" program

Context variation 3: location2 = empty carrier (mandatory) location4 = empty tray (mandatory) Run "move tray from S4 to S2" program

Context variation 4:

location2 = empty carrier (mandatory) location3 = tray with assemblies (mandatory) Run "move tray from S3 to S2" program



Fig. 5. Activity diagram for handling unit R2

Such context variations were quite easy to implement. For testing purposes FESTO¹ equipment was used controlled by CPX-FEC device implementing easyIP & Modbus protocols and HTTP requests.

5 Summary and Future Work

Using an industrial example from CPS in production, this paper motivated the need for controlling variability and for the introduction of context variants into self-organization and self-contextualization of IoT/CPS solutions. Part of the work was to formalize the concept of context variant. Furthermore, the paper presents industrial cyber-physical system for two robots interaction in a configuration workstation. The systems is based on the Industrie 4.0 concept that is a new paradigm of intelligent manufacturing systems based on Internet of Things, internet services, cyber-physical systems, and cloud technologies. Robots interact with each other through the smart space infrastructure, which is developed based on Smart-M3 information sharing platform. Special software for the robot controllers has been developed that allows to implement scenarios based on control actions. The control actions are supported by the developed smart space services for each robot, which interact with each other in the smart space and control robots in the physical space.

¹ Festo AG & Co. KG, http://www.festo.de

The biggest limitation of our work is that it was implemented and used only in one prototype scenario, i.e. conceptual and technical feasibility have been shown but pertinence for real-world production has not been demonstrated yet. Context variant conceptualization were developed with the intention to serve as basis for implementing services for use in IoT/CPS solutions. We found it valuable to conceptualize the overall behavior of CPS.

Future work will include conceptual and technical activities. From a technical perspective, experimentation with the use of context variants is one of the planned activities. From a conceptual point of view, we plan to investigate effect of using context variants on the engineering process of IoT/CPS solutions. The specification of variations probably has to be part of the design and specification of the overall CPS, which also will affect requirement elicitation.

Acknowledgment. The research was partially supported by the Government of Russian Federation (grant 08-08), grant of the Russian Foundation for Basic Research #16-29-04349 and State Research no. 0073-2018-0002.

References

- 1. Serugendo, G. D. M., Gleizes, M.-P., Karageorgos, A.: Self-Organisation and Emergence in MAS: An Overview, *Informatica*, 30, 45–54 (2006)
- Preuveneers, D., Berbers, Y.: Internet of Things: A Context-Awareness Perspective, (eds. Yan, L., Zhang, Y., Yang, L. T., Ning, H.) *The Internet of Things: From RFID to the Next-Generation Pervasive Networked Systems*. Auerbach Publications, Taylor and Francis Group, New York, USA, 287-307 (2008)
- 3. Zhang, D., Huang, H., Lai, C.-F., Liang, X., Zou, Q., Guo, M.: Survey on Context-Awareness in Ubiquitous Media, *Multimedia Tools and Applications*, 1-33 (2011)
- 4. Dey, A. K.: Understanding and Using Context, *Personal Ubiquitous Computing*, 5(1), 4-7 (2001)
- Thörn, C., Sandkuhl, K.: Feature Modeling: Managing Variability in Complex Systems, (eds. Tolk, A., Jain, L. C.) Complex systems in knowledge-based environments. Theory, models and applications, 168. Springer, 129-162 (2009)
- Walraven, S., Van Landuyt, D., Truyen, E., Handekyn, K., & Joosen, W.: Efficient Customization of Multi-Tenant Software-as-a-Service Applications with Service Lines. *Journal of Systems and Software*, 91, 48-62 (2014)
- Mietzner, R., Metzger, A., Leymann, F., Pohl, K.: Variability Modeling to Support Customization and Deployment of Multi-Tenant-Aware Software as a Service Applications. *Proceedings of the 2009 ICSE Workshop on Principles of Engineering Service Oriented Systems*, IEEE Computer Society, 18-25 (2009)
- Lin, J., Sedigh, S. & Miller, A.: A Semantic Agent Framework for Cyber-Physical Systems, (eds. Elçi, A., Koné, M. T. & Orgun, M. A.) *Semantic Agent Systems*, Springer, Studies in Computational Intelligence, 344, 189-213 (2011)
- Sandkuhl, K. and U. Borchardt (2014) How to identify the relevant elements of "context" in Context-aware Information Systems? 13th International Conference on Business Informatics Research (BIR 2014), September 22-24, 2014, Lund (Sweden). LNBIP, Springer Verlag.
- 10...Mena, T., Bellamine-Ben Saoud, N., Ahmed, M. and Pavard, B. (2007): Towards a Methodology for Context Sensitive Systems Development, in Kokinov, B., Richardson, D.C., Roth-Berghofer, T.R. and Vieu, L. (Eds.), Modeling and Using Context, Lecture

Notes in Computer Science, Vol. 4635, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 56–68.

