MediaSense – an Internet of Things Platform for Scalable and Decentralized Context Sharing and Control

Theo Kanter
Department of Computer and System Sciences
Stockholm University
SE-164 40 Kista, Sweden
kanter@dsv.su.se

Stefan Forsström, Victor Kardeby, Jamie Walters,
Ulf Jennehag, and Patrik Österberg
Department of Information Technology and Media
Mid Sweden University
SE-851 70 Sundsvall, Sweden
{stefan.forsstrom, victor.kardeby, jamie.walters,
ulf.jennehag, patrik.osterberg}@miun.se

***Abstract*—Research in Internet-of-Things infrastructures has so far mainly been focused on connecting sensors and actuators to the Internet, while associating these devices to applications via web services. This has contributed to making the technology accessible in areas such as smart-grid, transport, health, etc. These early successes have hidden the lack of support for sensor-based applications to share information and limitations in support for applications to access sensors and actuators globally. We address these limitations in a novel open-source platform, MediaSense. MediaSense offers scalable, seamless, real-time access to global sensors and actuators via heterogeneous network infrastructure. This paper presents a set of requirements for Internet-of-Things applications support, an overview of our architecture, and application prototypes created in order to verify the approach in a test bed with users connected from heterogeneous networks.**

Keywords: Internet of Things, Context awareness, Sensors, Actuators, Open source

I. Introduction

Applications that utilize information from sensors to provide more personalized, automatized, or even intelligent behavior to the user are commonly referred to as Internet-of-Things (IoT) applications[1] or Machine-to-Machine (M2M) applications[2]. The reasoning is that these kinds of applications will become widespread when connected to form the IoT where everyday objects can display intelligent behavior. IoT applications can display context-aware behavior, since they may associate a user or an object with information about the surroundings and the current situation[3].

IoT applications exist in a variety of areas, such as environmental monitoring (pollution, earth quake, flooding, forest fire), energy conservation (optimization), security (traffic, fire, surveillance), safety (health care, elderly care), and enhancement of social experience and comfort. IoT applications are projected to have a big impact on how we interact with the world, people and things in the future. In order to enable IoT applications to make intelligent decisions, it is paramount to support timely access to a wide range of information sources on a global scale.

This paper therefore specifies a set of requirements that need to be considered when designing a platform for IoT applications. Related work is presented and described in relation to the requirements, along with limitations in existing solutions, foremost in the lack of support for applications to share information and provide timely access to information from global sensors and actuators. We present an overview of the MediaSense platform and how it addresses the requirements in order to offer scalable, seamless, and real-time access to global sensors and actuators via heterogeneous network infrastructures.

Section II outlines the requirements for IoT applications. Section III puts these requirements in relation to related work. Section IV outlines our solution called the MediaSense platform, whereas section V discusses our current results. Finally, section VI presents the conclusions and the research that still remains to be undertaken.

II. Requirements

Internet-of-Things applications put certain requirements on the supporting architecture and infrastructure. In our analysis, we derived a list of requirements that must be satisfied in order to provide adequate Quality of Service (QoS) and Quality of Experience (QoE) for various types of Internet-of-Things applications. In detail, these requirements are: *a*) **Scalable** – logarithmic or better scaling of communication load in end-points. *b*) **No central point of failure** – fully distributed and several ways to connect to the platform. *c*) **Bidirectional** – capable of communicating with both sensors and actuators. *d*) **Fast** – capable of signaling in real-time between end points. *e*) **Current** – all data retrieved should be the most current values. *f*) **Lightweight** – able to run on mobile devices with limited resources. *g*) **Seamless** – capable of handling multi-NAT traversal, heterogeneous infrastructures, and different end user devices. *h*) **Stable** – reliably handle transient nodes joining and leaving with high churn rates, while making sure that all queries into the platform should return an answer. *i*) **Extensible** – capable

 ISBN: 978-1-61208-193-9

of adding new features and modules without complete redistribution, such as persistence, authentication, and reasoning.

III. Related Work

Related work has mainly been focused on brokering of sensor information on the Internet, via different types of web services. Examples of these typical IoT architectures which utilize centralized servers or cloud-based web services are SenseWeb[4] and Pachube[5]. In detail, these approaches broker the information through a centralized web-service based architecture and thus they do not support requirement a), b), c), d), and e), i.e., **Scalable**, **No central point of failure**, **Bidirectional**, **Fast**, and **Current**.

