Towards an IoT Framework for Semantic and Organizational Interoperability

Ivana Podnar Zarko*, Sergios Soursos†, Ivan Gojmerac **, Elena Garrido Ostermann‡, Gianluca Insolvibile§, Marcin Plociennik ¶, Peter Reichl ‖, and Giuseppe Bianchi ††

*University of Zagreb, Faculty of Electrical Engineering and Computing, Croatia, ivana.podnar@fer.hr

†Intracom SA Telecom Solutions, Greece, souse@intracom-telecom.com

‡ATOS Spain SA, Spain, elena.garrido@atos.net

§Nextworks Srl, Italy, g.insolvibile@nextworks.it

¶Poznan Supercomputing and Networking Center, IBCh PAS, Poland, marcinp@man.poznan.pl

‖University of Vienna, Faculty of Computer Science, Austria, peter.reichl@univie.ac.at

**AIT Austrian Institute of Technology GmbH, Center for Digital Safety & Security, Austria, ivan.gojmerac@ait.ac.at

††CNIT / University of Roma Tor Vergata, Rome, Italy, giuseppe.bianchi@uniroma2.it

***Abstract*—While the current highly fragmented IoT ecosystem is characterized by an increasing number of platforms, their interoperability and collaboration is quite challenging to achieve, even more so due to numerous standardization initiatives. However, interoperability remains essential for IoT deployments to facilitate the emergence of novel cross-domain applications and business opportunities. In this paper we present the interoperability approach pursued by the H2020 project symbIoTe which aims at creating a *flexible interoperability framework* supporting both semantic and organizational interoperability. While semantic interoperability, as a prerequisite for platform cooperation, has been widely addressed in literature, symbIoTe goes a step further to propose novel aspects of organizational interoperability by introducing a concept of IoT platform federations and roaming IoT devices. We present the symbIoTe architecture, highlight its major technical contributions, and provide an overview of current system implementation with focus on cloud-based platform services representing a new page for IoT interoperability and platform federations.**

I. Introduction

With over 300 IoT platforms on the market[1], the current IoT ecosystem is highly fragmented and divergent: a series of vertical domain-specific solutions coexist and occupy the same homes, factories or municipalities, but cannot interoperate since they are built using proprietary software without open and standardized interfaces. It is expected that platform interaction and collaboration will exhibit the full potential of IoT services only by enabling cross-domain applications and dynamic smart environments where moving devices blend with the surroundings in accordance with Weiser's vision of ubiquitous computing. However, achieving true IoT platform interoperability is rather challenging, not only because of the need to discover devices supporting different protocols across heterogeneous platforms, but because sharing of resources across stakeholders requires semantic alignment, secure and trusted interactions, as well as well-defined bartering and trading schemes.

symbIoTe steps into this landscape to devise *a flexible and secure interoperability framework* across IoT platforms with the following contributions: 1) it builds an abstraction layer for transparent usage of IoT devices across platforms for rapid cross-platform application development, 2) it supports IoT platform federations, i.e., associations between two platforms facilitating their secure interaction, collaboration and bartering/trading of devices for the benefit of all interested stakeholders, and 3) it facilitates blending of next generation smart (moving) devices with surrounding environments and host platforms.

Syntactic and semantic interoperability have already been widely addressed in literature [1], [2] and as essential interoperability mechanisms are also addressed by symbIoTe. However, symbIoTe goes a step further to introduce novel flavors of *organizational interoperability* (platform federations with bartering/trading concepts and roaming IoT devices), while the architecture based on microservices facilitates a flexible and incremental deployment of symbIoTe functionality across a platforms space. This enables platform providers to choose an adequate interoperability model for their business needs and desired level of collaboration with other platforms within a symbIoTe-enabled IoT ecosystem.

In this paper we present the technical details relating to novel aspects of interoperability introduced by symbIoTe, and put them in relation to the symbIoTe architecture which is built around a hierarchical IoT stack, following the principles of the oneM2M Functional Architecture [3]. oneM2M and state-of-the-art solutions currently focus on semantic interoperability, while a limited set of operations is foreseen for platform interoperability, e.g., subscriptions to information about resources offered by different platforms. symbIoTe is devising interoperability solutions in parallel with six other H2020 projects under the trademark of the IoT Platforms Initiative (http://iot-epi.eu/). Its unique features compared to other projects are interoperability solutions which go across the stack (a similar approach is adopted by the Inter-IoT project, however it is still too early to identify technical

[1] Source: Beecham Research

similarities and differences), while other projects mainly focus on either device & gateway or Cloud domain.

