IoT-A and FIWARE: Bridging the Barriers between the Cloud and IoT Systems Design and Implementation

Alexandros Preventis, Kostas Stravoskoufos, Stelios Sotiriadis and Euripides G. M. Petrakis
Intelligent Systems Laboratory Department of Electronic and Computer Engineering, Technical University of Crete (TUC), Chania, gr-73100, Greece

Keywords: Cloud Computing, Internet of Things, IoT-A, FIWARE, Cloud Service Interoperability.

Abstract: Today, IoT systems are designed and implemented to address specific challenges based on domain specific requirements, thus not taking into consideration issues of openness, scalability, interoperability and use-case independence. As a result, they are less principled, lacking standards, vendor oriented and hardly replicable since the same IoT architecture cannot be used in more than one use-cases. To address the fragmentation of existing IoT solutions, the IoT-A project proposes an architecture reference model that defines the principles and standards for generating IoT architectures and promoting the interoperation of IoT solutions. However, IoT-A addresses the architecture design problem, and does not focus on whether existing cloud platforms can offer the tools and services to support the implementation of IoT-A compliant IoT systems. In this work we propose an architecture based on IoT-A that focuses on the FIWARE open cloud platform that in turn provides the building blocks of future Internet applications and services. We further correlate FIWARE and IoT-A projects to identify the key features for FIWARE to support IoT-A compliant system implementations.

1 INTRODUCTION

Over the recent years, Cloud Computing and IoT have been rapidbly advancing as the two fundamental technologies of the Future Internet concept. Although both focus on different domains and have evolved independently, various works integrate common solutions and identify significant advantages. These are as follows:

- The cloud's virtually unlimited resource pool could be paired with the exponentially growing demands of the IoT. As enormous amounts of data is generated, cloud offers the requiring facility to store and processes as well as to access for multiple third party users.
- The cloud offers an elastic environment that can scale on demand and according to the needs of the user, thus allowing the creation of more flexible IoT systems that can effectively adapt to their changing requirements.
- High availability is crucial for any IoT system and can be currently guaranteed by modern cloud providers.
- Cloud computing can bridge the gap between devices and applications by abstracting IoT management and composition services, acting as and intermediate layer and by hiding the complexities and the peculiarities of the IoT systems.

Thus, it is clear that Cloud Computing and IoT forming an ideal environment for massive data storage and manipulation. However, the lack of standardization in the IoT domain has resulted in the fragmentation of current IoT systems. Existing IoT architectures have been designed and implemented to address specific challenges with specific requirements, not taking into consideration issues of openness, scalability, interoperability and use-case independence. Considering the above and due to the vast variation of technologies in the IoT domain, current solutions are vertically closed forming many “Intranets of Things” rather than an “Internet of Things”.

To address this problem, the IoT-A project proposes a Reference Model and Architecture (RMA) defining the principles and guidelines for generating IoT architectures, providing the means to connect vertically closed systems. The adoption of the IoT-A promotes the interoperation of IoT solutions by enabling interoperability as follows:

- At the communication layer to support the co-existence of various communication technologies (existing or emerging).

Preventis, A., Stravoskoufos, K., Sotiriadis, S. and Petrakis, E.
IoT-A and FIWARE: Bridging the Barriers between the Cloud and IoT Systems Design and Implementation.
In *Proceedings of the 6th International Conference on Cloud Computing and Services Science (CLOSER 2016) - Volume 2*, pages 146-153
ISBN: 978-989-758-182-3

- At the service layer, thus ensuring smooth integration into the service layer of Future Internet applications.

Moreover, IoT-A compliant architectures assure that generated knowledge will be modular and re-usable across domain or use-case specific boundaries. In this work we propose to design FIWARE IoT systems deriving from IoT-A. We study FIWARE Generic Enablers (GEs) and investigate whether the proposed architecture can be implemented on FIWARE by testing FIWAREs ability to support IoT-A compliant IoT system implementations. We compare the functionality of GEs against IoT-A requirements and guidelines and point out weaknesses and missing points along with suggestions.

