Platforms for the Internet of Things – An Analysis of Existing Solutions

Author1, Author2

Department
Organization
Address
ZIP City
email-address@author1

Abstract: The Internet of Things (IoT) gets real. Already today, there are more connected devices than humans. In order to leverage this trend platforms are required to enable building and operating of applications. In contrary, there is hardly any systematic research available that evaluates the capabilities of existing IoT platforms. This paper contributes to the understanding of the current IoT platform landscape (1) by reviewing six selected commercial IoT platforms, and (2) by inducing an architectural model which depicts the main building blocks of an IoT platform in a comprehensive way.

1 Introduction

There seems to be a consensus in the industry that the Internet of Things will be the "next big thing" (e.g. [Adi00, Pro00]). Cisco predicts 50 Billion internet-connected devices by 2020 [00a]. ABI Research speaks of 10 Bn wireless connected devices today, and more than 30 Bn in 2020 [00b]. Already now, there are more connected devices than humans. But has the Internet of Things changed our lives or radically changed how business works? In the 2013 Gartner hype cycle of emerging technologies IoT just reached the "peak of inflated expectations" and the "plateau of productivity" is foreseen to be not reached before the mid-2020s. Machine-to-Machine (M2M) communications and wireless sensor networks (WSN) have already overcome the peak but are still assumed to reach the plateau in 5 to 10 and more than 10 years respectively [00c]. In order to reach the plateau of productivity it is vital that standards and platforms are established that ease the process of building and operating applications on top of an emerging technology. In the case of IoT and, as we see, the related technologies M2M and WSN, an ever increasing number of commercial platforms appear on the market.

The paper is structured as follows. First, we start by setting the term "Internet of Things" in context since it is used inflationary during the last years and is far from being unambiguous. Second, we refine our understanding of an IoT platform and present related work. Third, we review six unique platforms which were selected by a focus group. We will see that each platform has distinct features and shortcomings and is

useful for specific domains. Fourth, on the basis of the individual platform assessments, we induce an architectural model depicting functional building blocks of IoT platforms in a comprehensive way. Thereby, this paper tries to contribute to the understanding of existing IoT platforms. Finally, we discuss the results and identify future research problems.

2 Foundation and Related Work

2.1 The Internet of Things

In late August 2013 the term "Internet of Things" entered the Oxford dictionaries. It is described as "a proposed development of the Internet in which everyday objects have network connectivity, allowing them to send and receive data" [00d]. Obviously, this is more a vision than a definition. Actually, this vision goes back to the year 1991 and an article of Mark Weiser where he stated that computers will finally "weave themselves into the fabric of everyday life until they are indistinguishable from it" and should be aware of their environment [Wei91]. This article can be seen as the origin of the research field called ubiquitous computing.

The term "Internet of Things" itself appeared the first time as the title of a Forbes article in 2002. Kevin Ashton, who headed the Auto-ID Lab at MIT, exclaimed: "We need an 'Internet-for-things', a standardized way for computers to understand the real world" [Sch02]. The Auto-ID Labs are a world-wide research network of EPCglobal which is responsible for the standardization of barcodes and the successors like Radio-frequency identification (RFID). They focus on logistics where goods need to be uniquely identifiable and trackable in order to build a self-organizing network where goods find their way like data packets do in the ordinary Internet [MF10].

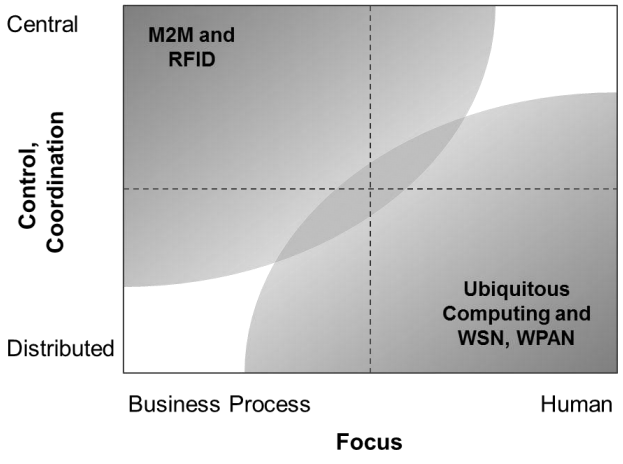

Figure 1: Approaches towards the Internet of Things

Another approach to the IoT stems from telemetry. Here, sensor data from remote places has to be transferred to a central site. With the advent of mobile phones and cellular networks, telemetry evolved to machine-to-machine communications. Typical applications are fleet management and asset tracking. Here, Everyday objects (e.g. cars) and even the environment (e.g. parking lots) get network connectivity.

