Fostering Access to Data Collections in the Internet of Things

Alexander Kröner
Georg Simon Ohm University of Applied Sciences
Nuremberg, Germany
Alexander.Kroener@th-nuernberg.de

Jens Haupert, Christian Hauck, Matthieu Deru, Simon Bergweiler
German Research Center for Artificial Intelligence
Saarbrücken, Germany
{jens.haupert, christian.hauck, matthieu.deru, simon.bergweiler}@dfki.de

***Abstract*—Using label technology, a physical object may be employed to build a continuously growing data collection, which can, for instance, be exploited for product quality monitoring and lifecycle management. Along the objects lifecycle, queries to such a collection may stay quite similar, e.g., get unusual observations. However, expectations to a good answer may change, as with time different entities will come into contact with the object. This article reports on work in progress concerning a framework for collecting data concerning things, which aims at 1) decoupling logic employed for interpreting such a collection from processing hardware and 2) using the collection itself for transporting such logic. Main contributions include an approach to hardware-abstraction of processing logic at the object or remote, and an app store for retrieving interpretation and presentation logic.**

Keywords*—*Ubiquitous computing; RFID tags; Distributed information systems; Supply chain management

I. INTRODUCTION

Within the Internet of Things, physical objects may function as a focus for digital data and services concerning the artifact itself as well as associated things, people and processes. This function enables an object to take a new role as a data collector and provider in a broad range of scenarios, e.g., users may manually associate data with an object in order to socialize and foster discussion in a community [1], tools may automatically collect usage data in support of pay-by-use accounting [2], and products may steer and document their production [3] and transport rules that support reasoning of healthcare applications [4]. Existing applications of such technology are typically deployed in ”closed” scenarios, i.e., requirements of users and applications are known before the collection process starts.

This reflects only to some extent a supply chain with continuously changing users and requirements. In order to facilitate communication between stake holders in such an ”open” scenario, a uniform interaction behavior of the collection would be advantageous, e.g., a uniform way to ”check integrity” of an object, i.e. compliance to criteria specified by a third party on an individual base for objects or kinds of objects.

In the following, Section II provides an example scenario, where one stakeholder has to employ logics provided by another stakeholder. Section III summarizes requirements that arise from this scenario. Then, Section IV wraps up work accomplished so far concerning so-called Active Digital Object Memories (ADOMe), a platform for processing logic in a way that allows for embracing a broad range of infrastructure approaches common to Internet of things applications. Section V extends this approach with a concept of an app store supporting distribution of the processing logic. Afterwards, work related to ADOMe and app store is briefly reviewed in Section VI, followed by a summary in Section VII.

II. SCENARIO

The following logistics scenario deals with integrity control during transportation of a heterogeneous set of goods (see Fig. 1). Each good is packaged in a way matching its nature (e.g., fragility) and value. All packages are tagged with some kind of label technology, which allows for automatically identifying the object. Depending on the respective kind of package, this technology may range from passive RFID (Radio Frequency IDentification) to embedded systems with integrated sensing and processing capabilities.

A retail chain advertises the quality of products sold in its stores, which is subject of the companys own, particular strong quality guideline. In order to leverage compliance to this guideline, the company provides business partners along the supply chain with constraints on parameters that need to be monitored. A supplier uses these parameters in order to configure an integrity test for each package destined for this particular retailer; for accuracy, input parameters should be sensed and processed by the package itself or by IT infrastructure near the object. Performing this configuration task is supported by a hardware-abstraction layer, which allows for assigning tests to packages independent from the kind of label technology provided by the respective package.

During loading a truck, by means of this layer a dialog between package, truck, and an app store hosting implementations of tests matching the retailers parameters. Result of this dialog is an assignment determining which technical component (truck or package) has to conduct the monitoring task, an assignment, which may differ for each package.