Having said that, we present the background and related work in Section2. In Section 3 we discuss the proposed architecture and then the correlation between IoT-A and FIWARE. Finally, in Section 4, we state our conclusions and discuss future research issues.

2 BACKGROUND WORK

This section presetns the IoT-A project that introduces standards, guidelines and directions for generating IoT architectures. We also discuss FIWARE and its relationship with IoT systems. IoT-A introduces a Reference Model and a Reference Architecture that are discussed bellow.

2.1 Architectural Reference Model

ARM is an abstract framework that comprises of a minimal set of unifying concepts, axioms and relationships for understanding significant relationships between the entities of the IoT domain. It consists of several sub-models that set the scope for the IoT design space:

- IoT domain model: It is a top-level description of the IoT domain that introduces the main concepts of the IoT like Devices, IoT Services and Virtual Entities (VEs), and it also relations between these concepts.
- IoT Information model: It defines the structure (e.g. relations, attributes) of IoT related information in an IoT system on a conceptual level.
- IoT Functional model: It identifies groups of functionalities, grounded in key concepts of the IoT Domain Model. Functionality Groups (FGs) provide the functionality for interacting with the instances of these concepts or managing the information related to the concepts (e.g., information about Virtual Entities or descriptions of IoT Services). FGs use the Information Model for structuring any information they manage.
- IoT Communication model: Introduces concepts for handling the complexity of communication in heterogeneous IoT environments. Constitutes one FG in the IoT Functional model.
- Trust,Security and Privacy (TSP) model: Introduces relevant functionality, interdependencies and interactions. The TSP model is also one FG in the IoT functional model.

2.2 Reference Architecture

The IoT-A Reference Architecture is designed to allow the generation of many, potentially different, compliant IoT architectures that can be tailored to specific use cases. The Reference architecture is based on the concepts of architectural views and architectural perspectives which are introduced in~\cite{rozanski-woods-viewpoints} and~\cite{rozanski-woods-perspectives} respectively. Architectural views derive from the concerns of stakeholders (i.e., people, groups, or entities with an interest in the realization of the architecture) in actual system design (e.g., developers, users, system administrators etc.).

Views are defined as ``representations of one or more structural aspects of an architecture that illustrate how the architecture addresses one or more concerns held by one or more of its stakeholders"~\cite{woods211} and are used to achieve the decomposition of the architectural description with each view addressing one aspect of the architectural structure. The IoT-A reference architecture comprises of the following views:

- Physical Entity View It describes all physical entities and their relations (e.g., sensors, actuators, environment measurements) in an IoT system. This view is not covered by IoT-A because it is use-case independent.
- IoT context View: It provides context information about physical entities such as the Physical Entity View, this view is also not covered by IoT-A as it is use-case independent.
- Functional View: It describes the system’s runtime Functional Components and, also, their responsibilities, default functions, interfaces and primary interactions. The Functional View

derives from the Functional Model and reflects the developers perspectives on the system.

- Information View: It is used to generate an overview about static information structure and dynamic information flow. It is based upon the IoT information model.
- Deployment View: It explains the operational behaviour of the functional components and the interplay of them.
- Information View: It shows how the information flow is routed through the system and what requests are needed to query for or to subscribe to information offered by certain functional components.

Figure 1 demonstrates the relationship between IoT-A architectural views and model in the process of designing an actual system architecture.

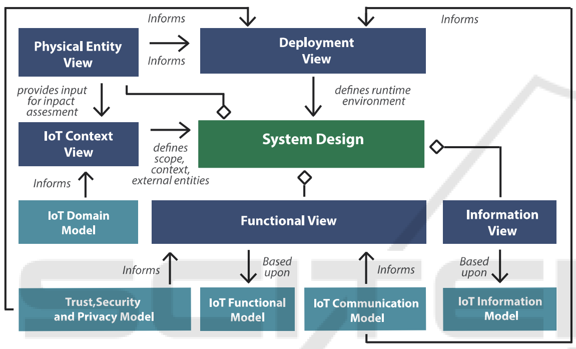


Figure 1: Relationship of IoT-A architectural views and models.

