Towards a flexible control center for cyber-physical systems

Martin Franke
Technische Universität Dresden
Nöthnitzer Str. 46, D-01187 Dresden
martin.franke@tu-dresden.de

Diana Brozio
Technische Universität Dresden
Nöthnitzer Str. 46, D-01187 Dresden
diana.brozio@tu-dresden.de

Thomas Schlegel
Technische Universität Dresden
Nöthnitzer Str. 46, D-01187 Dresden
thomas.schlegel@tu-dresden.de

ABSTRACT

Today a variety of functionality, like home automation, entertainment or health advice, is running on widely spread consumer hardware, like home servers, mobile phones or consoles. An intelligent interconnection of such hardware forms cyber-physical systems (CPS). This kind of novel systems composes complex ubiquitous systems, in order to connect the physical reality with the digital world. In this paper we describe a possibility to integrate and control sensors and actuators in a seamless manner to an existing system. To achieve the objective, we use semantic model technologies, like ontologies and reasoning over these for this flexible and knowledge-oriented integration of cyber-physical systems. The knowledge is acquired on the one hand by model instances and on the other hand by runtime information and user interactions with the participating devices.

Author Keywords

cyber-physical system; smart space; semantic models; ubiquitous systems

ACM Classification Keywords

H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous

General Terms

Theory; Design; Performance; Human Factors; Verification.

INTRODUCTION

The development of cyber-physical systems is a complex process. It handles the vision of the "disappearing technologies" and the ubiquitous computing [12] with the help of wireless technologies [4].

Jaroucheh et. al [5] presents requirements for middleware-based context-aware applications that are very similar to cyber-physical systems. These conditions are: coordination of all resources, interoperability and heterogeneity of devices and systems, mobility of the user, autonomous behaviour of the system and potentially auto-discovery of services and devices. In order to achieve this requirements, descriptive models and a middleware, which can handle multiple devices and resources decentralized, are necessary.

The PERSONA project [1] builds one solution for middleware-based context-aware applications. The particularity is the self-organizing infrastructure that handles point-to-point communication with all devices based on service discovery and service adaptation. Components have to register with communication buses to the PERSONA middleware for finding each other and collaborating over these buses. Over these buses, it is also possible to interact in a multi-modal manner with the system. Another outcome of this project is the concept of a context reasoner that aggregate and interpret context information to situations.

Another solution for middleware-based context-aware systems is the Network Automated Machine (NAM) [7]. The NAM is a tool for describing a network of nodes, e.g. devices and provided services in a cyber-physical system. A node is described by its set of physical resources and a set of functional modules that are in turn characterized by a set of provided services, consumed services, consumed context events and provided context events. The service model is based on the IOPE approach of OWL-S [6] that defines a set of input and output parameters, and the precondition and effects of the service. The NAMs can interact with each other over rules, the so called policies, that invokes the specific service on the node.

The central issue of the introduced approaches is the lack of flexibility in the use of multiple devices and interaction concepts. The PERSONA middleware defines communication channels in the time of registration, so the components subscribe for one or more fixed context information from special services. The NAM approach can handle later added cyber-physical components, but is also to fixed in the communication channels of nodes. Thereby we focussed on the aspect of flexibility, based on semantic model descriptions for interaction methods and derived data flow in this work.

The reminder is structured as follows. The second section introduces the VICCI project with its vision and goals. The third section shows a possible architecture concept derived from the requirements for cyber-physical systems. In the

fourth section we outline two scenarios according to our concept. Finally, the fifth section concludes our paper and gives insight to our future work.

THE VICCI PROJECT

The principal purpose of VICCI [11] is the dynamic assisting of the user in cyber-physical systems. This assistant leads to help visualizing and controlling of cyber-physical systems, from Individual Smart Spaces (ISS) towards Smart Communities (SC) [9], in an intuitive, efficient manner.

As motivation of the project deemed the rapid development and ubiquity of technological components, like embedded computers or high-level sensors. The control and the combining of this high technology components is a complex and difficult process. Also an efficient reuse of existing infrastructure in this smart spaces has to implemented.

The control center can be understood as an adaptive, ubiquitous dashboard, that can be viewed from all devices, that are connected to it. The user shall be able to interact with the control center leveraged by different devices and interaction concepts. For this a multi-device interaction is necessary, e.g. the user interact with the same actuator on its PC and mobile device. This ability is acquired by encapsulate functionality in so called Apps. Because of the user-centred approach of the VICCI project, only Apps with an user interface are examined, however they are can run in background without the need to close.

