Preprints of the 19th World Congress
The International Federation of Automatic Control
Cape Town, South Africa. August 24-29, 2014

Development of a Framework for Dynamic Function Deployment and Extension by Using Apps on Intelligent Field Devices

Mathias Schmitt*. Matthias Loskyll*.
Detlef Zuehlke*

* *German Research Center for Artificial Intelligence (DFKI), Innovative Factory Systems (IFS)*
Trippstadter Strasse 122, 67663 Kaiserslautern, Germany
Forename.Surname@ dfki.de }

Abstract: The aim of this paper is to give an insight regarding the current work in the field of mobile interaction in industrial environments by using established interaction technologies and metaphors from the consumer market. The main objective is the development and implementation of a holistic “app framework”, which enables dynamic function deployment and extension by using mobile apps on intelligent field devices. As a result, field device functionalities can be updated and adapted effectively in accordance with well-known “app concepts” from the field of consumer electronics to comply with the urgent requirements of more flexible and changeable factory systems of the future. In addition, a much more user-friendly and utilizable interaction with field devices can be realized. Proprietary software solutions and device-stationary user interfaces can be overcome and replaced by uniform, cross-vendor solutions.

1. INTRODUCTION

The increasing individualization requirements, combined with the associated technical heterogeneity and complexity of products, require more flexible factory systems able to cope with today’s rapidly changing market needs. Consequently, industrial field devices also need to be adapted continuously to changes in the production process (Abele *et al.*, 2011).

While nowadays proprietary software-tools and stationary, device-linked user interfaces are used to interact with field devices, there is a need for utilizable and holistic, manufacturer-comprehensive interaction solutions to satisfy the demands of flexibility and adaptability. The human, as a crucial part of factory systems, needs to become enabled with smart technologies and methods to overcome these issues and to meet the rising complexity.

The transfer of established interaction technologies and metaphors from the field of consumer electronics to industrial applications appears promising. Mobile devices (e.g. tablet, smartphone) are characterized by low initial cost, including a variety of sensors and actors at the same time, novel interaction methods (e.g. multi-touch) and open platforms enabling flexible software-applications on the devices (e.g. apps). As a result, novel approaches for software distribution and deployment originated in the consumer market (Verclas *et al.*, 2012). Meant herewith are modular software components in the form of apps, which are provided via standardized communication services in the network (e.g. App Store), and allow an effective, user-friendly interaction via uniform, mobile user interfaces (e.g. smartphone, tablet). However, a direct mapping of established "app concepts" in the field of consumer electronics to industrial applications is not trivial, due to the fact that current industrial requirements have not yet been taken into account.

Therefore, the aim is to develop and implement a multi-vendor "app framework", which enables dynamic function deployment and extension by using apps on industrial field devices. Thus, a much more user-friendly and useful interaction between the user and the field device can be realized.

For the realization of the proposed app framework, established interaction technologies and metaphors from the consumer market are used, and consequently adapted for industrial use. Based on this, a reference architecture is developed, which is the basis for the implementation of the app framework. The development and realization of a technology demonstrator, as an instantiation of the proposed approach, will serve as a proof of concept.

In the following, an app framework for dynamic function deployment and extension by using apps on industrial field devices will be presented. First, an overview of related work and technological background is given, followed by a general description of the app framework. Thereafter, the reference architecture and its core components are described in detail. The development of a demonstrator for the implementation of the presented reference architecture serves as a proof of concept. The paper ends with concluding remarks and an outlook.

2. STATE OF THE ART

Shortening innovation and product life cycles, together with a growing demand for customized products, lead to growing demands concerning the flexibility and inconstancy of production processes. However, current factory systems are limited in their ability to cope with these requirements, since they are primarily developed and realized as monolithic systems for performing one specific task (Abele *et al.,* 2011). Reconfigurable, adaptable production processes usually imply a much more frequent adjustment of equipment and processes. Strongly related to this, industrial field devices must also be more frequently adapted to changing circumstances.