The paper is structured as follows: Section II presents the symbIoTe architecture and introduces the interoperability aspects. Section III defines the key technical decisions and features guiding the system design, while we outline the implementation status in Section IV. Section V concludes the paper.

II. The symbIoTe Architecture

The symbIoTe approach is built around a layered IoT stack connecting various devices (sensors, actuators and IoT gateways) within Smart Spaces with the Cloud. Smart Spaces share the available local resources (connectivity, computing and storage), while platform services running in the Cloud support IoT Platform Federations and open up the Interworking Interface shown in Figure 1 to third parties. The architecture comprises four layered domains, 1) Application Domain, 2) Cloud Domain, 3) Smart Space Domain and 4) Device Domain, as depicted in Figure 1. Hereafter we list the main functional objectives for each of these domains:

- Application Domain (APP): enables platforms to register IoT devices which they want to advertise and make accessible via symbIoTe to third parties, while symbIoTe Core Services search for adequate IoT devices across platforms. It also hosts domain-specific back-end services (Domain Enablers) which are designed to ease the process of cross-platform and domain-specific application development (specifically for mobile and web applications).
- Cloud Domain (CLD): provides a uniform and authorize access to virtualized IoT devices exposed by platforms to third parties through an open API (Interworking interface). In addition, it offers services for IoT Platform Federations enabling close platform collaboration, in accordance with platform-specific business rules.
- Smart Space Domain (SSP): provides services for discovery and registration of new IoT devices in dynamic local smart spaces, dynamic configuration of devices in accordance with predefined policies in those environments, and well- documented interfaces for devices available in smart spaces.
- Smart Device Domain (SDEV): relates to smart devices and their roaming capabilities. We assume that devices have the capabilities to blend with a surrounding smart space while they are on the move. In other words, smart devices can interact with devices in a visited smart space managed by a visited platform, in accordance with predefined access policies.

A. Interoperability Aspects

symbIoTe allows for flexible interoperability mechanisms which can be achieved by introducing an incremental deployment of symbIoTe functionality across the listed architectural domains (APP, CLD, SSP and SDEV). This approach will enable platform providers to choose an appropriate level of integration of symbIoTe-specific services within their platforms, which will in effect influence the level of platform collaboration and cooperation with other platforms within a symbIoTe-enabled ecosystem. For example, a platform may only choose to expose its Interworking Interface and selected IoT services to third parties in order to advertise them by using the symbIoTe Core Services, or it may opt for a closer collaboration with another platform by forming a platform federation. Platform federations require additional symbIoTe components to be included and integrated within a platform space in CLD.

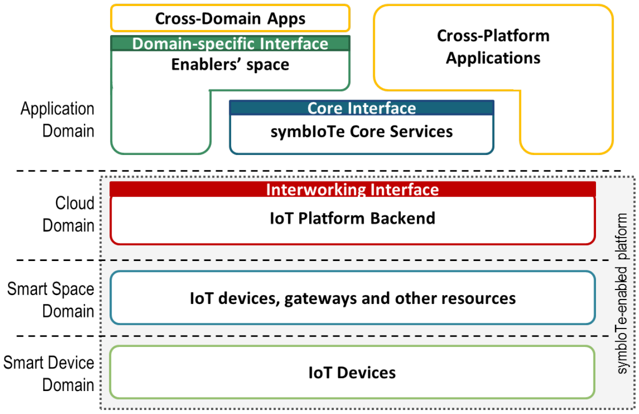

Fig. 1. The symbIoTe high-level architecture

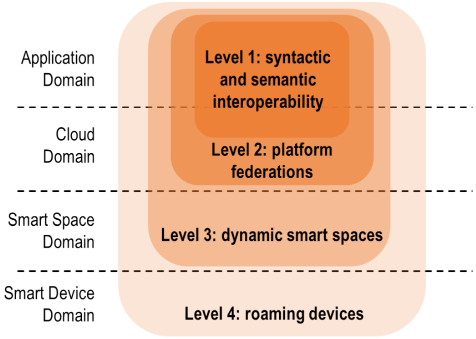
Application Domain
Level 1: syntactic and semantic interoperability
Cloud Domain
Level 2: platform federations
Smart Space Domain
Level 3: dynamic smart spaces
Smart Device Domain
Level 4: roaming devices

Fig. 2. symbIoTe Compliance Levels

We define four different Compliance Levels (CLs) for IoT platforms, as depicted in Figure 2. They reflect different interoperability modes, which an IoT platform can support. Different interoperability modes affect the functionality which needs to be supported by platforms, and require specific symbIoTe components to be integrated within different domains.