We prefer to use semantic technologies like ontologies and reasoning over these, for automated and knowledge-oriented combining of Apps with their used sensors and actuators. The next section show our architecture concept for the ubiquitous visual frontend and the underlying backend.

ARCHITECTURE CONCEPT

Poovendran [8] describes a major difference between CPS and a regular control system or an embedded system in the use of communications, which adds reconfigurability and scalability as well as complexity and potential instability. CPS still has significantly more intelligence in sensors and actuators as well as substantially stricter performance constraints.

The following chapters introduces the potential frontend- and backend-architectures in VICCI.

Ubiquitous Visual Frontend

Comprehensive cyber-physical systems allow the interaction of devices and objects about use-borders away. For our work an adaptive operating surface for the CPS controlling center is to be developed importantly.

Hervas et. al [2] describes the necessary of user interface adaptation to offer personalized information to the user. The kind of user, display and associated visualization requirements may change while handling with the control center for cyber-physical systems. By representing context information, the environment will be able to react to situation changes and determinate the services to be displayed.

Figure 1. Adaptive visual control for multi-devices

Figure 1 shows an abstract of a potential visual robot-control in a smart home scenario. It provides adaptive visualization on connected devices, including smartphones, tablets and PCs.

The topical trend towards the stronger interlinking aims at mobile devices, like tablets or smartphones. Because of the reduced screen size, in comparison to the customary PC, the visualizations and interactions draughts must be adapted to the hardware specifications of such devices. So the information content varies. In addition to the implementation of an adaptive, multi-device solution, it's necessary to support an adaptive, multi-modal control of the CPS by applications. So the interaction with the system can also vary. At the PC the user can handle with the mouse, on a tablet by finger touch or via voice with smartphones.

The integration of different interaction concepts combined with several devices supported the ubiquitous appendage. In this case, we need on the one hand semantic descriptions for the physical resources, like screen size, computation power or technical interaction capabilities for the execution devices. And on the other hand, applications that describe their functional abilities to react and adapt itself on this contextual information. It is than possible to interact with one App on several devices with different interaction methods deduced from the application and the device description.

This context adaptation shows increasingly a key requirement for mobile and ubiquitous systems for our purpose and will be considered in further work. For a flexible data flow in such a system, we introduce our backend architecture for seamless integration in following section.

Seamless Integration Backend

As mentioned before, the user interaction with the cyber-physical system (CPS) is done by the user over Apps. This section shows the infrastructure of our CPS, see Figure 2. The individual CPS elements, applications as well as sensors and actuators, discover the Semantic Middleware to register and communicate seamlessly with the system. Based on the concept of semantic model descriptions on each layer, it leads us to a highly flexible and distributed system.

Application Layer

Apps are executed on the Application Layer, which can distributed over multiple devices like smartphones, tablets or home servers. This layer is contemplate in first order as an abstract layer, so the interoperability is given over multiple

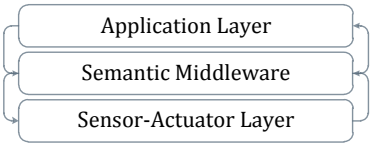

Figure 2. VICCI Backend - Main Structure

heterogeneous devices with different operating systems. The only requirement for every application is to instantiate the App model and provide an interface for data transmission in both directions with the Semantic Middleware. An idea for this connection is presented in the section Conclusion. The App model describe in addition to the user interaction properties the information about input and output data. This means on the one hand the processing and visualization of sensor data in the bottom-up direction and on the other hand the transmission of control instructions to actuators in the top-down direction, that is demonstrated in the section Scenarios. The service part of the App model is adopted from the IOPE approach from OWL-S [6]. But the input and output parameters are described and parametrized in a semantic way, like "get *all* temperature data in the *bathroom*", "get *all* rooms, *that are cooler than 18*°C" or "open *all* windows in the *living room*", which are forwarded to the Semantic Middleware. As mentioned, this description holds an functional part, like "get ...temperature in ..." and a parametrized part "*all*, *living room*", which can be changed during runtime. With this approach the whole system stays highly flexible, because there is no need to adapt the App functionality or change parameters if more sensors and actuators are added to the CPS. The pre- and postcondition of the service, respectively the function of the application, are used for error checking.