By using embedded, intelligent information and communication technologies, the technological enabler to address these challenges is already given. The vision of the so-called 4th Industrial Revolution describes the autonomous control of factory systems in this context by using these distributed, collaborating Cyber-Physical Systems (Lee *et al.,* 2008; Broy *et al.,* 2010). In the future, this development will also allow the augmentation of intelligent field devices with modular software components to ensure a higher degree of reusability without any hardware installations or changes. By the use of modular software components running directly on the field devices, a significantly higher flexibility can be achieved.

In current factory systems there are basically stationary, proprietary user interfaces for commissioning, monitoring and maintaining field devices in use. Particular industrial field devices have their own vendor-specific interaction solutions and operating philosophies, which provide user interface integration directly on the field device. With the increasing distribution of these intelligent field devices, the relevance of utilizable user interfaces will increase as well. The human as an integral part of production systems must be supported by smart concepts and methods to reduce complexity and establish transparency.

With Electronic Device Description Language (EDDL) and Field-Device-Tool/Device-Type-Manager (FDT/DTM), two manufacturer-independent approaches have emerged for the parameterization of field devices in industrial automation. By using EDD (Electronic Device Description) for field device integration, device functions and parameters are set by vendors in a standardized, text-based description language (Electronic Device Description Language, short EDDL). Disadvantages of this technology are limitations in the description of complex, graphical representations and functionalities (EDDL, 2013).

The Field Device Tool (FDT) provides a field device integration framework, which allows the management of various device descriptions (Device Type Manager, short DTM). Each DTM is a standalone software object used to interact with the field device and must be installed on the software system which is used. Among compatibility problems, FDT technology disadvantages include the close linking to the DTM provided by the manufacturer to the Windows platform and associated dependencies (FDT, 2013). In fact, both approaches do support the commissioning of field devices (e.g. configuration, parameterization). However, they do not address the reconfiguration or flexible reloading of software functionality into the device. Especially in the context of growing flexibility and adaptability, both EDDL and FDT/DTM are insufficient.

The transfer of established interaction technologies (e.g. Tablets, Smartphones) and metaphors (e.g. multi-touch, apps) for mobile support in the area of commissioning, monitoring and maintenance of field devices is proving to be visionary (Schmitt *et al.,* 2013 a, b). The combination of modern interaction paradigms and modular software components allows an increase in flexibility and dynamism in function deployment and extending industrial field devices (e.g. reloading of field device apps), as well as significantly higher usability and effectiveness in their application (e.g. an automatic software update mechanism). Additionally, the use of standardized communication and user interfaces enable wireless, mobile interaction with field devices.

Since the introduction of the iPhone in 2007, so called app concepts have been established in the field of consumer electronics to simplify the installation, upgrade and uninstalling of apps on mobile devices (Verclas *et al.*, 2012). In most cases, a so called "App Store" in the network serves as a digital distribution platform for modular software components. This service allows users to search and download apps automatically from a catalogue. The main advantage of this concept is the efficient management and maintenance of the apps across the entire software life cycle. In addition, the apps are provided to the user as modular software components via standardized software interfaces. Another advantage is the openness of the platforms, allowing so called third-party applications, which means that a decoupling of software and hardware manufacturing is becoming possible (Schmitt, 2013).

Although there are already a few companies offering industrial apps, these software components are mostly marketing-oriented (e.g. portfolio catalogue). In addition, there are already some vendor-specific approaches, e.g. which provide apps on mobile devices, in order to address their own field devices via fieldbus and PLC (Sasse, 2013). These are usually monitoring functionalities (Bosch, 2013; Siemens, 2013). However, concerning the latest research literature, there are clearly no holistic approaches for the transfer of established app concepts out of the field of consumer electronics and into the industrial domain, to enable a dynamic and flexible adaption of field device functionality by using apps on field devices.