Along with architectural views, IoT-A also introduces architectural perspectives in IoT system architectures that are collections of activities, checklists, tactics and guidelines to guide the process of ensuring that a system exhibits a particular set of closely related quality properties (i.e., such as performance, security, availability etc.) that require consideration across a number of the system's architectural views~\cite{rozanski-woods-perspectives-2005}. Perspectives are "applied" to views to ensure acceptable qualities and guide changes where required. The quality properties that have been identified by IoT-A as the most important in the IoT domain are:

- Evolution and Interoperability to enable communication between devices and services.
- Availability and Resilience as the ability of the system to stay operational and handle failures.
- Trust, Security and Privacy to handle such parameters.
- Performance and Scalability related to the monitor of the system's state and configuration of perfomance thresholds.

2.3 FIWARE Platform

Today, FI-WARE develops cloud services to build novel FI applications that use remotely accessible modules (GEs) on a pay on demand model. FI-WARE is a leading cloud vendor that offers open specification for services that could be spread at different geographically locations (and hosted in various nodes-namely XIFI nodes), and are available for utilization over the Internet. In FI-WARE, the cloud model defines three main roles namely as the service consumer (the developer), the service provider (the FI-WARE open specification) and the service creator (the GE implementer).

Traditionally, the service creator generates a service that is hosted by FI-WARE and represents the user requirements. FI-WARE is an innovative, open cloud-based infrastructure for cost-effective creation and delivery of Future Internet applications and services named GEs. GEs are considered as software modules that offer various functionalities along with protocols and interfaces for operation and communication. These include the cloud management of the infrastructure, the utilization of various IoT devices for data collection and the provision of APIs (e.g. tools for data analytics) and communication interfaces (e.g., gateways, messaging etc.). GEs are implementations of open specifications of the most common functionalities that are provided by FI-WARE and are stored in a public catalogue, thus developers could easily browse and select appropriate APIs to use.

3 OPPORTUNITIES FROM THE CORRELATION BETWEEN FIWARE AND IoT-A

The role of Reference architectures such as IoT-A in the process of creating actual systems is to provide the key building blocks for the generation of the system's architecture.

The generated architecture can then be used to guide the process of the actual system implementation. Reference architectures are application independent being more abstract than architecture which are designed with specific constraints and requirements in mind. On the other hand, architectures, which are generated by extracting essentials (parts of existing architectures, mechanisms, standards) from references

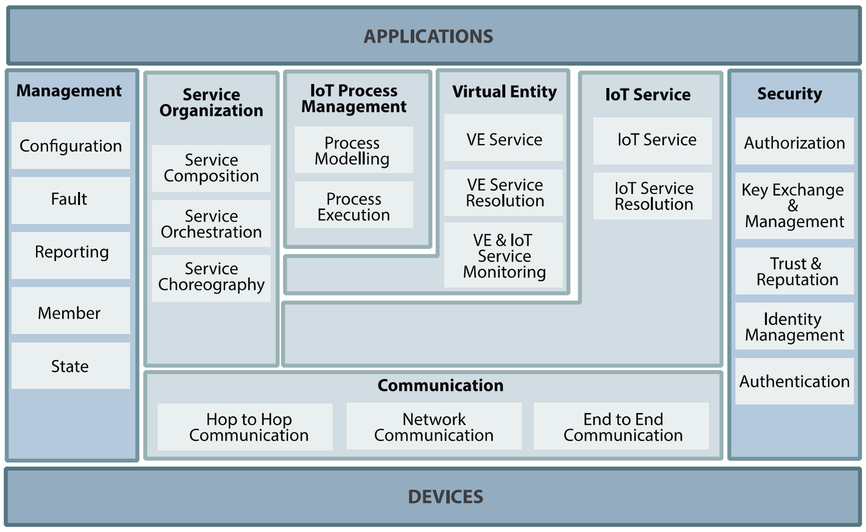


Figure 2: The proposed architecture.

architectures, should be application specific but platform independent allowing various implementations across different platforms. Finally actual system implementations are platform specific relying on platforms such as FIWARE. The relationship between Reference architectures (IoT-A), architectures and actual system implementations is illustrated in Figure 3.