The industry, especially network providers, often uses the term M2M and IoT interchangeably. In contrast, most of today's M2M applications are heavyweight custom solutions that are in no way based on standards and interoperability. Notably, the term M2M is usually business-centered whereas the vision of ubiquitous computing is human-centered [Mat00]. Furthermore, in typical M2M scenarios, the computation and intelligence is centered at one specific location (e.g. central enterprise app in an RFID scenario) whereas ubiquitous computing is concerned with distributive computing where computational tasks can be distributed among the network [MS03] (see figure 1).

Related to ubiquitous computing, the research field of wireless sensor networks (WSN) and wireless personal area networks (WPAN) emerged in the mid-1990s. Both build self-organizing networks of smart objects which can interact with the physical world around them [DP10]. Today's typical WSN/WPAN scenarios involve proprietary wireless protocols like ZigBee and Z-Wave where gateways are needed to translate between proprietary and IP networks. These gateways often need application specific programming and are not interchangeable. In addition, there are small WPAN solutions using Bluetooth where the smartphone typically acts as a gateway or mediator (e.g. smart watches).

In recent years, efforts have been made to incorporate the IP protocol stack into smart objects. Since the requirements of smart objects differ greatly from that of a usual participant of today's Internet, the IP stack has to be adapted [RN10]. This has to be done such that a transparent end-to-end connection between devices over the Internet is achieved. For this purpose various protocols and technologies like 6loWPAN (IPv6 for low power wireless personal networks), ROLL (routing over lossy links) and CoAP (constrained application protocol) are being in the process of standardization by the Internet Engineering Taskforce (IETF). On the M2M side, standardization processes driven by the European Telecommunications Standards Institute (ETSI) aim to prepare the LTE networks for massive low-throughput non-human communications and therefore full IP connectivity in individual things.

2.2 IoT platforms

IoT platforms provide a comprehensive set of generic, i.e. application independent IS functionalities which can be leveraged to build IoT applications [AtIM10, CaJS12, LP00]. In the context of our work, prior research in the domain of IoT platforms has identified key IoT building blocks on the basis of a deductive as well as inductive manner.

Castro et al. is one of the very few papers which conduct an analysis of M2M platforms based on an inductive approach [CJS12]. The goal of their paper is to provide an initial

analysis of the requirements for an M2M platform. They analyze six available IoT platforms in terms of their scope. However, they do not provide an overall architecture of functional building blocks. Moreover, while it is indeed hard to select platforms on a set of objective criteria in a transparent way, they just select platforms for their analysis without providing any argumentation. To the best of our knowledge, there is no research available deriving an overall architecture of functional building blocks on the basis of existing platforms.

Several other approaches derive functional IoT platform building blocks on the basis of a deductive process. *Atzori et al.* for example provide an overview on IoT developments [AIM10]. As one part of their overview, they describe major building blocks of IoT solutions. While they do not depict a comprehensive survey of each building block, they thrive to convey the role each building block will likely play in the IoT. Overall, they identify three major IoT building blocks: technology (identification, sensing and communication technologies), middleware, and applications. *Atzori et al.* point out the importance of Service Oriented Architectures and decompose middleware into the four building blocks service composition, service management, object abstraction, as well as trust, privacy and security management. As a second example, *Lempert et al.* provide a very detailed architecture based on an ESB architecture [LP00]. However, the overall development process of the artifact remains largely unclear. They hardly argue why each of the building blocks is needed and how they deduced the model.

Summing up, while there are several papers available describing what a IoT platform should offer from an academic perspective, hardly any information is available on the actual capabilities of existing IoT platforms. However, relevant research builds upon relevant practice problems (delta between “what is available” and “what is needed”) [ES04]. Without a solid understanding of existing IoT platforms relevant practice problems can hardly be identified. Therefore, this paper tries to contribute to the understanding of existing IoT platforms by giving an overview over available solutions and inducing an IoT architecture model.

3 Analysis of Existing IoT Platforms

There is a plethora of IoT platforms available today [MM12]. Therefore, to select and analyze relevant IoT platforms a focus group was leveraged. Focus groups are an established means to investigate new ideas and to check the applicability of a research object by practitioners [Tre10]. The focus group was exclusively set up to analyze the IoT platforms and consisted of seven people. Tab. 1 depicts the members of this group, which met three times to select and assess the IoT platforms.