Cloud-based infrastructures such as CeNSE[6] claim to address the scalability issues. However, cloud-based infrastructures centralize components for authentication and brokering, therefore not satisfying requirements b) **No central point of failure** and e) **Current**.

Project SENSEI[7] proposes a logical architecture for the IoT, but has as yet not provided answers about how sensors are integrated and how such information will be made available in a real-time and scalable fashion, therefore not fulfilling requirements a) **Scalable** and d) **Fast**. Additionally, its architecture contains components which centralize brokering and therefore does not satisfy requirement b) **No central point of failure**.

The SOFIA architecture[8] offers a scalable middleware approach to context aware applications. SOFIA is based on an ontological data model, which can provide filtering of information in relation to related context to create context awareness. The reasoning over an ontological model is inherently slow and the solution there does not satisfy requirement d) **Fast** or e) **Current**.

The COSMOS system[9] is also a middleware for context-centric access control for wireless architectures. COSMOS applies context information to create a novel security model for context-centric access control where mobile agents acts as proxies for mobile devices inside the middleware. This introduces an additional step in data communication which does not satisfy requirement d) **Fast**.

IV. The MediaSense Platform

In response to the shortcomings of earlier solutions in regards to fulfilling the requirements presented in section II, this paper presents a novel architecture for developing applications on the Internet of Things, which satisfies the requirements. The architecture is encapsulated as the MediaSense platform and it is a distributed architecture that enables IoT applications based on sensor and actuator information. An overview of the platform and its components is presented in Figure 1, which show how the platform is distributed over a number of entities connected to the Internet. The figure show how an application that is running a client of the MediaSense platform (a MediaSense instance) communicates with other entities running the platform. A client can acquire sensor and actuator information of the other participants. Furthermore, the platform can act as both a producer and consumer of sensor and actuator information at the same time, enabling bidirectional exchange of context information.

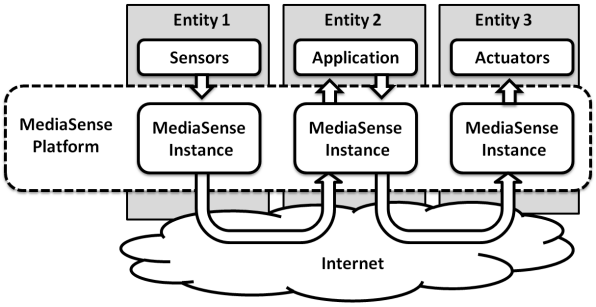

Figure 1. Overview on the function of the MediaSense platform.

A more detailed overview of the whole architecture, including all the layers and components, is shown in Figure 2. This paper will focus on this figure and the remainder of this section will explain the purpose and operation of each layer and their components.

A. Interface Layer

The interface layer is the public interface through which applications interact with the MediaSense platform. The interface layer includes a single component, the MediaSense application interface, which is a generic and standardized Application Programming Interface (API) for developers to build their own IoT applications.

1) MediaSense Application Interface: The purpose of the MediaSense application interface is to provide a single entry point for developers to create applications on top of the MediaSense platform. The interface is thus a standardized API and it provides access to all of the available functionality that the platform provides. Hence, the MediaSense application interface provides many different means of interacting with the MediaSense platform, such as accessing the dissemination core directly or trough any running add-ins, all depending on the applications demands, requirements, and sought after QoE.

B. Sensor and Actuator Layer

The purpose of the sensor and actuator layer is to enable a generalized method to produce information and provide it to the MediaSense platform. The problem is that there exist a large number of different sensors and actuators, which use many different technologies. This needs to be addressed in order to provide the platform with the information and functionality that applications require. The sensor and actuator layer is therefore separated into four components: the actual sensors and actuators, different sensor and actuator