- Level 1 (L1) Compliant Platform: This is a ”lightweight” symbIoTe CL since a platform opens up only its Interworking Interface to third parties to advertise and offer its virtualized resources through the symbIoTe Core Services. L1 compliance supports the syntactic and semantic interoperability of IoT platforms in a symbIoTe ecosystem, and affects only APP and CLD.
- Level 2 (L2) Compliant Platform: This level assumes that

platforms federate, which requires additional functionality needed for close organizational interoperability, for example for device bartering/trading.

- Level 3 (L3) Compliant Platform: This CL assumes that platforms integrate symbIoTe components within their smart spaces to simplify the integration and dynamic reconfiguration of IoT devices within local spaces.
- Level 4 (L4) Compliant Platform: This level offers support for device roaming and can enable the interaction of smart objects with visited smart spaces. A prerequisite is that a platform is already marked as L1, L2 & L3, so that smart spaces can discover new visiting devices and integrate them dynamically (e.g., grant access to certain local resources) in accordance with Service Level Agreements (SLAs) between platforms (i.e. platforms are in federation).

L1 compliance can be directly mapped to semantic and syntactic interoperability, as identified in the ETSI Whitepaper [4], and subsequently adopted by IERC [5]. L2, L3 and L4 platforms can clearly be categorized as systems supporting organizational interoperability. symbIoTe proposes here an original approach with finer granularity of organizational interoperability by placing specific interoperability concepts in the CLD for L2, in the SSP for L3 as well as in both SSP and SDEV for L4 compliance. In particular, L2 platforms form platform federations, L3 platforms support dynamic and reconfigurable smart spaces, while L4 platforms support roaming of smart devices which can use services in visited smart spaces. To achieve L2 compliance, a platform should first adhere to L1 compliance, while an L4 platform requires a full symbIoTe framework.

L1 compliance relates to services placed in two domains, APP and CLD. An IoT platform which wants to become part of the symbIoTe ecosystem needs to integrate the symbIoTe Interworking Interface with its existing components, e.g., with services exposing sensor-generated data or actuation primitives. This facilitates open yet uniform access to IoT services across platform. Note that a platform chooses which devices it wants to register and make discoverable via the symbIoTe Core Services. In addition, the platform issues access tokens to its devices to third parties and keeps the control of both device services and access. symbIoTe plays here a mediation role and uses distributed and decoupled mechanisms for authentication and authorization, namely the Attribute Based Access Control (ABAC) with token-based authorization [6].

Figure 3 shows the benefits of L1 compliance by an example depicting two platforms A and B using the symbIoTe Core Services. When an application searches for devices and identifies adequate ones, the application accesses the devices offered by the two platforms through the Interworking Interface. In other words, cross-platform applications i) use the symbIoTe Core Services to find adequate devices across platforms and ii) access, integrate and use those devices through a uniform and open interface. Note that symbIoTe stores only resource metadata within the Core Services to provide adequate search mechanisms, while cross-platform applications access and use resources directly at the platform side.

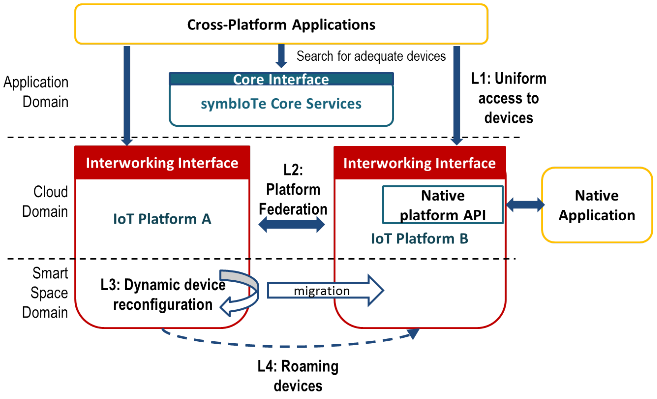

Fig. 3. Illustrating symbIoTe CLs

L2 compliance involves components placed both in APP and CLD, but requires a significant extension of an existing platform deployment to enable a closer collaboration between two platforms, i.e., platform federation. This collaboration should adhere to a specified SLA and support symbIoTe-specific bartering and trading mechanisms. Figure 3 also illustrates an example federation where it is possible to expose certain IoT services from platform A within the space of platform B. This creates an opportunity that an existing application (native application) expands the set of available services within platform B, since they appear as native services to an existing application built exclusively for platform B.