Three key criteria were applied to select IoT platforms for analysis: (1) Market share of an IoT platform as of today, (2) strategic positioning of platform provider in the market, and (3) degree of platform innovation. We are well aware that these criteria are vague and that the overall selection process is heavily dependent on the individual experience as well as subjective assessment of the focus group participants. However, by outlining the focus group as well as the key selection criteria, we try to be as transparent as

possible. The selected IoT platforms will be described in the following subchapters according to three criteria. First of all, the platform provider and its corresponding business model will be portrayed. Second, the intended application domain will be presented. Finally, the platform itself with its structure and functionally will be depicted.

#	Member	Organization
1	Director portfolio strategy and product management team	Software Vendor: M2M, Business Process Mgmt
2	Member portfolio strategy and product management team	Software Vendor: M2M, Business Process Mgmt
3	Director for communities & partner networks in the Internet of Things and Services	Software Vendor: M2M, Business Process Mgmt
4	Head of IoT research lab	Software Vendor: M2M, Business Process Mgmt
5	Scientific lead of IoT research lab	University
6	Research assistant, member of IoT research lab	University
7	Research assistant, member of IoT research lab	University

Tab. 1: Characteristics of the focus group

3.1 Axeda

Axeda [00e] was founded in 2000 and is a classical M2M solutions provider. The first version of the Axeda platform was released in 2009 and in 2011 they introduced a freemium and pay-as-you-grow business model comparable to Amazon Web Services. There is a free trial version for managing a fixed number of connected products and a fixed amount of daily transactions. Exceeding the free trial the license price grows gradually with the number of products and transactions. This business model is well suited for the ever-growing number of M2M and IoT start-ups. Besides, Axeda offers a broad ecosystem with many partners and over 150 customers from different industries. Moreover, it was named as the M2M application development platform market leader by Strategy Analytics earlier this year.

The Axeda platform focuses on supporting the application development process rather than providing connectivity capabilities; although it offers an agent which can be implemented into a physical product to manage the connection to the platform using the proprietary Axeda wireless protocol. Once a product is able to communicate with the platform, the actual product, communication channel and protocol are “hidden” behind a data model which can be accessed and transformed by a rules and a Groovy scripting engine. If the agent is running on the connected product, a simple form of remote programming is possible, i.e. thresholds can be transferred down to the product. Moreover, preconfigured applications, e.g. for location tracking, geofencing, building dashboards or automatic software deployments are available and can be customized to individual needs. Integration with enterprise software like ERP or CRM can be achieved in three ways. The Axeda platform can consume SOAP or RESTful web services via its scripting engine, it can provide web services which can be called from the enterprise software, or the enterprise application can connect to the platform’s message queue.

3.2 Digi, Etherios

Digi International Inc. [00f] was founded in 1985. Initially, the company focused on PC boards. Until today, their focus shifted towards embedded and non-embedded M2M-solutions. Embedded solutions comprise communication modules, satellite communication devices, integrated circuits, and single board computers. Non-embedded-solutions include routers, gateways, and servers.

Etherios, Inc. was set up in 2008 as a cloud service provider. Its services cover the domains of cloud computing in CRM, system integration and application development. They put a special focus on the connection of traditional M2M platforms and the cloud based CRM platform of Salesforce.com, e.g. their "Social Machine" integrates arbitrary machine data into the Salesforce.com cloud solution in real time.

In 2012, Digi acquired Etherios. The integration of both companies extends Digis product and service portfolio enabling Digi to go beyond a pure hardware specialist. The merged company acts as a solution provider offering development, service, support, and licensing services. However, hardware sales remain the main source of revenues for Digi with 95.5 % of total revenue in 2012 [00g]. However, service revenues are anticipated as a more stable source of revenue.

Digis M2M platform started in 2009 as "IDigi Energy" to create a base for smart energy networks. "IDigi Tank" was a second use case providing a tank storage monitoring platform. In the following, Digi adopted the "ZigBee Smart Energy" standard, launched development kits, and explored new use cases like container monitoring and telehealth monitoring. The integration of Thingworx' visualization tools and Etherios' Salesforce.com capabilities further enhanced their platform capabilities.

From a technical point of view, Digi's platform comprises communication modules, gateways and a software platform. Each of the three levels offers extension points for own development. XBee communication modules are based on the industry standard ZigBee. ConnectPort Gateways are programmable in Python. An open REST API allows the development of own web applications.