Finally, the packages arrive at the retailers receiving area. There, an employee uses a mobile device in order to identify the respective object and access its digital records values sensed during transport as well as testing methods and their results. The access is performed using an app, which downloads from the app store a method suited to visualize the test chosen by the logistics expert. This visual adaptation is performed automatically in the background; even for very different kinds of packages the employee experiences always the same way of interacting with the respective object.

 ISBN: 978-1-61208-289-9

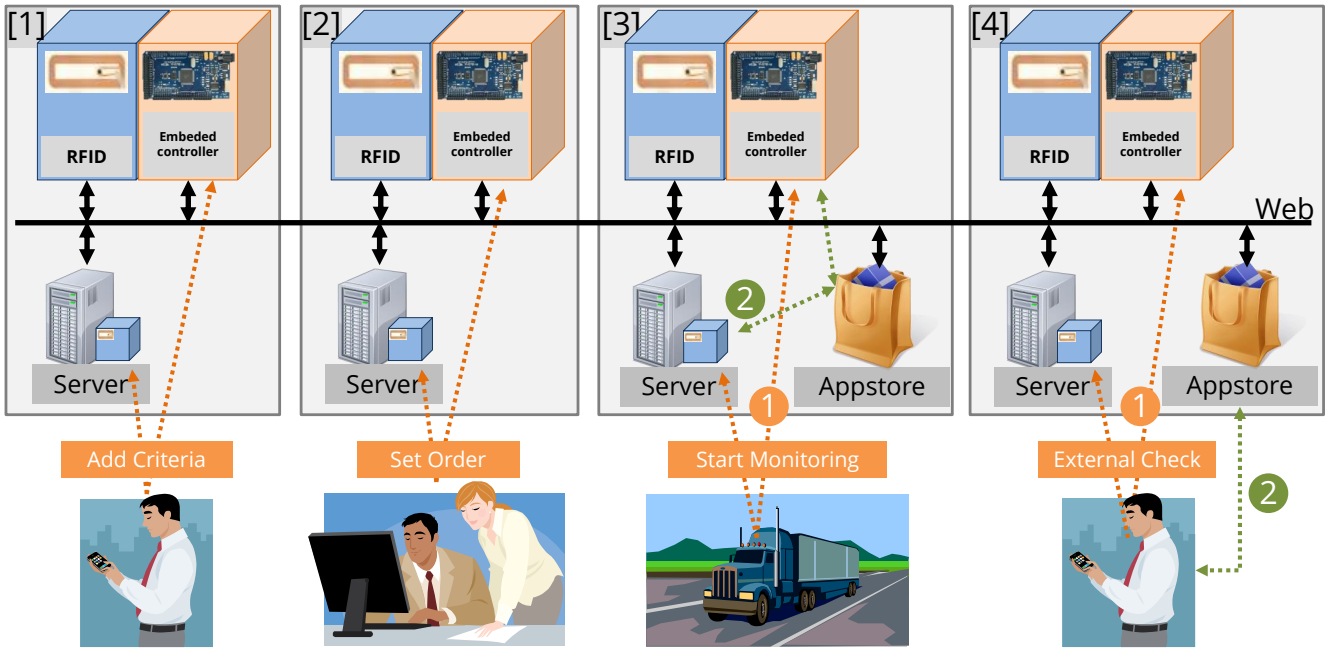

Fig. 1. Logistics scenario with on-product criteria storage [1], monitoring orders [2], active logic processing [3], and external checks [4].

III. Requirements

The envisioned framework has to balance between the need for flexibility (to support different hardware platforms, open processes, new requirements in the future), for standardized structures that ease data retrieval and communication among different and cross-domain content providers and for normative description of object-related logic. Corresponding requirements to the architecture model can be divided in three parts that reflect a 3-tier approach to the realization of such a model, namely the collection model for data storage (1), support for active analysis functionality (2), and a common access architecture and infrastructure support for hardware abstraction with an app-store-like approach (3).