Nowadays complex field devices are usually developed and sold as a complete, self-contained unit. For the customer, the firmware is normally a fixed and unchangeable part of the product. Concerning flexibility and changeability demands, this approach is not very promising. Due to this, new interaction concepts for industrial field devices are needed, e.g. to provide a separation between the sale of hardware, including a firmware framework serving as a software platform, and the sale of available software components in the form of apps as a functional extension (e.g. monitoring app). The apps can either be developed and offered by the component manufacturer or by the so-called third-parties. Conceivable are both fully programmable field devices, as well as the enabling of functionality by the user downloading apps on the field device.

3. REFRENCE ARCHITECTURE FOR THE IMPLEMENTATION OF THE APP FRAMEWORK

Therefore, the goal is to develop a holistic app framework, which allows dynamic function deployment and extension by using apps on industrial field devices. To achieve this, first, a reference architecture is developed as a basis for the implementation of the proposed app framework (see Figure 1). Thus, a high degree of reusability and scalability for the integration of the app framework into factory systems is ensured.

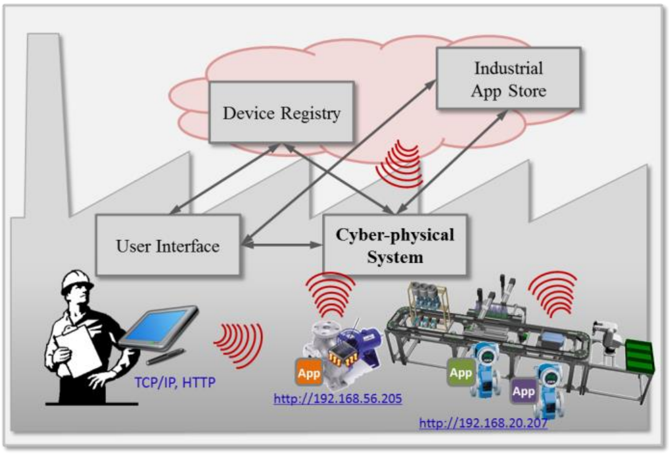

Fig. 1. Reference architecture of the app framework

First, the implementation of the reference architecture and thus, the technological implementation of the industrial app framework takes place. As a result, the realization of core components, which are defined by the reference architecture in the form of specific software elements, evolves. Each software element can be accurately assigned to a system component of the reference architecture. The individual system components of the reference architecture put together result in an overall system and form the basis of the framework demonstration.

Afterwards, the development and implementation of a technology demonstrator takes place. Therefore, the previously implemented software components of the reference architecture are integrated into a coherent, overall system. Based on an industry relevant use case, the developed app framework for industrial field devices can be demonstrated and evaluated. The technology demonstrator serves primarily as a technical proof of concept of the app framework.

In general, the app framework provides manufacturer-independent, modular software components in the form of field device apps, which are provided for the user in the factory network. In this so-called industrial App Store, e.g. there is a vendor catalogue, which lists all available apps for registered field devices. The registration of available field devices is done via a factory-internal device registry, which runs in an internal cloud. For this purpose, all available field devices in the environment dynamically subscribe to this registry (e.g. via an electronic device description). Via standardized, mobile interaction devices (e.g. Tablet), the user is able to access the registry. Therefore, on the one hand, the registry provides an interface to the user and, on the other hand, a standardized interface to the field device. Alternatively, field devices can be equipped with passive data memories. Via the so-called "touch and connect method" (Floerchinger *et al*., 2011), the data memories can be read to initiate communication between the mobile and field device.

The apps can be installed and run directly on the field device. Thus, not only is a flexible function deployment and extension possible (e.g. flexible reloading of apps), but also a comfortable management of software components across the entire software life cycle. The fact that apps are not running on the mobile device anymore, but directly on the field device, is a visionary aspect.

4. OVERALL SYSTEM AND CORE COMPONENTS

Subsequent, the overall system including the core components in their environment is described. Each component has a mandatory function and is correlated to the other components. These relationships lead to dependencies, which need to be considered. The core components of the system and their correlation are shown in Figure 1.