 ISBN: 978-1-61208-193-9

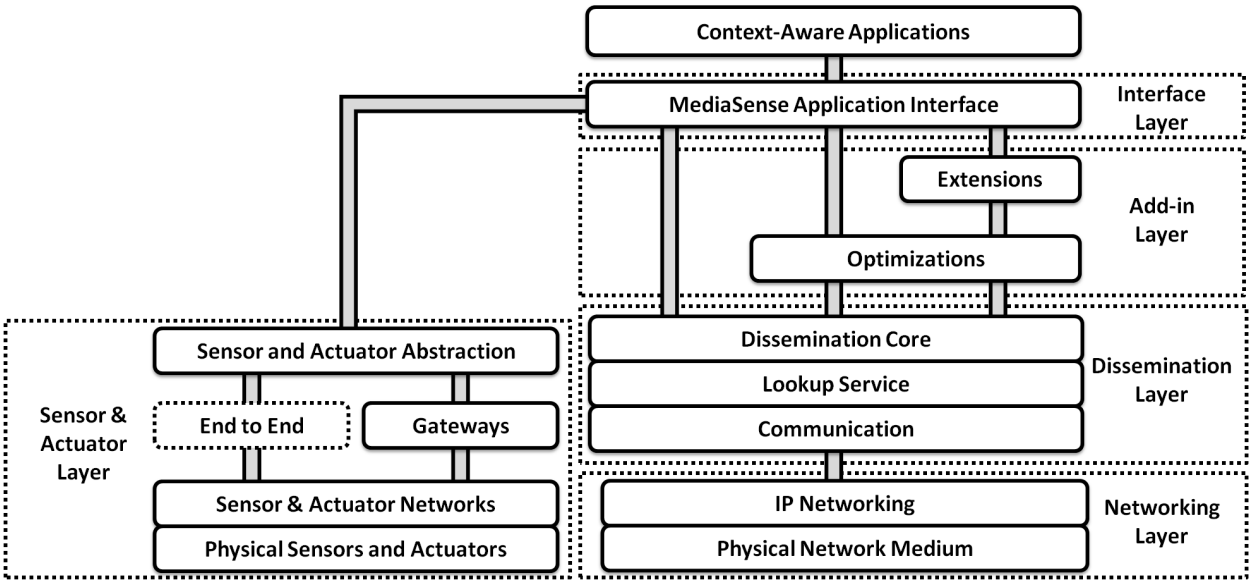

Figure 2. Overview of the MediaSense platform's architecture.

networks, a sensor and actuator gateway, and an abstraction component.

1) Physical Sensors and Actuators: The sensors and actuators provides the actual connection to the physical world. Sensors sense their surroundings and are thus the sources of context information to the whole system. By sensors we mean anything that can produce contextual information, for example, GPS location, temperature, pressure, humidity, and health status. But also context information that are difficult to physically sense, such as name, favorite food, mood, preferences, etc. By actuators we mean any type of object which can access the physical world and perform some form of actuation in it. Typical actuators include controlling devices such as light switches and heating temperature settings. But actuators can also include any type of shared resource made available to interact with via the MediaSense platform, such as shared data storage.

2) Sensor and Actuator Networks: Sensors and actuators are usually connected through some form of sensor or actuator network. These are for example, wireless sensor networks (WSN) or wireless sensor/actuator networks (WSAN). The purpose of these networks is to gather data from many connected physical sensors through a network of sensors and actuators. WSN/WSAN is used in order to achieve larger area coverage, higher quality of service, lower energy consumption, and cheaper hardware.

A sensor and actuator network commonly designate a node to coordinate access to and from the outside world, it is not uncommon for such a node to be more advanced in terms of processing power or battery reserves. Based on the properties of the network, this will be the node that communicates upward to either a gateway or directly to the sensor and actuator abstraction.

3) Gateways: As a consequence of the wide range of protocols and technologies used in sensors and actuator networks, gateways are sometimes required. In detail, a gateway translates the sensor specific network technology into a common communication protocol. Thus a gateway mediates communication with each specific sensor and actuator type. Therefore a separate gateway has to be built for each new sensor or actuator network that wants to connect into the MediaSense platform. The gateway then provides access to the sensors and actuators regardless of the underlying technology used by the sensor and actuator networks.

However, if the sensor and actuator network has the ability to directly talk to the abstraction component, the gateway can be ignored for that particular network. This is denoted as the "End to End" part inside the sensor and actuator layer. These are for example IPv6 capable sensor and actuator networks [10] or other types of networks with extend capacities and computational power.