Summing up, the combination of Digi and Etherios allows a seamless integration of embedded modules into high level CRM systems. Their solutions are well-established in the market thanks to their early market entrance.

3.3 Eclipse M2M

The Eclipse M2M [00h] industry working group is a consortium of companies in the context of the Eclipse foundation. Founded in 2012, its target is to "encourage, develop, and promote opens source solutions that can be used to overcome market inhibitors found in most M2M ecosystems" [00i]. Current members are Axeda, Eurotech, IBH Systems, IBM and Sierra Wireless. Until now, the working group initiated seven different projects: Mihini, Koneki, Paho, Ponte, SCADA, Concierge, and Kura. First results are available for Mihini, Koneki, and Paho.

The Mihini project creates an application environment for the Things in the IoT. Like all three projects, it is based on the programming language Lua. Lua is a lightweight scripting language, which was developed in 1993. It is not adopted to the M2M scenario. This adoption is realized by the Mihini framework. It implements e.g functionalities like I/O management, device management, and application management. It runs on top of Linux systems like the Raspberry Pi. One the one hand, this decision enhances the hardware base of the framework. On the other hand, it does not allow to build small, energy saving devices.

The Paho project focuses on the communication between M2M-Devices by protocols like MQTT and OMA-DM. MQTT is a communication protocol that allows both Request/Response and Publish/Subscribe communication based on TCP. Currently used in messaging applications like "Facebook Messenger", it also allows a stable communication between devices. In contrary, OMA-DM is a device management protocol. It allows the configuration, update and error management on devices. Both protocols are widely used.

Finally, the Koneki project focuses an easy development environment. It offers the Lua Development Tools and the OMA-DM simulator. The Lua Development Tools are an Eclipse-integration for Lua. Syntax highlighting, auto completion and debugging tools follow. The OMA-DM simulator allow to test device management functionalities in advance. A core functionality is the simulation of a firmware update.

In conclusion, the Eclipse M2M group leverages existing Eclipse projects, and provides widely adopted technical protocols like MQTT and OMA-DM. Its embedded framework is heavyweight, own hardware doesn't exist. The adaption of standard M2M communication mesh protocols is missing.

3.4 Thingsquare

Thingsquare [00j] is a Swedish company founded in 2012 by Adam Dunkels who was named as one of the most important innovators under 35 by MIT Technology Review in 2009. This was due to his accomplishment in squeezing the internet protocol in a version that only uses 100 bytes of RAM (uIP) which has been extensively used by companies. The technology that has led to founding Thingsquare is Contiki an operating system just like Windows or Linux but not for PCs, but rather for low-power, memory-constrained and connected devices. Therefore particularly for the things or smart objects in the IoT. Contiki respectively Thingsquare Mist, the commercial derivative, are both released under the BSD license and are open source. The business model, as depicted on the corporate web site, is to provide services and trainings around open source products comparable to the Red Hat Foundation with Fedora Linux. Thingsquare promises to ease up the development process of connected products. It emphasizes the use case of connecting people through smartphone apps with physical products. Examples are the thermostat of Munich based start-up tado and the crowd-funded connected light Lifx, which collected 1.3 Million US$ in a Kickstarter project.

Physical products or things are equipped with a wireless chip running the Thingsquare Mist OS. On this basis, the things in physical proximity automatically build a self-healing wireless mesh network and connect securely (AES encryption) to the Thingsquare cloud backend server via a device called edge or border router. Each device has its own IP- address (IPv6) and the routing in the mesh is based on IETF RPL. The router is distinguished by the fact that it can translate between IPv4 and IPv6 and it is connected to Ethernet or Wifi. The Thingsquare approach suits very well to the vision of IoT since internet standards are used end-to-end, down to the thing and the connection between the mesh and the actual internet is accomplished in a transparent way.

The backend server then provides an API that allows for example smartphone apps to connect with the things. Furthermore, Thingsquare announced a web-based IDE called "Thingsquare Code" to build applications for Mist online. This includes the compilation and distribution process of embedded software to things. Therefore there is no need for a complex tool chain to program embedded devices and change can be done over-the-air at any time. Also the health or state of the things can be monitored online. Beyond these basic features, neither services for data or device management nor services for business logic are publically available or described.

3.5 Thingworx

Created in 2009, Thingworx's [00k] focus differs quite significantly from platforms like Eclipse M2M. Eclipse M2M focuses on the communication of nodes and the web. Contrary, Thingworx focuses on the integration, transformation and presentation of the created data. It does not provide a solution to the question "how is data collected?" but it addresses the problem of "What happens to the data after it has been collected?"