4.1 Field Device Registry

The concept of Cyber-Physical Systems (CPS) defines software-intense embedded systems, which are connected to the global network (the internet). In order to realize a central administration platform for CPS in factory systems, an internal registry concept is required, where available CPS are listed with all pertinent information. The registry provides an interface to the backend (e.g. user interface) and to the CPS. As a result, CPS can register and cancel automatically with specific data profiles in the registry, which leads to a dynamic CPS management in the factory. Via a standardized interface the user is able to access the registry and thus, the available CPS accord to the context. The device registry is classified as a factory specific component that can only be accessed internally by the user and CPS via intranet.

4.2 Industrial App Store

The industrial App Store is a central deployment tool for modular software components (e.g. apps) and realizes two essential functions. On the one hand, the App Store is a distribution and marketing platform for developers and vendors of industrial apps, and on the other hand, a platform for users and CPS to purchase appropriate apps. In accordance with the selected CPS (and thereby with the attached field devices), all compatible apps are listed and selectable. Therefore, the App Store provides an interface to the CPS and the user. The App Store is realized as an external service, which can be accessed via the global network.

4.3 User Interface

The user interface enables the user to interact with the overall system and thus, with the system components via standardized interfaces using e.g. tablets or smartphones. This includes the interaction with the introduced device registry and the App Store, but also with the CPS itself. The component user interface enables the user (e.g. mobile worker) as a central controller and decision-maker in the factory.

4.4 Cyber-Physical Systems

The integration of CPS into the industrial environment leads to autonomous, collaborating and distributed intelligence. There are mainly two distinct approaches in the presented app framework: Enclosed CPS and decentralized CPS (see Figure 2):

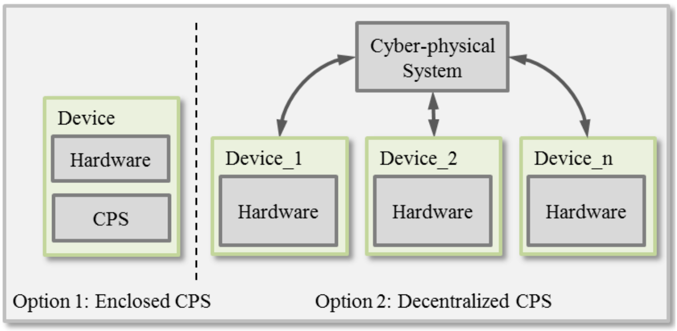

Fig. 2. Considered approaches of CPS in the app framework

The first implementation provides the CPS, according to an embedded system, as an inherent part of the field device. The hardware itself is fixed, which means that the CPS is referred to a defined field device. In the second implementation, the CPS is outsourced and serves as a decentralized embedded system for a variety of field devices (e.g. sensors, actors). This allows the connection of multiple devices to the same CPS.

The CPS is the centrepiece of the presented concept and interacts as an autonomous, distributed and collaborating system with all other components. The CPS is responsible for both the initiation of interactions (e.g. request to App Store) and the execution of operations itself (e.g. installation of an app). Therefore, the CPS provides interfaces to each of the system components, which are the App Store, Device Registry and User Interface. Once the user has selected a CPS via the registry, the interaction takes place directly via the CPS. In addition to the automatic device registration (e.g. after the setup of a new CPS), the CPS executes all operations concerning the App Store (e.g. installation of apps). Because of the fact, that the CPS represents the most important and complex part of the presented system, a more detailed insight into the software architecture of the CPS is given in the following.

For the implementation of a dynamic feature deployment and extension by using apps on field devices, an overall system is described on an abstract level. The description includes both the system components and their relationships to each other. The CPS plays a fundamental role in this system.

5. COMPLEMENTARY CPS SOFTWARE ACHITECTURE

In this section, the developed software architecture of the CPS to realize apps on field devices is presented. The complementary architecture for CPS is divided into three different layers: the functional layer, middleware layer and system service layer (see Figure 3).