4) Sensor and Actuator Abstraction: The abstraction component provides a standardized method of interaction with all sensors and actuators. It abstracts all sensors and actuators into a generalized and standardized format, which is connected into the MediaSense platform through the application interface as any other application. It thus provides the platform with access to all types of sensors and actuators in one of two ways. Either through a gateway that translates any communication protocol and access method used by the sensor and actuators network, or by directly communicating with the sensor and actuator network if the sensor and actuator network is powerful enough.

 ISBN: 978-1-61208-193-9

C. Add-in Layer

The purpose of the add-in layer is to enable developers to add optional functionality and/or optimization algorithms to the MediaSense platform. An add-in can be used in order to make the MediaSense platform meet specific application requirements, sought after quality of experience, or available capacity in regards to computational power and bandwidth. Thus the add-in layer manages different extensible and pluggable add-ins, which can be loaded and unloaded in runtime when needed. The add-in layer can include any number or type of add-ins, but they are divided into two categories, optimization components and extension components. Whereas the optimization components offer ways of optimizing the behavior and functionality of the system, and the extension components enables extended functionality which applications might demand.

1) Extensions: The extension components provide add-ins for enabling extended functionality, such as context-awareness and reasoning, in the MediaSense platform. These extension add-ins can for example include, logical context objects, semantics, reasoning, ranking of context information, search engines, query languages, and context agents.

2) Optimizations: The optimization components provides add-ins for optimizing the MediaSense platform in many different forms. These optimizations add-ins can for example include clustering of information, caching, persistence, intelligent routing, and decision making to determine when to optimally send data.

D. Dissemination Layer

The dissemination layer enables dissemination of information between all entities that participate in the system and are connected to the IoT. In detail, the Distributed Context eXchange Protocol (DCXP) [11] is used. DCXP offers reliable communication among entities that have joined a peer-to-peer network, which is used to enable exchange of context information in real-time. The operation of the DCXP includes resolving of so called Universal Context Identifiers (UCI) and subsequently transferring context information directly with a resolved entity. Therefore, the dissemination layer includes three components, a dissemination core, a lookup service, and a communication system. The dissemination core exposes the primitive functions provided by DCXP, the lookup service find and resolve other entities who has joined the system, and the communication component abstracts a transport layer communication.

1) Dissemination Core: DCXP offers primitive functions for publishing, retrieving, and transferring information in a peer-to-peer manner, as well as joining and leaving the peer-to-peer network. Hence, it is the dissemination core that exposes these primitive functions to the above layers, thus making these services available to the MediaSense platform. Furthermore the dissemination core hides the underlying lookup service and communication technology from the above layers. Thus allowing different choices for lookup service and communication technology without any changes to the other layers.

2) Lookup Service: The lookup service provides the means of resolving UCI's to find the location of a sought after piece of information or entity. The lookup service can be implemented in a number of different ways, for example as a distributed hash table, distributed graph, or cloud server. Examples of already tested and evaluated systems are Chord and PGRID, which was done in [12] and [13], respectively.

3) Communication: The communication component offers the possibility to exchange the communication protocol of the dissemination layer and thus of the whole MediaSense platform. It also makes it possible to provide multiple concurrent communication protocols, such that the components can request different quality of service based on their chosen communication protocol. Examples of possible communication protocols are: TCP, UDP, Reliable-UDP, and Stream Control Transmission Protocol (SCTP).

E. Networking Layer

The MediaSense platform is designed to operate over heterogeneous infrastructure, including wireless and mobile. The purpose of the networking layer is to connect different entities over current IP based networking infrastructure. In general the networking layer has two components, an IP network and a physical network medium.

1) IP Networking: The IP network component is the IP endpoint for a particular entity, which is running an instance of the MediaSense platform. The IP networking components thus provides the ability to communicate with other entities on the Internet, regardless of type the type of connection. In detail, this can include both IPv4 and IPv6 networks.

2) Physical Network Medium: The physical network medium component denotes that the MediaSense platform is agnostic of the underlying infrastructure. Hence, the MediaSense platform can run over heterogeneous networks and via different types of physical infrastructures. This includes different technologies such as Ethernet, 802.11 b/g/n, and other variants of mobile broadband and fiber optic networks.