In 2011, Thingworx defined four building blocks for its platform [00l]: Search, Mashability, Composability and Crowdsourcing. These building blocks are realized by the different key features of Thingworx. SQUEAL allows the search of distributed data and devices. The Codeless Mashup Builder allows the creation of applications by "drag and drop". The Composer integrates different visualizations, data storages and business logics. Finally, social networks are used to allow the collaboration of users and developers.

The most important capability of Thingworx is the integration of different channels. Devices can be connected by a variety of protocols like REST, MQTT, or traditional sockets. It offers an interface to business systems like SAP, Oracle and Salesforce.com. It also embeds cloud services like Twitter and Amazon Payments. Finally, social services like Facebook, Twitter and Google+ are linked.

In conclusion, Thingworx focuses on data management in a simple way. It reduces complexity for non-technical users by providing mashup technologies.

3.6 Xively

Xively [00m], originally known as Pachube, was created 2007 as a data brokerage platform by Usman Haque. The name Pachube, pronounced as patch bay (switching board as used in telephone operation centers), describes the intention of the platform. Like the telephone operator that connects a person to other persons, Pachube allows mashing up live or historic data streams, especially from sensors, to build new applications that were not thinkable in the single domains before.

Pachube was noticed by the broader public following the nuclear accident in Fukushima, Japan in 2011. There, many individuals used low-cost DIY Geiger counters to monitor the radioactive fallout across the country [Cou00]. The data was streamed to Pachube and together with the maps API from Google a live map could be created. Later in 2011 Pachube was acquired by LogMeIn and renamed to Cosm. Due to trademark issues in May 2013 the service was renamed again and is known as Xively today. In the course of this process a payment model was released to the public in addition to the free trier of the service. The business model is mainly around service and consulting, but also based on the frequency of sending and receiving data to and from the platform.

Xively provides a multitude of tools to help individuals and companies to build and manage connected products and applications based on connected things. There is an extensive list of API wrappers for different programming languages and platforms. This allows to build embedded software (C, ARM mbed, Arduino, Electric Imp), cloud applications (Java, Python, Ruby), web apps (JavaScript, PHP), as well as smartphone apps (Android, Objective-C) that easily connect to the Xively service.

The Xively web service itself allows to provision, activate and manage devices. In essence this means to give each device a unique identity and specific rights to create and receive data on the platform. Therefore it can be seen as a rudimentary form of business logic, since it allows activating and deactivating additional services coupled to a specific device. Besides, Xively has integrated triggers which can call a web service if, e.g. a threshold in a data stream was exceeded. There is also an API provided for visualizations enabling the development of interactive graphs and dashboards. Many of the development tools around Xively are open source, hosted on GitHub and everyone is explicitly invited to participate.

4 IoT Platform Model

4.1 Derivation of the Model

The induction of the consolidated model is based on the construction of reference models [Bro03, Sch98]. In essence, functional building blocks describing similar functionality are consolidated in the induced model (cf. Table 2). We distinguish three different abstraction layers in the model.

First of all, the analyzed platforms greatly vary in their general scope. Some platforms offer capabilities to develop and run applications on end user devices, i.e. general-purpose computers like PCs or smartphones. Others provide functionality to develop and run embedded applications on "things". Finally, the platforms provide functionally to centrally coordinate and process execution. In M2M scenarios most often M2M middleware solutions and enterprise applications take over the central coordination and processing role. In consumer oriented IoT scenarios like smart home a gateway might act as the central coordination instance.

The functionality provided by the platforms can further be classified into two general functionality types. There are libraries and code frameworks which can be leveraged for application development and execution. We refer to this type of functionality as core functionality. Furthermore, there are tools for development as well as life cycle management, i.e. managing the platform at runtime. As these tools regularly "span across" the core functionalities provided by the platform, we refer to this type of functionality as cross functionality.

The last abstraction layer comprises more fine grained functionality offered by the platforms. The elements of this layer are directly induced from the analyzed platforms. Table 2 make this process transparent by depicting which platform offers which functionality.