Semantic Middleware

The Semantic Middleware (SeMiWa) has the task to acquire, store, interpret, aggregate and route all data flow in the CPS. The acquiration is done over a network interface, which stores all information and knowledge about the individual CPS elements (model instances) in a registrar and opens interfaces for transmitting data to the SeMiWa. The interpretation unit decomposes the semantic annotated IOPE descriptions and annotates in further progress plain data according to device and aggregation models. Another function of the interpretation unit is the knowledge tracking of errors and their solutions adapted from the reaction of the user, if pre- or postcondition fails. The routing unit opens interfaces to Apps and sensors/actuators, and handles the notification of events, based on the IOPE descriptions and the registrar information.

Sensor-Actuator Layer

The Sensor-Actuator Layer is, similar to the Application Layer, designed as an abstract layer, which can also be spread over multiple devices like microcontrollers, robots or home servers. On this layer, low-level sensors and actuators have to implement the so called Semantic Driver, a process that provides a network interface from the hardware devices to the SeMiWa and annotates the plain data against the given sensor/actuator model. If no or an incomplete model for the device is present, SeMiWa tries to annotate it with the right model instance and set the Semantic Driver during runtime.

SCENARIOS

The following section describes two scenarios, which show the data flow through our system. The common situation of both scenarios is the engaged state, in which all components of the cyber-physical system are registered at the Semantic Middleware (SeMiWa) and exchanged their model instances with it.

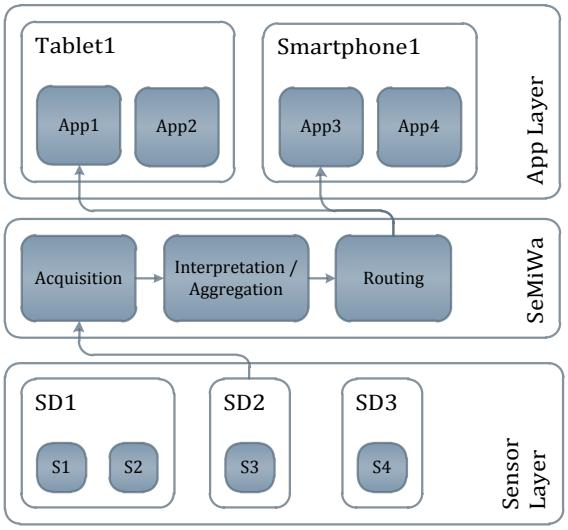

Figure 3. Data flow from sensors

The first scenario Figure 3 shows the data flow bottom-up from the sensor to the user interface via the SeMiWa. The Sensor-Actuator Layer is simplified for this example as pure Sensor Layer. The Sensor Driver *SD2* acquires the plain sensor data from *S3*, and transmit it to the middleware. In due to the registration of *SD2* within SeMiWa, this data can be interpret based on the exchanged model. SeMiWa routes this semantic sensor data to all subscribed Apps, which are interested in this event (*App1* on *Tablet1* & *App3* on *Smartphone1*). These Apps processes the data and visualize it to the user.

The second scenario Figure 4 shows the data flow top-down from one App to one actuator. The numbers in the figure symbolize the order of the ongoing steps. *App1* sends a semantically annotated and parametrized control instruction "open *all* windows in the *living* room", to SeMiWa. After the interpretation of this message, a constraint error is detected based on the precondition "don't open windows in *living* room, if the heatings in *living* room are opened". The user is notified about this error in *App1* and decide to use *App2* to trim off the heatings (*SD2*). The system tracks this decision based on the error, the involved Apps in the right order and the instructions which solved the error. After dissolving of the error, the user uses *App1* again to "open *all* windows in the *living*

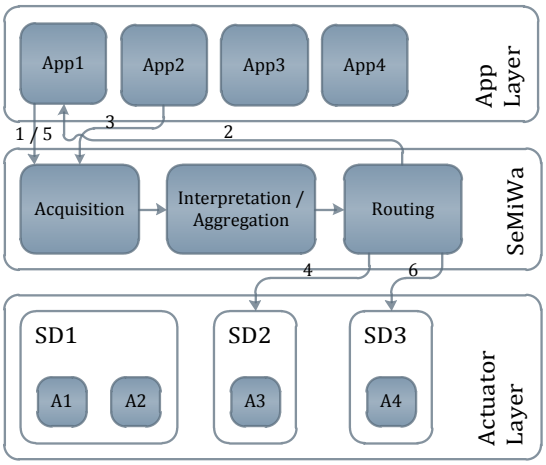

Figure 4. Control flow to actuators

room". This message can now be forwarded to *SD3* without any problems. The system can now assist the user with recommendations, based on previously made decisions if the same error chain reoccurs.