- 11.Ambient Networks Phase 2. Integrated Design for Context, Network and Policy Management, Deliverable D10.-D1, 2006, Retrieved September 09, 2014 from http://www.ambient-networks.org/Files/deliverables/D10-D.1_PU.pdf.
- 12.Zimmermann, A., Lorenz, A., Oppermann, R.: An Operational Definition of Context, *CONTEXT 2007*, LNAI, 4635, Springer-Verlag, 558–571 (2007)
- 13.Raz, D., Juhola, A. T., Serrat-Fernandez, J., Galis, A.: *Fast and Efficient Context-Aware Services*, John Willey & Sons, Ltd, 2006.
- 14.De Mola, F., and Quitadamo, R. (2006). Towards an Agent Model for Future Autonomic Communications, *Proceedings of the 7th WOA 2006 Workshop From Objects to Agents*, Catania, Italy, Sept. 26-27, 2006, Retrieved June 18, 2012, from:
- $15. Deutsche Bank Research http://www.dbresearch.de/PROD/DBR_INTERNET_EN-PROD/PROD00000000333571/Industry+4_0%3A+Upgrading+of+Germany\%E2\%80\%99s+industrial+capabilities+on+the+horizon.pdf$
- 16.Horvath, I., Gerritsen, B. H. M.: Cyber-Physical Systems: Concepts, Technologies and Implementation Principles. Horvath, I., Rusak, Z., Albers, A. Behrendt, M. (eds.) Proceedings of TMCE 2012, 19-36 (2012)
- Smirnov, A.; Sandkuhl, K.; Shilov, N.; Telsya, N. (2015) Context Variation for Service Self-Contextualization in Cyber-Physical Systems. Proc. BIS 2015, LNBIP 208, pp. 1–13, 2015. Springer. DOI: 10.1007/978-3-319-19027-3_25
- 18.Henricksen, K., Indulska, J. and Rakotonirainy, A. (2002): Modeling con context information in pervasive computing systems; Pervasive '02: Proceedings of the First International Conference on Pervasive Computing; Pervasive '02: Proceedings of the First International Conference on Pervasive Computing; 167–180; Springer-Verlag, London, UK
- 19.Wang, X., Zhang, D., Gu, T. and Pung, H. (2004): Ontology based context modeling and reasoning using owl; 18–22.
- 20.Porter, M. E., & Heppelmann, J. E. (2014). How smart, connected products are transforming competition. Harvard business review, 92(11), 64-88.
- 21.Smirnov A., Sandkuhl K., Shilov N., Telsya N. (2015) Context Variation for Service Selfcontextualization in Cyber-Physical Systems. In: Abramowicz W. (eds) Business Information Systems. BIS 2015. Lecture Notes in Business Information Processing, vol 208. Springer, Cham.
- 22.Schilit, B. N., Theimer, M. M.: Disseminating Active Map Information to Mobile Hosts, Network, IEEE; 8(5), 22-32 (1994)
- 23.Smirnov, A., Levashova, T., Shilov, N., Sandkuhl. K. (2014): Ontology for cyber-physicalsocial systems self-organisation. Proc. FRUCT-16, pp. 101-107, IEEE 2014, ISBN 978-5-7577-0489-0.
- 24.Kashevnik, A., Teslya, N., Padun, B., Kipriyanov, K. and V. Arckhipov (2015) Industrial Cyber-Physical System for Lenses Assembly: Configuration Workstation Scenario. *Proceedings of the 17th Conference of the Open Innovations Association FRUCT*, 2015, pp. 62-67.
- 25.OASIS (2006) OASIS SOA Reference Model Reference Model for Service Oriented Architecture 1.0. OASIS Standard, 12 October 2006. Available at: <u>http://docs.oasis-open.org/soa-rm/v1.0/soa-rm.pdf</u>, accessed 2018-09-03.