Domain	Functionality Type	Functionality	Axeda	Digi, Etherios	Eclipse M2M	Thingsquare	Thingworx	Xively
General-purpose computer, e.g. PC or smartphone	Core functionality	User interface		+			+	
		Lightweight application		+			+	
		Internet connectivity		+			+	
	Cross functionality	Development		+			+	
		Lifecycle mgmt		+			+	
Thing, e.g. decentral sensor, actor	Core functionality	Environmental interface		(+)	(+)	(+)		
		Embedded application	(+)	+	+	+		
		M2M/WPAN connectivity	(+)	+	+	+		
	Cross functionality	Development	(+)	+	+	+		
		Life cycle mgmt	(+)	+	+	+		
Central coordination instance, e.g. gateway or enterprise app	Core functionality	Core application	(+)				+	(+)
		Application integration	+	+		+	+	(+)
		Data management	+	+			+	+
		Identity management	+	+		+	(+)	+
		Device management	+	+				+
		Connectivity	+	+		+	+	+
	Cross functionality	Development	+	+			+	(+)
		Life cycle mgmt	+	+			+	

Tab. 2: Induced architecture model

4.2. Description of the Model

Most of the previously depicted platforms provide functionally to centrally coordinate and process execution. Six functional core building blocks can be distinguished in this context (cf. Table 2, Figure 2).

The business logic of the *application* can be supported by the platform in different ways. Runtime engines, e.g. for Java, process engines, rule engines, and event engines exist. While engines like Java provide a powerful programming environment, rule and event engines provide less powerful yet simpler ways to implement business logic. Most of the platforms focus on the latter approach (e.g. Thingworx, Xively) and thrive for ease of implementation.

Especially platforms with an M2M history like Axeda provide strong out of the box functionality to connect to enterprise applications like ERP, CRM or SCM systems. Furthermore, platforms like Thingworx provide *integration* frameworks to enable rapid connectivity to web services, e.g. via REST-based communication. Key aspects being addressed are authentication, authorization and secure communication. Finally, cloud connectivity to services like Amazon Web Services and social services like Facebook or Twitter is offered.

Data management capabilities include machine data collection and storage as well as data integration services. Some platforms offer capabilities to store and analyze vast amount of machine data. In the context of trends like “Big Data” these capabilities are promoted heavily. Other platforms focus on integrating data from different sources on the basis of simple data integration (“mashup”) environments. They promise to integrate structured as well as unstructured data from social sources.

The data management functionalities also enable device abstraction. The IoT relies on a vast and heterogeneous set of objects, each one providing specific data through its own dialect [CJS12]. Therefore, some platform providers offer predefined data models, so that device specific data models can easily be transferred into device independent models. Hence, in a scenario like fleet management on board units from different vendors can be used without extensive integration effort.

Identity management allows administrating user identities as well as resources, e.g. devices or application services across the platform. Granting and revoking access to resources as well as providing authentication services are fundamental capabilities for executing security policies. Furthermore, monitoring and analyzing access logs are the basis for auditing and security analysis.

Device management provides capabilities to provision, activate and manage devices. This covers functionalities such as remote access, configuration, administration, software management, device monitoring, and troubleshooting. Furthermore, some platforms support the automated delivery of firmware and configuration updates during run-time.

Today’s IoT platforms provide different functionalities to *connect* to things. Most often they support RESTful API communication via HTTP or standard protocols such as

MQTT, enable protocol translation or provide so called agents running on specific device types. These agents are basically software libraries, which can be embedded into a physical product to manage the connection to the platform.

Things, dedicated sensors and actors measure or change the environment. Thereby, they serve as an *environmental interface*. Their *applications* are *embedded* on microcontrollers or adapted to dedicated operating systems. Finally, energy and network restrictions force them to use low-power, fail-safe IoT *communication* protocols like RPL. IoT platforms like Eclipse M2M and Thingssquare provide operating systems for embedded devices like Contiki OS, e.g. enabling wireless mesh networks and secure connectivity. Furthermore, they offer IDEs to build embedded applications as easy and efficient as possibly. These platforms also facilitate the compilation and distribution process of embedded software to things. Moreover, they provide low level monitoring capabilities to manage device networks at run-time. Finally, these platform projects develop and maintain standards for communication like MQTT as well as device management like OMA-DM.

Some platforms provide functionality to rapidly build frontend applications for desktops as well as smartphones. Thingworx for example both claims to provide means for compressing the design-develop-deploy cycle. At the heart of these promises are IDEs that enable composition of applications that "integrate the data, activities, and events from people, systems, and the physical world" [00l]. More specifically these tools build upon modeling and configuration rather than "classical" programming. They offer tools to create HTML5-based *user interfaces* and *lightweight frontend applications*, e.g. on the basis of dashboards. Standard internet *connectivity* is leveraged to connect these frontend applications to the core application running on a central instance.