Data storage is the basic functionality of any ADOMe. In order to enable a party to analyze data added to the memory by another party, a common storage model is required, which is shared by all parties along the object's life cycle (Requirement 1, R1). In addition the model should be flexible enough to cover a large variety of data formats (including encodings and further data types) and should ease the process of data retrieval (R2). Due to the cross-domain usage of such memories involving several partners, a common infrastructure that provides an abstract memory access (including protocol and data exchange specifications) independent from any memory implementation or hardware platform is necessary to ease the task of memory access for existing and newly created applications (R3). Setting up activity and analysis support on top of the data storage can extend the functionalities of object memories, by allowing the memory to process data autonomously based on given algorithms. Based on this functionality we want to enable applications to ask common (but pre-defined) semantic questions (R4), rather than processing the entire memory data, which might be a complicated process due to possibly very large and capacious memories and in contrast a slow connection speed. In addition the memory should proactively process data with rules based on expert knowledge, and deploy results to memory storage or return the result on queries (R5). Finally, the system should provide a hardware abstraction layer, to enable a transparent access for clients (R6), which includes a compensation of missing object features by the environment as well as a mechanism to retrieve machine and platform-compatible logic code for the given use case (R7).

IV. The ADOMe Framework

To fulfill the given requirements, we already implemented a framework for active object memories. The data within such an ADOMe are structured according to the Object Memory Model (OMM) [5]. This model partitions a memory in blocks with data of the same dedication and nature (R1). For the content of a block the model does not prescribe any data format or encoding. A block can be created and modified by external entities, which also define the content and encoding of the data of this specific block. The data of such a block can be either stored directly within the block as payload or are just linked to an external web-based source. In addition each block consists of a set of metadata, describing the content the block, creator and contributors, encoding and intention. Applications may use this metadata (e.g., attributes such as format, encoding, and namespace) to choose the best fitting information from the memory or utilize attributes like title and description to present memory blocks in a human readable form to end users (R2).

Accessing data collections for some object usually requires identifying this object as well as looking up the related collection infrastructure. The ADOMe framework does not impose particular constraints on the employed object identifier: in order to look up a memory, the framework resolves identifiers against the "IDs block" of hosted OMMs, which may contain arbitrary identifiers. Thus, access requires in the first place the address of the framework instance. This address is encoded as URIs (Unified Resource Identifier) suited to represent any

 ISBN: 978-1-61208-289-9

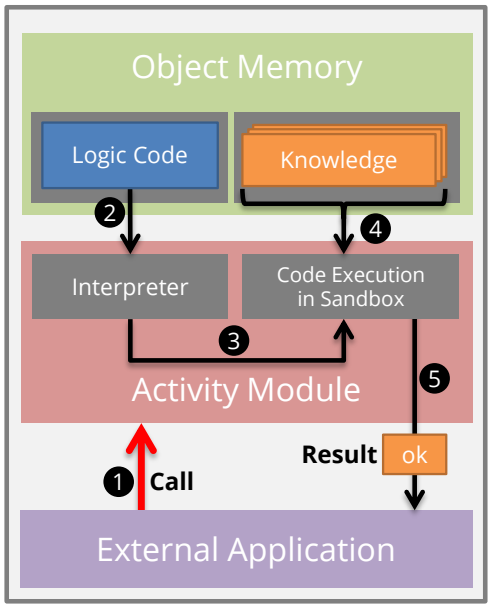

Fig. 2. Execution of ADOMe architecture triggered by external call (1-5).

Fig. 3. Performing built-in and on-product integrity check on a smart package equipped with ADOMe and accessed by a mobile handheld device.

installment. Based on the information about ID and URI, a root path to an object memory inside its container can be determined and directory and feature negotiation services can be accessed by a fixed REST-based (Representational State Transfer) interface that delivers URLs to pieces of the memory as well as functions implemented by the memory (R3).