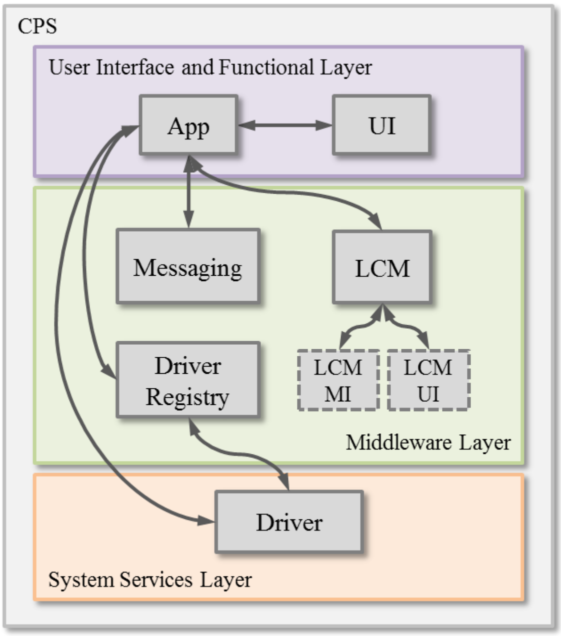

Fig. 3. Representation of the complementary CPS software architecture to realize Apps on field devices

5.1 The Middleware Layer

The middleware is the mediation layer between CPS functionalities (e.g. apps) and the system services (e.g. device drivers). The overall function of the middleware is to enable the performance of cross-developer apps directly on the CPS and furthermore, to allow interaction with the attached hardware (e.g. sensors) via the apps and specific drivers. The middleware consists of different components with distinct tasks.

The Lifecycle Manager (LCM) controls and administrates the apps across the whole software-lifecycle (install, update, uninstall, start, stop). An interface to the apps (e.g. UI), as well as to external components (e.g. status request) is provided. Within the LCM, a distinction between the user interface (LCM "User Interface") and machine interface (LCM "Control Interface") is made. In order to do that, the LCM provides a direct interface to the functionality layer and to CPS internals components.

The driver registry is responsible for the administration and provision of hardware-compatible drivers to system services. In accordance with the used app, specific drivers for interaction with the hardware are connected to the system service layer. This enables access from the app to the device level.

The messaging function supports inter-app communication. Thus, apps are enabled to interact with each other (e.g. process orchestration) or with the infrastructure of the CPS or the environment (e.g. database).To realize this, an interface to the functional layer (e.g. app) and environment (e.g. CPS) is required.

5.2 The Functional Layer: Interface to the User

The functional layer provides the apps and the associated user interfaces. This layer allows the access and interaction with devices (e.g. sensor) via apps. Thus, the functional requirements of the device are addressed. This layer consists of the components app and app UI. The app provides an interface to both the driver registry, to install compatible drivers and to other drivers, e.g. to acquire data or move actors. Furthermore, apps communicate via messaging with other apps and the infrastructure (e.g. exchange of information). The app UI is the interface to the user and allows interaction with the app (e.g. configuration, debugging, functional tests).

5.3 The Driver Layer: The System Services

The system service layer establishes the interfaces to the hardware level and thus, allows the interaction with field devices (e.g. actor). The system services provide specific drivers for apps and the hardware, which need to be controlled. The drivers allow interaction with the connected hardware via the appropriate app (e.g. status request, action triggering). The driver provides an interface to the app itself, to the driver registry, and to the hardware. The developed software architecture for the CPS is the technological enabler to realize apps on field devices. The complementary architecture provides a functional layer, serving as a runtime environment for apps, a middleware, which establishes the interface between the functional layer and the system service layer. The system service layer provides drivers for the interaction with the connected hardware of the CPS. In summary, the software architecture of the CPS is a cut-off from the app to the field device. This allows the processing of standardized apps directly on the field device, independent of the hardware. Thus, a middleware for apps on industrial field devices is developed and realized.

6. SOFTWARE ARCHITECTURE OF THE APPS

In realizing dynamic function deployment and extension by using apps on field devices, it should be noted that the app itself plays a crucial role. Therefore, in this section, a more detailed insight into the app software architecture is given.