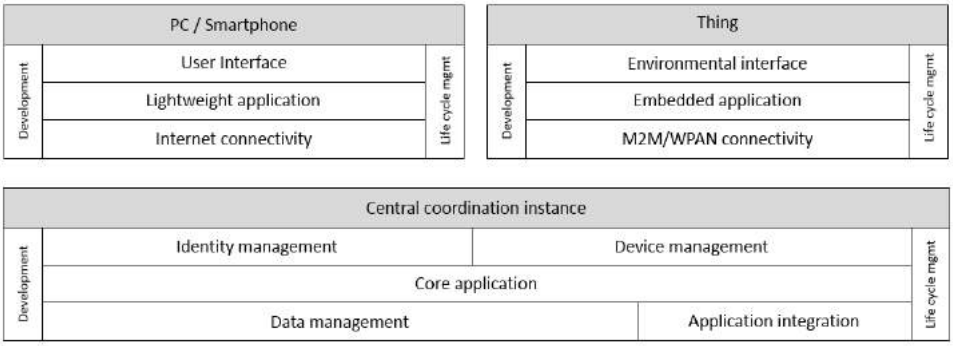

Figure 2: Building blocks of an IoT architecture

5 Summary and Outlook

This paper contributes to the understanding of existing IoT platforms by giving an overview over available solutions and inducing a course grained IoT architecture model depicting fundamental functional building blocks. On the basis of our analysis we can

conclude that while all platforms are marketed under the “IoT platform” label they are indeed quit different in respect to their functionality. Summing up, the solutions can be categorized into three different clusters: (1) Thingssquare and Eclipse focus on things by providing capabilities to develop embedded software being able to communicate on the basis of internet standards. (2) Axeda, Etherios and Xively provide middleware, i.e. functionality that enables the integration, coordination and management of multiple things. (3) Thingworx provides middleware as well as capabilities to rapidly build frontend applications. Finally, the platforms clearly reveal their origin. While Axeda, Eclipse, Etherios, and ThingWorx build upon the classical M2M scenario “central enterprise app receives data from various things”, only Thingssquare and Xively build upon key WSN ideas like self-organizing networks of smart objects.

Our work is limited in that it only builds upon six existing platforms. Therefore, we encourage further research to extend and deepen the scope of our analysis. Furthermore, we encourage a systematic review of scientific work on IoT platforms. Moreover, there is a strong need to better understand the interplay between “new” IoT platforms and existing enterprise application landscapes. Ultimately, we see a new generation of things with unknown computing power and flexibility. This does not only transform things into more mighty general purpose computing devices, but also enables higher programming languages opening up devices for a much large developer community.

Further research could focus on this phenomenon and evaluate how this progress changes the corresponding software development ecosystem and its impact on business solutions.

References

[00a] *Cisco*. URL http://share.cisco.com/internet-of-things.html. - abgerufen am 2013-09-11

[00b] *ABI Research*. URL https://www.abiresearch.com/press/more-than-30-billion-devices-will-wirelessly-conne. - abgerufen am 2013-09-11

[00c] *Gartner’s 2013 Hype Cycle for Emerging Technologies*. URL http://www.gartner.com/newsroom/id/2575515. - abgerufen am 2013-09-11

[00d] *Internet of things: definition of Internet of things in Oxford dictionary (British & World English)*. URL http://oxforddictionaries.com/definition/english/Internet-of-things?q=Internet+of+things. - abgerufen am 2013-09-11

[00e] *Axeda Machine Cloud & M2M Platform*. URL http://www.axeda.com/. - abgerufen am 2013-09-11

[00f] *Digi Device Cloud*. URL http://www.etherios.com/products/devicecloud/. - abgerufen am 2013-09-16

[00g] *Digi Annual Report 2012*. URL http://www.digi.com/pdf/annual-report-2012.pdf. - abgerufen am 2013-09-16

[00h] *Eclipse M2M*. URL http://m2m.eclipse.org/. - abgerufen am 2013-09-16

[00i] *Eclipse M2M Charta*. URL http://www.eclipse.org/org/industry-workgroups/m2miwg_charter.php. - abgerufen am 2013-09-16

[00j] *Thingsquare - Connecting the Internet of Things*. URL http://thingsquare.com/. - abgerufen am 2013-09-11