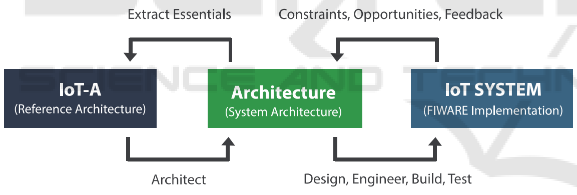


Figure 3: Relationship between Reference architecture(IoT-A), architecture and the actual system implementation.

In the following we propose an architecture deriving from IoT-A and decide whether this architecture can be implemented on FIWARE. Although architectures should be composed by several views, reflecting the concerns of the different stakeholders on the system, in our study we focus on the Functional View of IoT-A adopting the developers' perspective as our main concern on the system is the implementation.

3.1 Proposed Architecture

The proposed architecture is illustrated in Figure 2. The architecture is composed by seven Functionality Groups (FGs) each of these consisting of several Functional Components (FCs). Each FCs provides different and unique functionality and is able to communicate with the others in a distributed environment. In the following we present the FGs and discuss the functionality of their FCs.

3.1.1 IoT Process Management FG

This FG provides the functional concepts necessary to conceptually integrate the IoT world into traditional (business) processes. Applications can utilize the tools and interfaces defined for the IoT Process Management FG in order to stay on the (abstract) conceptual level of a (business) process, while, at the same time, making use of IoT-related functionality without the necessity of dealing with the complexities of IoT Services. It consists of two FCs:

The Process Modelling FC provides the tools required for modelling IoT-aware business processes that will be serialised and executed in the Process Execution FC which is responsible for deploying process models to the execution environments. Also, the Process Execution FC utilizes components of the Service Organization FG to align application requirements with service capabilities.

3.1.2 Service Organisation FG

Acts as a communication hub between several other Functional Groups by composing and orchestrating Services of different levels of abstraction. The Service Orchestration FC resolves the IoT Services that are suitable to fulfil service requests coming from the Process Execution FC or from Users while the Service Composition FC is responsible for creating services with extended functionality by composing IoT services with other services. Finally, the Service

Choreography FC offers a broker that handles Publish/Subscribe communication between services.

3.1.3 Virtual Entity FG

Provides all the necessary functionality for the interaction of VEs with the IoT system, for VE look-up and discovery and for providing information concerning VEs. The VE Resolution FC allows associations between VEs and IoT services providing discovery and look-up functionalities for these associations while VE & IoT Service Monitoring FC is responsible for automatically finding new associations based on service descriptions and information about VE's. Finally, the VE Service FC handles entity services (for example, services that provide access to an entity via operations that enable reading and/or updating the value(s) of the entity's attributes).

3.1.4 IoT Service FG

This FG contains IoT services as well as functionalities for discovery, look-up, and name resolution of IoT Services. The IoT Service FC exposes IoT resources (e.g., information retrieved from sensors) making them accessible to other parts of the IoT system. This FC can also be used for delivering information to a resource in order to control actuator devices or to configure the resource (e.g., manage access control and permissions on the resource). IoT Services can be invoked either in a synchronous way by responding to service requests or in an asynchronous way by sending notifications according to subscriptions. Finally, the IoT Service Resolution FC provides service discovery and resolution functionalities along with service description management capabilities.

3.1.5 Communication FG

The Communication FG abstracts the interaction schemes derived from the variety of communication technologies in IoT systems in order to provide a common interface to the IoT Service FG. The Hop To Hop Communication FC provides the first layer of abstraction from the device's physical communication technology, the Network Communication FC enables communication between networks and, finally, the End to End Communication FC offers reliable transfer, transport and, translation functionalities, proxy/gateway support and setting configuration parameters when the communication crosses different networking environments.