CONCLUSION

We introduced our vision about an interactive control center for cyber-physical systems. This means on the one hand a highly adaptive user interface, which can be spread by Apps on multiple devices based on the underlying model descriptions.

On the other hand, we presented our backend architecture concept. This Semantic Middleware (SeMiWa) helps us to develop a highly flexible and robust system, based on semantic model descriptions on each layer. The input and output descriptions of Apps and the Semantic Drivers stay flexible according to the composed statements, like "close all windows in all rooms". So it is unnecessary, if windows are removed or added to the cyber-physical system. The pre- and postcondition descriptions help us to detect errors and provides a way for knowledge-tracking of the user-made solutions.

As future work, we are focus on the challenging problem to create applicable semantic models for our issues and provide applicable user interfaces on the participating devices. Dashboards can be a good candidate to centralize this large amounts from distributed information, in spite of restricted representation possibilities. Further more an adaptive interaction concept for multi-user- and multi-device-dashboards has to developed for enabling the interoperability and heterogeneity of all participating devices. For the early prototypes, we are going to use technologies like OSGi[1] and Soprano[2] for the development of SeMiWa to get an efficient, and probably realtime, runtime system with lifecycle management. For the App prototypes, we use an Android[3] Smartphone with an UPnP connector for in-house communication and XMPP for WAN communication outside the same subnet. This leads to a flexible, distributed system and interoperability on each layer with all devices [10, 3].

[1] http://www.osgi.org/

[2] http://soprano.sourceforge.net/

[3] http://www.android.com/

ACKNOWLEDGEMENTS

This work is funded under reference ESF-100098171 by means of the European Social Fund (ESF) and the German Free State of Saxony.

REFERENCES

1. Fides-Valero, ., Freddi, M., Furfari, F., and Tazari, M.-R. The persona framework for supporting context-awareness in open distributed systems. In *Ambient Intelligence*, E. Aarts, J. Crowley, B. de Ruyter, H. Gerhuser, A. Pflaum, J. Schmidt, and R. Wichert, Eds., vol. 5355 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2008, 91–108. 10.1007/978-3-540-89617-37.
2. Hervas, R., R., and Bravo, J. Towards the ubiquitous visualization: Adaptive user-interfaces based on the semantic web. *Interacting with Computers 23* (2011), 40–56.
3. Horng, M.-f., and Chen, Y.-t. A new approach based on XMPP and OSGi technology to home automation on Web. *2010 International Conference on Computer Information Systems and Industrial Management Applications (CISIM)* (Oct. 2010), 487–490.
4. Issarny, V., Caporuscio, M., and Georgantas, N. A Perspective on the Future of Middleware-based Software Engineering. In *Workshop on the Future of Software Engineering : FOSE 2007* (Minneapolis, United States, 2007), 244–258.
5. Jaroucheh, Z., Liu, X., and Smith, S. A perspective on middleware-oriented context-aware pervasive systems. In *2009 33rd Annual IEEE International Computer Software and Applications Conference*, vol. 2. IEEE Computer Society Press, 2009, 249–254.
6. Martin, D., Paolucci, M., McIlraith, S., Burstein, M., McDermott, D., McGuinness, D., Parsia, B., Payne, T., Sabou, M., Solanki, M., Srinivasan, N., and Sycara, K. Bringing semantics to web services: The owl-s approach. In *Semantic Web Services and Web Process Composition*, J. Cardoso and A. Sheth, Eds., vol. 3387 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2005, 26–42.
7. Muro, M., Amoretti, M., Zanichelli, F., and Conte, G. Towards a Flexible Middleware for Context-aware Pervasive and Wearable Systems. In *Engineering* (2010).
8. Poovendran, R. Cyber-physical systems: Close encounters between two parallel worlds. *Proceedings of the IEEE 98*, 8 (aug. 2010), 1363 –1366.
9. Suo, Y., and Shi, Y. Towards initiative smart space model. In *Pervasive Computing and Applications, 2008. ICPCA 2008. Third International Conference on*, vol. 2 (oct. 2008), 747 –752.
10. Suo, Y., and Shi, Y. SSCP: An OSGi-based communication portal for Smart Space. *2009 Joint Conferences on Pervasive Computing JCPC*, 2008 (2009), 309–314.
11. VICCI Research Group. Visual and interactive cyber-physical systems control and integration, 2012. `http://vicci.inf.tu-dresden.de/`.
12. Weiser, M. The computer for the 21st century. *Scientific American 265*, 3 (January 1991), 66–75.