[00k] *Thingworx*. URL http://www.thingworx.com/. - abgerufen am 2013-09-16

[00l] *Thingworx Building Blocks*. URL http://www.thingworx.com/2011/01/doing-things-that-matter-part-2-how-to-fix-today's-web-using-the-internet-of-things/. - abgerufen am 2013-09-16

[00m] *Xively – Public Cloud for the Internet of Things*. URL https://xively.com/. - abgerufen am 2013-09-11

[Adi00] EL ADI, AHMED: *The Next Big Thing: The Internet of Things*. URL http://blogs.sap.com/innovation/mobile-applications/the-next-big-thing-the-internet-of-things-027162. - abgerufen am 2013-09-11

[AIM10] ATZORI, LUIGI ; IERA, ANTONIO ; MORABITO, GIACOMO: The Internet of Things: A survey. In: *Computer Networks* vol. 54, Elsevier B.V. (2010), Nr. 15, pp. 2787–2805

[Bro03] VOM BROCKE, JAN: *Referenzmodellierung*, 2003

[CJS12] CASTRO, MIGUEL ; JARA, ANTONIO J. ; SKARMETA, ANTONIO F.: An Analysis of M2M Platforms: Challenges and Opportunities for the Internet of Things. In: *2012 Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, Ieee (2012), pp. 757–762 — ISBN 978-1-4673-1328-5

[Cou00] COURTLAND, RACHEL: *Radiation Monitoring in Japan Goes DIY*. URL http://spectrum.ieee.org/tech-talk/energy/environment/radiation-monitoring-in-japan-goes-diy. - abgerufen am 2013-09-11

[DP10] DARGIE, WALTENEGUS ; POELLABAUER, CHRISTIAN: *Fundamentals of wireless sensor networks: theory and practice* : Wiley. com, 2010

[ES04] ESEARCH, S YSTEMS R ; HEVNER, BY ALAN R ; MARCH, SALVATORE T ; PARK, JINSOO ; RAM, SUDHA: DESIGN SCIENCE IN INFORMATION vol. 28 (2004), Nr. 1, pp. 75–105

[LP00] LEMPERT, SEBASTIAN ; PFLAUM, ALEXANDER: Towards a Reference Architecture for an Integration Platform for Diverse Smart Object Technologies, pp. 53–66

[MF10] MATTERN, FRIEDEMANN ; FLOERKEMEIER, CHRISTIAN: Vom Internet der Computer zum Internet der Dinge. In: *Informatik-Spektrum* vol. 33 (2010), Nr. 2, pp. 107–121

[MS03] MATTERN, FRIEDEMANN ; STURM, PETER ; IRMSCHER, K. ; FÄHNRICH, K.-P. (eds.): *Kommunikation in Verteilten Systemen (KiVS)*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2003 — ISBN 978-3-540-00365-6

[Mat00] MATTERN, FRIEDEMANN: Die technische Basis f{ü}r M2M und das Internet der Dinge

[MM12] MUNJIN, DEJAN ; MORIN, JEAN-HENRY: Toward Internet of Things Application Markets. In: *2012 IEEE International Conference on Green Computing and Communications*, Ieee (2012), pp. 156–162 — ISBN 978-1-4673-5146-1

[Pro00] PROFFITT, BRIAN: *Microsoft Sees Its Next Big Thing In The Internet Of Things*. URL http://readwrite.com/2013/06/27/microsoft-stakes-claim-in-internet-of-things#awesm=~oh6VWEdJvleytN. - abgerufen am 2013-09-11

[RN10] RODRIGUES, JOEL J P C ; NEVES, PAULO A C S: A survey on IP-based wireless sensor network solutions. In: *International Journal of Communication Systems* vol. 23, John Wiley & Sons, Ltd. (2010), Nr. 8, pp. 963–981

[Sch02] SCHOENBERGER, CHANA R: The Internet of Things Chips at the checkout counter. In: *Forbes* vol. 169, FORBES MAGAZINE (2002), Nr. 6, pp. 155–161

[Sch98] SCHÜTTE, REINHARD: *Grundsätze ordnungsmäßiger Referenzmodellierung: Konstruktion konfigurations-und anpassungsorientierter Modelle* : Gabler, 1998

[Tre10] TREMBLAY, MONICA CHIARINI: Communications of the Association for Information Systems Focus Groups for Artifact Refinement and Evaluation in Design Research Evaluation in Design Research vol. 26 (2010)

[Wei91] WEISER, MARK: The computer for the 21st century. In: *Scientific American* vol. 265 (1991), pp. 94–104 — ISBN 1558602461