Using the REST interface and OMM metadata, enables applications to employ their very own decision criteria to retrieve memory contents and process them outside of the framework. They even may store the employed decision criteria as contents within the memory, describe them through metadata, and thus share them with other applications (cf. [4]). However, this approach requires involved applications to implement the same command processor (e.g., a particular rule engine). Furthermore, the approach is little suited if a memory is required to autonomously prepare answers for future requests, as required by many applications focused on monitoring. Therefore, the ADOMe framework allows for extending its basic collection function with additional processing methods, which can then be explicitly selected applications to issue queries like check integrity (R4). Here, two concepts are mandatory for an ADOMe: a feature container providing means for statically implemented, optional, customer specific features, and a sandbox containing a logic processor. Both concepts can be triggered by external calls (see Fig. 2 and 3), by memory change events and by an internal timer (R5).

The feature container of an ADOMe provides a generic means to extend the functionality of object memories with so-called optional features. Installed by the owner of the framework during memory initialization, other parties may invoke and parameterize these features, for instance a producer may tailor its products memories to their respective application area. Optional features have to implement a REST-interface. For deployment, the framework provides a registration mechanism, which links the features services to the REST interface of the ADOMe. In addition, an ADOMe provides a processor for software logic that allows for deploying custom evaluation functions for extracting meaning from data in the memory. If installed by the owner, a feature can be used by other parties to install their own compression, selection and accumulation methodologies, and to handle content types during the memories life cycle not foreseen during memory initialization. The logic processor runs in a sandbox environment that provides a generic interface integrated into the ADOMe's REST interface and a safe API to the memory storage of the ADOMe. The aforementioned framework has been implemented and has been used to realize several use cases utilizing active logic code. Hence, the framework can serve as the foundation for the following extension.

V. Hardware Abstraction & AppStore

Based on the given scenario we focus on a heterogeneous set of goods equipped with different techniques, ranging from simple barcodes to embedded systems with storage and processing capabilities. A common architecture to access different objects with the same approach and to access a similar functionality would be highly desirable from the user point of view. Currently we are developing a concept for hardware abstraction that allows accessing users and the objects themselves to complement their missing functionalities on their own, or at least to inform the outer environment about requested but not available functionalities.

This concept is based on the aforementioned ADOMe framework that is built on modular software components. This allows the framework to run on a large variety of platforms raging from high-end servers to small embedded systems (e.g., with storage capability only). The URI we use for memory identification and the RESTful interface for communication allows for transparent memory access independent of the concrete hardware and software implementation, and enables systems with missing functionalities (e.g., logic processing) to outsource these to server-based solutions by passing-through incoming requests (R6).

To combine software modules with object memory hardware for logic processing several approaches are possible. Applications can install and use software modules located in the memory storage (see Fig. 4). The processing and execution

 ISBN: 978-1-61208-289-9

Fig. 4. Client application displaying ADOMe-based measurements created by a software module downloaded from an external appstore.

location (either in embedded systems on-product or on server-based solutions off-product) is transparent for the access application. In case of no fitting software module available in memory, an access to a so called app store is possible based on a semantically defined logic description. This store contains a large set of logic modules for different applications and platforms, ranging from fixed domain-specific code to generic modules capable of parameterization. If available the framework downloads, installs and executes the downloaded module from the store. Client applications performing operations based on memory data can be extended with in-memory or app store modules the same way (R7).

VI. Related Work

This work is related to research and development concerning frameworks that leverage collecting and processing data related to physical objects, and as such related to the Internet of Things. Related research comprises embedded systems as well as web-based data stores. So-called Collaborative Business Items (CoBIs) ilustrate the benefits of delegeting small parts of a well-defined business process - e.g., monitoring and self-assessment tasks - to objects with embedded sensing and processing capabilities [6]. In order to decouple such a service from the employed hardware, SmartProducts [7] seek to dynamically integrate resources - including web-based structures - in the object's environment into the service realization. Complementary to our proposal, this work puts particular emphasis on semantic device and data descriptions for products with embedded technology. An example of collecting object-related data in a web-based data store is the Tales of Things electronic Memories (TOTeM) system. It seeks to foster communication between humans via personal stories digitally linked with things [1]. Its infrastructure shares aspects of an ADOMe, in particular a unified approach (which actually inspired parts of OMM) for structuring data concerning a thing, and open web-based information storage. EVRYTHNG [8] follows a similar approach in order to enable information sharing concerning objects; it extends it with Active Digital Identities for objects, where services linked with an object employ information collections (concerning the object, or objects of the same kind) in order to adapt to the user. The question of how implementation and provision of such services can be supported is addressed by Xively [9]. The web-based service supports not only hosting and sharing object data, but also software products and descriptions concerning devices, which we propose extending in a way that supports exchanging such components across devices using unified data structures and semantic descriptions.