3.1.6 Security FG

This is the FG that is responsible for security and privacy matters in IoT-A-compliant IoT systems. The Authorization FC is used to provide access control and access policy management while the Authentication FC is used for user and service authentication. The Identity Management FC addresses privacy by issuing and managing pseudonyms and accessory information to trusted subjects so that they can operate anonymously and the Key Exchange and Management (KEM) FC enables secure communications ensuring integrity and confidentiality by distributing keys upon request in a secure way. Finally, Trust and Reputation FC collects user reputation scores and calculates service trust levels.

3.1.7 Management FG

The Management FG is responsible for the composition and tracking of actions that involve the other FGs. The Configuration FC is responsible for initialising the system's configuration (e.g., gathering applying configurations from FC's and Devices). It is also responsible for tracking configuration changes and planning for future extensions of the system. The Fault FC is used to identify, isolate, correct and log faults that occur in the IoT system. The Member FC is responsible for the management of the membership of any relevant entity (FG, FC, VE, IoT Service, Device, Application, User) to an IoT system working in cooperation with the Authorisation and Identity Management FCs of the Security FG. The Reporting FC generates reports about the system and, finally, the State FC can change or enforce a particular state on the system by issuing a sequence of commands to the other FCs. Moreover, it is constantly monitoring the state of the system notifying subscribers about changes.

3.2 FIWARE Implementaion

In the following we show how the proposed architecture can be realized and implemented in FIWARE using FIWARE GEs. We go through each FG of the architecture and show which GEs implement can supply the functionality of FCs pointing out missing points and weaknesses and making suggestions.

3.2.1 Service Organization FG

The functionality of the Service Choreography FCThe is encapsulated by the Orion Context Broker

GE. The broker offers Publish/Subscribe capabilities, providing NGSI9/10 interfaces, allowing clients to do several operations like register context producer applications, update context information and get notifications when context information changes take place or with a given frequency. Finally the broker allows also to query context information and stores context information update so queries are resolved based on that information. The functionalities of the other two FCs of this FG are not covered by any FIWARE GE.

3.2.2 Service Organization FG

The functionality of the two FCs in the IoT Service FG is partially covered by the IoT Discovery, the IoT Broker and the Backend Device Management (IDAS) GEs. The IoT Discovery GE allows context producers to register their IoT Objects in linked-data format and in turn allows context consumers to discover them using a set of search techniques. Focuses on semantically-annotated IoT descriptions to supports querying via SPARQL. Although the IoT Discovery GE offers a device discovery mechanism, it does not offer a service resolution mechanism thus, covers only partially the functionality of the IoT Service Resolution FC. A service discovery and resolution mechanism should be added to this GE. The IoT Broker GE and the Backend Device Management GE encapsulate both the functionality of the IoT Service FC each of them covering different parts of it.

The IoT Broker GE retrieves and aggregates information from IoT devices acting as a middleware component that separates IoT applications from devices. The GE is based on NGSI and closes the gap between information-centric applications and device-centric IoT installations by communicating simultaneously with large quantities of IoT gateways and devices in order to obtain exactly the information that is required by the running IoT applications. By using the broker all IoT devices can be abstracted to be viewed as NGSI entities on a higher level hiding the complexity of the Internet of Things from developers. The IoT broker does not currently support delivering information to resources in order to control them or configure them. This is supported by the Backend Device Management GE. The Backend Device Management GE, provides an API for M2M application developers and a device communication API for device (sensor/actuators/gateways) communication which currently implements the SensorML and Lightweight SensorML following protocols.

The GE Collects data from devices (status etc) and translates them into NGSI events available at a context broker. Application developers can use data and send commands through the broker. An open source Reference Gateway called "FIGWAY" is also offered for Raspberry PI and Z-wave devices. Although this GEs complements the IoT broker towards supporting the specifications of the IoT Service FC it does not make a clear separation of the layers as they are defined by IoT-A and it's functionality intervenes into the communication layer. Moreover, The GE currently works only with FIGWAY meaning that it can collect data and manage only specific devices controlled by the Raspberry PI and Z-wave gateways. It should be extended to support any kind of gateway and support all protocols other than CoaP (e.g., XMPP, MQTT, REST).