V. Results and Discussion

The current results include development and launching of an open source development website (www.mediasense.se) for the MediaSense platform. This website will act as a portal for all developers who want to utilize the MediaSense platform in their applications. The MediaSense platform is provided free and under an open source license, in order to make it available for anyone to use.

A. Verification

Initial testing and evaluation of the open source platform has been conducted with users in a testbed with fixed and mobile access to the Internet. Proof-of-concept applications

 ISBN: 978-1-61208-193-9

(a) Object tracking (b) Intelligent home (c) Health monitoring (d) Energy consumption (e) Energy profiling

Figure 3. Examples of applications using the MediaSense platform.

have been built both on set-top boxes and smartphones, in order to show that the MediaSense platform can be applied in a wide range of scenarios, e.g., health care, intelligent home, object tracking, and social applications. Figure 3, shows some of these proof-of-concept applications that use the MediaSense platform. From left to right they represent object tracking (for tracking sensor enabled objects), intelligent home automation (for interacting with the intelligent home), health monitoring (for medical status and alerts), energy consumption (for smart energy monitoring), and energy profiling (for energy awareness). The initial testing and evaluation indicates that the MediaSense platform is on par with UDP traffic over mobile Internet access, which is demanded by real-time applications.

The application shown in Figure 3a addresses object tracking and verifies the importance of requirements a) scalable, b) **No central point of failure**, f) **Lightweight**, and h) **Stable**, due to the large number of transient entities and objects.

The application in Figure 3b targets the intelligent home and verifies the importance of requirements c) **Bidirectional** and g) **Seamless**, due to that it both handles sensors and actuators, as well as different networks.

Figure 3c shows a health monitoring application that verifies the importance of requirements d) **Fast**, e) **Current**, h) **Stable**, and i) **Extensible**. This is due to the highly sensitive data which requires secure authentication and real-time delivery to minimize delay of critical health care.

The application in Figure 3d targets energy consumption and verifies the importance of requirements d) **Fast**, and e) **Current**, requiring a steady stream of current data to monitor the changes in energy consumption.

Lastly, the energy profiling application in Figure 3e verifies the importance of requirements c) **Bidirectional**, d) **Fast**, e) **Current**, and i) **Extensible**, because profiling also has to include reasoning in order to create energy awareness in the application.

B. Addressing the Requirements

The presented architecture and platform can support all of the posed requirements in section II. In detail, the requirements are addressed as follow.

a) **Scalable** is addressed by using a scalable lookup service in the platform. For example, the Chord and P-grid solutions which scale logarithmic with the amount of entities in the whole system.

b) **No central point of failure** is addressed by using a fully distributed system in the dissemination layer, without any centralized component. A fully distributed system is needed due other requirements, such as fast, current, and scalability.

c) **Bidirectional** is addressed by allowing two-way communication in the dissemination layer and accepting both sensors and actuators in the system.

d) **Fast** is addressed by using a distributed lookup service with logarithmic, or better, lookup delays that utilize a communication protocol which can provide real-time communication. This, in order to support real-time applications, and that no obsolete information is being considered in application logic.

e) **Current** is addressed by using a peer-to-peer system as the dissemination layer. This because a peer-to-peer system is communicating with the source, and thus is always proving the most current value.

f) **Lightweight** is addressed with the possibility of choosing lightweight components and only loading the required add-ins. For example only loading reasoning, semantics, etc., when needed. Thus avoiding computational heavy components if they are not required. Our platform is therefore able to run on mobile devices such as smartphones.

 ISBN: 978-1-61208-193-9

g) **Seamless** is addressed by having the platform agnostic of the underlying infrastructures. The networking layer supports heterogeneous IP-based infrastructures if the lookup service and commutation protocol can penetrate Network Address Translation (NAT).

h) **Stable** is addressed by utilizing a stable lookup service. The lookup service must thus support transient nodes leaving and joining, as well as intermittent disruptions and unexpected disconnections. Furthermore, the reliability is solved by using a reliable lookup service and commutation protocol. This is also required in order to support real-time applications, with application logic depending on fresh information. Thus the lookup service and the communication protocol must provide reliable services.

i) **Extensible** is addressed by allowing multiple add-ins to be dynamically loaded on demand. Which is solved in the add-in layer, and is a prerequisite for other requirements, such as persistence and authentication. Persistence can be addressed by adding such an add-in to the platform. Relevant data can be stored based on importance, in order to provide better lookup, and reliability. Furthermore, controlling the reach of information can be addressed by creating an add-in that can authenticate entities and encrypt the data.