VII. Conclusion and Outlook

This article summarized work in progress concerning a framework for setting up so-called Active Digital Object Memories. Its contribution is twofold: In order to leverage the application of such data collections in open scenarios, this framework seeks to 1.) embrace different ways of deploying answering logic in a collection, and 2.) provide abstraction from the technical diversity of existing infrastructures for collecting object-related data, which employ technology embedded into physical objects, virtual data stores located in the Web, and combinations of both approaches. Future work will address in the very first place the proposed method of distributing processing logic within this framework: the app store implementation. A first prototype illustrates the feasibility of this approach in a manufacturing scenario involving passive RFID, Android tablets, and embedded devices; however, more efforts are needed to verify that concept for broader range of embedded system platforms as well as logic hosted for deployment.

Acknowledgment

This research was funded in part by the German Federal Ministry of Education and Research under grant number 01IA11001 (project RES-COM). The responsibility for this publication lies with the authors.

References

[1] R. Barthel, K. Leder Mackley, A. Hudson-Smith, A. Karpovich, M. de Jode, and C. Speed, "An internet of old things as an augmented memory system," in *Personal and Ubiquitous Computing*, vol. 17. Springer London, 2011, pp. 321–333.

[2] D. Fitton, F. Kawsar, and G. Kortuem, "Exploring the design of a memory model for smart objects," in *Ambient Intelligence and Smart Environments*, vol. 4: Workshops Proceedings of the 5th International Conference on Intelligent Environments. IOS Press, 2009, pp. 33–38.

[3] P. Stephan, G. Meixner, H. Kling, F. Flrchinger, and L. Ollinger, "Product-mediated communication through digital object memories in heterogeneous value chains," in *Proceedings of the 8th IEEE International Conference on Pervasive Computing and Communications*. IEEE, 2010, pp. 199–207.

[4] M. Schneider, M. Velten, and J. Haupert, "The objectrules framework - providing ad hoc context-dependent assistance in dynamic environments," in *Proceedings of the Sixth International Conference on Intelligent Environments. July 19-21, Kuala Lumpur, Malaysia*. IEEE Computer Society CPS, 2010, pp. 122–127.

[5] A. Kröner, J. Haupert, M. Seiler, B. Kiesel, B. Schennerlein, S. Horn, D. Schreiber, and R. Barthel, "Object Memory Modeling - W3C Incubator Group Report," http://www.w3.org/2005/Incubator/omm/XGR-omm-20111026/, 2011, [retrieved: June 2013].

[6] C. Decker, T. Riedel, M. Beigl, L. Moreira, S. Souza, P. Spiess, and S. Haller, "Collaborative business items," in *Proceedings of IE 07: 3rd International Conference on Intelligent Environments, Ulm, Germany*, 2007, pp. 40–47.

[7] M. Mühlhäuser, "Smart Products: An introduction," in *Constructing Ambient Intelligence: AmI 2007 Workshops*. Springer, 2007.

[8] EVRYTHNG Ltd., "EVRYTHNG Every Thing Connected," http://evrythng.com/, [retrieved: June 2013].

[9] Xively (by LogMein Inc.), "Xively - Internet of Things Platform Connecting Devices and Apps for Real-Time Control and Data Storage," http://xively.com/, [retrieved: June 2013].

 ISBN: 978-1-61208-289-9