L3 and L4 compliance mainly affect platform software which is deployed within a SSP, and may require specific software also at the level of IoT devices. Features similar to the ones appearing in APP and CLD are needed within SSP, but with quite a different and reduced scope relating to local resources. Since number of platforms can occupy the same SSP, L3 compliance refers to dynamic reconfiguration of devices within a SSP, so that a device is reconfigured on the fly to become part another platform preventing thus vendor lock-in. L4 compliance relates to interoperability at the SDEV level. An example is when a device registered in platform A visits an environment operated by platform B. The device can use the surrounding infrastructure operated by platform B in accordance with an SLA between the two platforms.

III. Technical Details

A. Cross-Platform Application Development (L1)

Cross-platform application development is enabled by the symbIoTe *Core Services* and L1 platforms. In this setup symbIoTe acts as an intermediary between IoT applications and platforms where platforms can register their resources[2], while applications can search for adequate resources. The access to resources remains on the platform side; however,

[2] Resource: is a uniquely addressable entity and may refer to IoT devices, virtual entities, network equipment, computational resources and associated server-side functions (e.g., data stream processing). This definition is on purpose highly generic and in line with oneM2M specifications.

it is provided through a uniform open interface extended by the symbIoTe solution for authentication and authorization.

Let us explain this scenario by an example where two platforms are used for environmental monitoring: openUwedat [7] gathers and processes data from in-situ environmental stations, while the OpenIoT platform can be used for collecting environmental parameters by means of wearable sensors and smartphones [8]. If both platforms register, e.g., CO sensors with Core Services and implement the Interworking Interface, an application looking for CO sensors located in a certain area will receive a list URIs pointing to platform services exposing the measurements through the open interface. The application can thus fetch the readings directly from platform's REST interfaces, but only if it can present an adequate token to the platform which certifies the authenticity of both the issuer (i.e., the symbIoTe Core Services in this case) and the owner (the application).

The example showcases the need for semantic interoperability since symbIoTe needs to understand that both platforms are offering gas sensors measuring CO concentrations, although they are using different platform-specific information models. symbIoTe chooses to follow an approach which requires that all registered resources are defined using a minimalistic Core Information Model (CIM) which all platforms need to adhere to. The CIM is specified as an ontology providing the basic information about sensors, actuators and services, while further resource details can only be described using platform-specific information models. This provides a lot of flexibility for platform owners, but may represent a weakness for the symbIoTe ranking function if it does not understand many details about the resources. We envision that a mapping solution will be available to map an information model which symbIoTe does not understand to the one which symbIoTe understands. Further details are provided in [9].

Provision of data and system security in distributed, hierarchical systems like symbIoTe requires sophisticated mechanisms of user authentication and authorization. Attribute based access control (ABAC) fulfills these requirements since it is based on the assignment of attributes to various applications,components and entities in the system. An attribute is defined as a particular property, role or permission associated to an entity in the system, assigned after an authentication procedure by the system administrator. ABAC controls the access to resources by Access Control Policies. An access policy defining a specific combination of attributes needed to grant access to resources is assigned to each resource by the producer of that resource, i.e., a platform. Therefore, a client application may be granted access to a resource only if it possesses a set of attributes that match the predefined access policy. In symbIoTe this policy can contain at the same time attributes assigned to users and objects and also particular environment conditions connected to a request. Further details are provided in [6].

Borrowing their name from the FIWARE components, the *symbIoTe Enablers* also reside at the APP, but their purpose is to provide domain-specific functionality by aggregating resources belonging to different IoT platforms. For example, an enabler for air quality monitoring could collect air quality readings from appropriate sensors being managed by different platforms within the same city, perform certain processing techniques so as to analyze the collected data and provide the output in an as-a-Service manner to applications. This way, the application does not need to interact with multiple platforms and does not need to have domain-specific knowledge to process air quality data. In addition, cross-domain applications using multiple enablers can leverage and combine services offered by different domain enablers. Thus, application developers can easily create innovative applications by focusing only on the cross-domain logic, without having to care for the domain-specific details or direct interactions with multiple IoT platforms.