The app, as a software module, provides the functionality to field devices. Each CPS or field device can process a variety of apps, while functionalities can be composed of different apps as well. The app consists of the components user interface, configuration, messaging, persistence and the program logic. The messaging component can be divided into the message receiver and the message sender. The receiver accepts relevant messages and forwards them, if wanted, to the program logic. The sender forwards messages to other apps on the CPS or to the infrastructure (e.g. CPS internal data base). The configuration allows access to the functionality settings of the app via the user interface. Additionally, there is a persistence component, which serves as a data collection point for device relevant information. The program logic realizes the functionality of the app itself and is different from app to app. This is the component that needs to be adapted to functional requirements by the developer (see Figure 4).

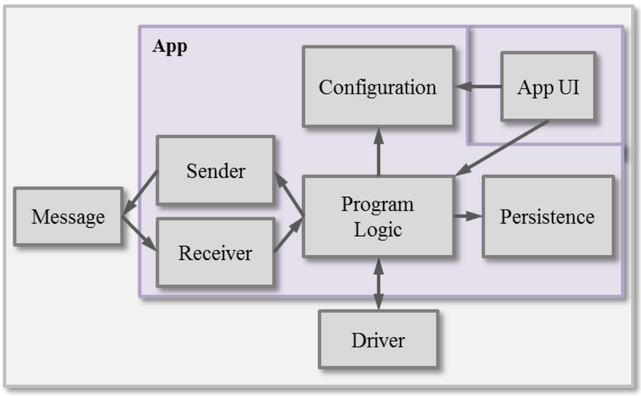

Fig. 4. Representation of the developed software architecture of field device apps

By developing a consistent software architecture for apps, both developers and users are given a concrete implementation basis. The architecture is divided into a generic and a function-specific part. As a result, only the program logic, which is linked to the functional requirements, need to be developed in each case.

7. USE CASE AND DEMONSTRATOR

As a proof of concept, exemplary use cases in the area of mobile configuration and maintenance of field devices have been implemented prototypically in the SmartFactory^{KL} (Zuehlke, 2009; Zuehlke, 2010). Therefore, customized flashlights are mounted in an assembly module. The flashlight consists of three components: bottom of the case, electronics and cover of the case. For the assembly, the components are placed into a fit and grouted within a fixed order. The grouting station consists of an analogue position sensor (Sharp GP2-0430) and an electronic linear axis (Festo EGSL-BS-45-100-3P) with a grouting stamp, which are controlled by a CPS (Aria G 25, development board). The task of the sensor is the detection of the components via infrared (presence check), in order to initiate the grouting process.

Due to the introduction of a new product variant (smart key finder) the components, which need to be grouted, change partially (e.g. new electronic). As a consequence, not only the presence of components has to be checked, but also the distances between the components and the stamp. Via the distances, the product variant can be indicated and thus, the required grouting power. The hardware itself is able to produce both variants, only the processing software of the sensor has to be adapted. Therefore, the user connects via a tablet to the corresponding sensor by selecting it in the device registry. Afterwards, all compatible apps are listed in the industrial App Store. After downloading and installing the "distance app" on the CPS, the user is able to configure the sensor to the individual needs. Defined distances and positions are set for both product variants within the apps. The sensor forwards the information to the CPS. In order to that, the CPS can distinguish between the flashlight and the key finder. With help of the measurements of the sensor, the CPS adapts the grouting power and the travel of the stamp to the product variant.

By reloading the "distance app" onto the CPS, the sensor has been functionally extended, to the new process requirements. Concrete, a simple presence sensor has been extended to a distance senor by reloading a new app. Furthermore, a condition "monitoring app" for the CPS has been implemented, to set key performance indicators (KPI). The user is able to define e.g. threshold values for the set KPI. If the values are exceeded, the user will receive a message on his tablet. In this case, the CPS was extended by a novel functionality.

8. SUMMARY AND CONCLUSIONS

Changeable and flexible factory systems equally imply the need for a consequent adaption of field device functionalities according to the changing production conditions. Especially in the area of field device integration there are primarily stationary user interfaces and proprietary software solutions available to interact with field devices. Thereby, the user is more frequently faced with an increasing variety of field devices from different manufacturers, which leads not just to a considerable effort for the configuration or reconfiguration of field devices, but also carries the risk of incorrect operation by the user.