3.2.3 Communication FG

The Protocol Adapter GE handles low level communication between devices and the rest of the IoT system encapsulating the functionality of the Hop to Hop Communication FC. The GE handles communication of devices using CoaP over 6LowPan protocol but currently supports only the IBM Mote hardware running the Moterunner operating system. Although it does cover the required functionality, the protocol Adapter is currently working with specific devices(Mote) over a specific platform (Moterunner) and, thus, needs to be extended to be compatible with all available devices and platforms.

The Network Communication FC functionality is partially covered by the Gateway Data Handling GE which is designed to provide a common access in real time to all data. Using a simple local XML storage, the GE locally stores relevant processed data and offers Filtering, aggregating and merging real-time data from different sources.

Transforms data into events in a way that applications need only to subscribe to events (data) which is relevant to them. The GE offers more functionality of what the Network Communication FC specifies but goes against the clear separation of layers as the IoT-A specification does not contemplate any data processing and handling in the Communication FG.

3.2.4 Security FG

In the Security FG the KeyRock Identity Management (IDM) GE provides secure and private authentication from users to devices, networks and services, authorization, user profile management and privacy-preserving disposition of personal data,

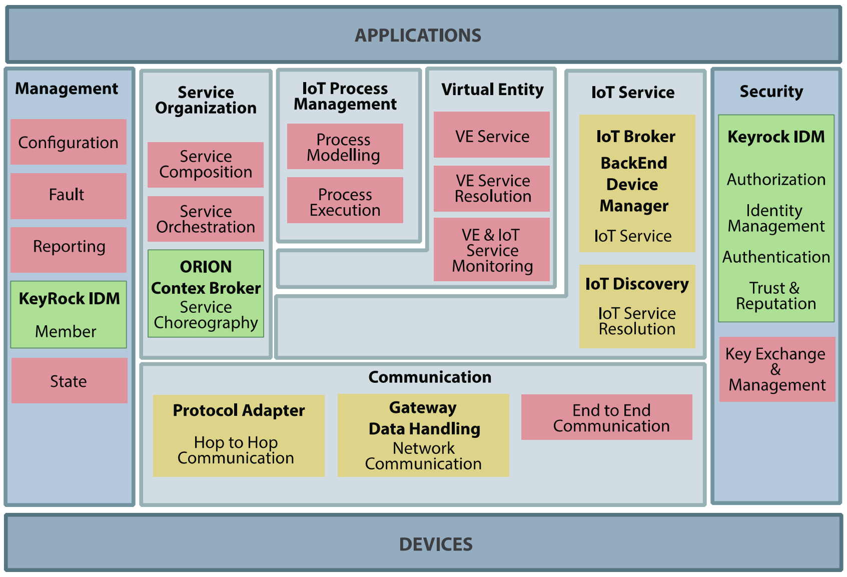


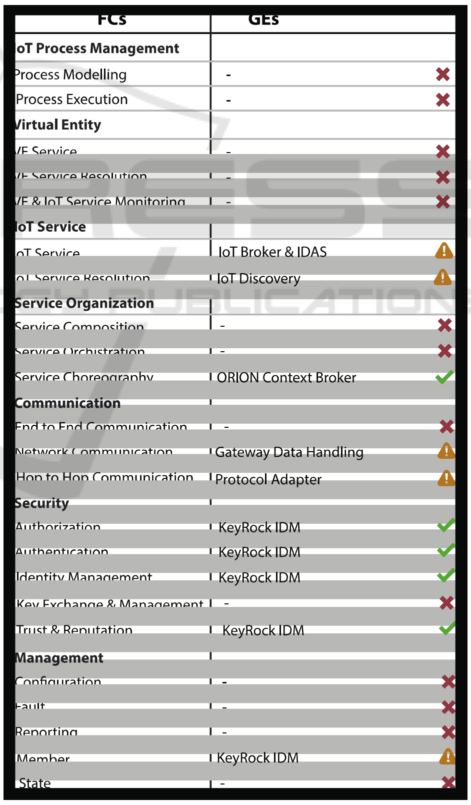
Figure 4: FIWARE implementation.

encapsulating the functionality of four out of five FCs. The Key Exchange and Management FC functionality is not currently supported. Also, as the KeyRock architecture encapsulates the functionality of four out of five FCs, the architecture becomes less modular as there is no clear separation of modules.