VI. Conclusion and Future Work

The contributions of this paper begins in Section II with the specification of a set of requirements that need to be considered when designing a platform for IoT applications. Then existing related work is evaluated in Section III, where several of these approaches address some of the requirements, but they all have their limitations. We therefore propose the MediaSense platform and its components in Section IV. It provides a distributed IoT infrastructure that offers scalable, seamless, real-time access to global sensors and actuators. The sensors and actuators are connected via devices that act as end-points in the peer-to-peer based infrastructure. Applications running locally on such end-point devices can thereby access information from sensors, and control actuators, connected to any other end-point. The proposed MediaSense platform fulfills all the stated requirements, as shown by the demo applications and descriptions in Section V. Further, the platform is provided for free, under an open source license.

Current efforts are directed toward interfacing between the connected-things infrastructure and the world of our experience, through extending the platform with a semantic layer and datamining capabilities, decision making, reasoning, and optimizations. Moreover, we are working on a generic integration with IoT and cloud infrastructures in new projects.

Acknowledgment

This research is partially funded by VINNOVA, the Swedish Governmental Agency for Innovation Systems. The authors also want to thank partners from industry and academia, in particular Acreo AB, Ericsson, and Stockholm University.

References

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[2] G. Wu, S. Talwar, K. Johnsson, N. Himayat, and K. Johnson, "M2m: From mobile to embedded internet," *Communications Magazine, IEEE*, vol. 49, no. 4, pp. 36–43, 2011.

[3] J. Hong, E. Suh, and S. Kim, "Context-aware systems: A literature review and classification," *Expert Systems with Applications*, vol. 36, no. 4, pp. 8509–8522, 2009.

[4] W. Grosky, A. Kansal, S. Nath, J. Liu, and F. Zhao, "Senseweb: An infrastructure for shared sensing," *Multimedia*, vol. vol. 14, pp. pp. 8–13, 2007.

[5] O. Haque. Pachube. [Online]. Available: www.pachube.com

[6] Central Nervous System for the Earth (CeNSE). Hewlett Packard. [Online]. Available: http://www.hpl.hp.com/research/intelligent_infrastructure/

[7] A. Gluhak, M. Bauer, F. Montagut, V. Stirbu, M. Johansson, J. Vercher, and M. Presser, "Towards an architecture for a real world internet," *Towards the Future Internet: a european research perspective*, pp. 313–324, 2009.

[8] A. Toninelli, S. Pantsar-Syväniemi, P. Bellavista, and E. Ovaska, "Supporting Context Awareness in Smart Environments: A Scalable Approach to Information Interoperability," in *Proceedings of the International Workshop on Middleware for Pervasive Mobile and Embedded Computing*, 2009.

[9] P. Bellavista, R. Montanari, and D. Tibaldi, "Cosmos: A Context-Centric Access Control Middleware for Mobile Environments," in *Mobile Agents for Telecommunication Applications*, 2003, pp. 77–88.

[10] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki-a lightweight and flexible operating system for tiny networked sensors," in *Local Computer Networks, 2004. 29th Annual IEEE International Conference on*. IEEE, 2004, pp. 455–462.

[11] T. Kanter, S. Pettersson, S. Forsstrom, V. Kardeby, R. Norling, J. Walters, and P. Osterberg, "Distributed context support for ubiquitous mobile awareness services," in *Communications and Networking in China, 2009. ChinaCOM 2009. Fourth International Conference on*, Aug. 2009, pp. 1–5.

[12] T. Kanter, P. Österberg, J. Walters, V. Kardeby, S. Forsström, and S. Pettersson, "The MediaSense Framework," in *Proceedings of Fourth IARIA International Conference on Digital Telecommunications (ICDT)*, Colmar, France, July 2009.

[13] J. Walters, T. Kanter, and E. Savioli, "A Distributed Framework for Organizing an Internet of Things," in *the 3rd International ICST Conference on Mobile Lightweight Wireless Systems*, 2011.

 ISBN: 978-1-61208-193-9