Therefore, the transfer of established consumer electronics and apps into industry is forced to realize a user-friendly and effective interaction with field devices. Based on this, a reference architecture has been developed, which allows the realization of industrial apps on field devices. The current developments in industry, and in particular the emergence of CPS, are the technological enablers for this approach. As a result, dynamic function deployment (e.g. set up) and extension (e.g. reload of functions) by using apps directly on field devices becoming possible. Furthermore, interaction via standardized, mobile user interfaces has been realized, in order to overcome current proprietary and stationary solutions. In terms of economic aspects, the presented reference architecture provides a novel approach in the industry for software deployment on a field device level. In this way, novel opportunities, in particular in the field of after sales services, arise. As a proof of concept, the reference architecture has been implemented in the SmartFactory^{KL}.

REFERENCES

Abele, E.; Reinhart, G. (2011). Zukunft der Produktion: Herausforderungen, Forschungsfelder, Chancen. Carl Hanser Verlag, München

Bosch on Mobile Devices (2013). Apps from Bosch. Available at http://apps.bosch.com/en/index.htm, 11.11.2013.

Broy, M., Kargermann, H., Achatz, R. (2010) Agenda Cyber Physical Systems, Outlines of a new Research Domain, acatech, Berlin, Germany.

EDDL (2013). Electronic Device Description Language. Available at http://www.eddl.org/Pages/default.aspx, 11.11.2013.

FDT Group (2013). Field Device Tool / Device Type Manager. Available at http://www.fdtgroup.org/, 17.10.2013.

Floerchinger, F.; Schmitt, M. (2011). A Concept for a user-friendly first Communication Initiation between Stationary Field Devices and Mobile Interaction Devices. In S. Bittanti, A. Cenedese, S. Zampieri (ed.), Proc. of the 18th International Federation of Automatic Control (IFAC) World Congress, 1614-1619, International Federation of Automatic Control, Milano, Italy.

Lee, A. E., (2008). Cyber Physical Systems: Design Challenges, Int. Symposium Object / Component / Service-Oriented Real-Time Distributed Computing (ISORC), pp. 363-369, Orlando, SUA.

Sasse, N. (2013). Smart Apps für creative Automation. Computer & Automation. Vol. 9, 26-30.

Schmitt, M. (2013). Using Consumer Electronics and Apps in industrial Environments – Development of a Framework for Dynamic Feature Deployment and Extension by using Apps on Field Devices, accepted for IPC 2013, November 22-23, Sibiu, Romania.

Schmitt, M.; Meixner, G.; Gorecky, D.; Seissler, M.; Loskyll, M. (2013). Mobile Interaction Technologies in the Factory of the Future. IFAC/IFIP/IFORS/IEA Symposium on Analysis, Design, and Evaluation of Human-Machine Systems. August 11-15, Las Vegas, USA.

Schmitt, M.; Zühlke, D. (2013). Smart Phones und Tablets in der industriellen Produktion - Nutzerfreundliche Bedienung von Feldgeräten; atp edition – Automatisierungstechnische Praxis, Vol. 3, S. 58-65, Deutscher Industrieverlag.

Siemens (2013). Siemens Apps. Available at http://www.siemens.de/mobile/apps/Pages/Home.aspx, 11.11.2013

Verclas, S.; Linnhoff-Popien, C. (Hrsg.) (2012). Smart Mobile Apps. Mit Business-Apps ins Zeitalter mobile Geschäftsprozesse. Springer-Verlag.Berlin,Heidelberg.

Zuehlke, D. (2009) SmartFactory - A Vision becomes Reality. In: Keynote Papers of the 13th IFAC Symposium on Information Control Problems in Manufacturing (INCOM 09), June 3-5, Moscow, Russian Federation, ICS / RAS.

Zuehlke, D. (2010) SmartFactory - Towards a Factory-of-Things. Annual Reviews in Control, volume 34, 129-138.