3.2.5 Management FG

The functionality of the Member FC is covered by the KeyRock GE which handles all entities membership in the IoT system. This way KeyRock encapsulates functionality of FCs in two different layers of the architecture demoting modularity and task separation.

The functionality of the remaining Management FG FCs is not offered by any FIWRARE GE. In Figure 4 we illustrate the architecture substituting FCs with FIWARE GEs. Green colour denotes that the GE can effectively providing the specified functionality of the FC, yellow that it does so partially and red denotes there is currently no GE that can provide the functionality of the FC. Finally, we provide an array with all FCs and the corresponding GEs in Figure 5.



FCs	GEs	
oT Process Management		
Process Modelling	-	✗
Process Execution	-	✗
Virtual Entity		
VE Service	-	✗
VE Service Resolution	-	✗
VE & IoT Service Monitoring	-	✗
IoT Service		
IoT Service	IoT Broker & IDAS	⚠
IoT Service Resolution	IoT Discovery	⚠
Service Organization		
Service Composition	-	✗
Service Orchestration	-	✗
Service Choreography	ORION Context Broker	✓
Communication		
End to End Communication	-	✗
Network Communication	Gateway Data Handling	⚠
Hop to Hop Communication	Protocol Adapter	⚠
Security		
Authorization	KeyRock IDM	✓
Authentication	KeyRock IDM	✓
Identity Management	KeyRock IDM	✓
Key Exchange & Management	-	✗
Trust & Reputation	KeyRock IDM	✓
Management		
Configuration	-	✗
Fault	-	✗
Reporting	-	✗
Member	KeyRock IDM	⚠
State	-	✗

Figure 5: IoT-A FIWARE Correlation Table.

4 CONCLUSIONS

Existing IoT architectures have been conceptualized and implemented to address certain challenges based on domain and use case specific requirements not considering issues of openness, scalability, interoperability and use-case independence. As a result they, are less principled, lacking standards and are vendor or domain specific. More importantly, they are hardly replicable in the sense that the same architecture cannot be used in more than one use cases. The IoT-A project addresses the above problems by providing the standards and specifications for designing IoT architectures.

In this work, we focused on the relationship between FIWARE and IoT-A. We proposed an architecture based on IoT-A and showed how it can be implemented on FIWARE using GEs. The results show that FIWARE cannot currently support IoT architectures fully compatible with the IoT-A standards and specifications as it is lacking the tools (GEs) to do so. New GEs need to be designed and implemented to provide the required functionality while a clear distinction of the architecture layers specified by IoT-A should be made by redesigning the GEs.

ACKNOWLEDGEMENTS

The research leading to these results has received funding from the European Unions Seventh Framework Programme (FP7/2007-2013) under grant agreement no 604691 (project FI-STAR)

REFERENCES

IoT-A references.

Preventis, A., Stravoskoufos, K., Sotiriadis, S. and Petrakis, E. (2015), A Future Internet Gesture Recognition Cloud Service, *14th International Symposium on Parallel and Distributed Computing*, 29 Jun - 01 July, 2015, Limassol, Cyprus.

Sotiriadis, S., Petrakis, G.M.E., Covaci, S., Zampognaro, P., Georga, E., Thuemmler, C. (2013) *"An architecture for designing Future Internet (FI) applications in sensitive domains: Expressing the Software to data paradigm by utilizing hybrid cloud technology"*, 13th IEEE International Conference on BioInformatics and BioEngineering (BIBE 2013), November 10-13, Chania, Greece.