B. Platform Federations (L2)

Platform federations are enabled by symbIoTe components which extend platform features to enable direct platform interactions. Interworking Interface is also used here to access and use resources offered by federated platforms, however additional functionality is needed for managing SLAs and resource Bartering & Trading between the platforms. To enable trade between two platforms, a trust relationship must exist. Different parameters can be used to evaluate this trust, by, e.g., comparing the actual Quality of Experience (QoE) with established SLAs.

Coming back to the example with two platforms for environmental monitoring, their federation makes sense in case of a partnering relationship where openUwedat is willing to barter its precise and reliable measurements with OpenIoT measurements to increase its spatial coverage, while OpenIoT needs openUwedat measurements e.g. to identify uncalibrated wearable sensors.

The question of how to perform bartering and trading of resources between IoT platforms is considered essential for establishing platform federations. Here, bartering refers to any scenario where a market participant exchanges her goods or services directly for goods or services originating from another market participant, without monetary implications. Note that, in the context of a platform federation, most of the typical problems concerning efficiency of such a mechanism disappear by definition: for instance, matching suitable partners is relatively easy, as all platforms participating in symbIoTe are assumed to be prosumers. Since a huge number of resources is available, any bartering deal can be based on small service units and hence circumvent the problem of indivisibility. Moreover, symbIoTe introduces the concept of vouchers, typically comprising access tokens and SLAs as well as details on the requested service (e.g. its value and related time constraints). Thus, the symbIoTe bartering mechanism will allow for achieving joint win-win situations within a platform federation in a very straightforward way.

symbIoTe aims to offer ways to access resources from other platforms without an immediate material counteroffer, i.e., by trading. Here, we have to distinguish two basic scenarios: a) a platform is offering access to own resources and asks for corresponding requests (bids) from other platforms (forward trading), or b) a platform is looking for access to resources offered by foreign platform(s) (backward trading). Here, an agreement on monetary compensation is fundamental for closing a deal. In microeconomics, such situations are usually treated within the framework of auction theory, i.e. forward auctions (access to resources is offered, and requests are submitted in the form of bids) and reverse auctions (access to resources is requested, and access conditions are looked for by the requesting platform). The symbIoTe approach is focusing on a suitable adaptation of Progressive Second-Price (PSP) auctions, or a more general Vickrey-Clarke-Groves mechanism with reserved prices, which have been proven to be incentive compatible and thus force auction participants to be honest concerning their estimations about the value of the offered/requested resources.

C. APP and CLD Components

Taking into account the features and technical details for L1 and L2 compliance stated in the previous subsections, we present the main components which are being designed and implemented for the APP and CLD. The corresponding component diagrams are depicted in Figure 4.

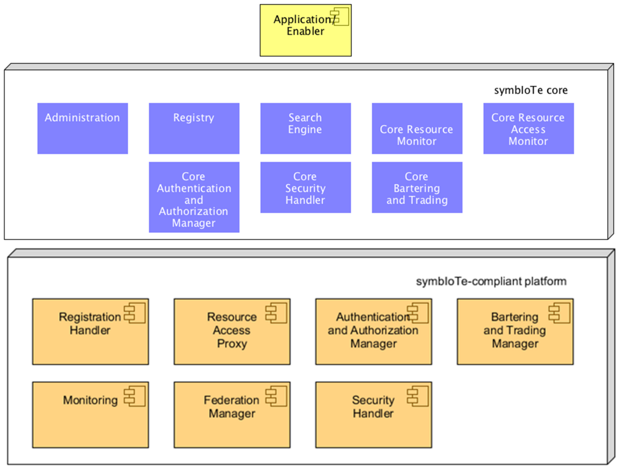
Application Enabler
symbIoTe core
Administration
Registry
Search Engine
Core Resource Monitor
Core Resource Access Monitor
Core Authentication and Authorization Manager
Core Security Handler
Core Bartering and Trading
symbIoTe-compliant platform
Registration Handler
Resource Access Proxy
Authentication and Authorization Manager
Bartering and Trading Manager
Monitoring
Federation Manager
Security Handler

Fig. 4. APP and CLD components

The essential Core Service component is the *Registry* that maintains a repository of symbIoTe-enabled platforms, registered resources and associated properties. The Core Services store and manage only IoT resource descriptions (i.e. resource metadata), while the access to those resources (e.g., sensor data and actuation primitives) is provided by the underlying platforms. *Search Engine* is needed so that applications can identify adequate resources recommended by the Core Services and access them directly on the platform side. To improve search results we include the following two components: 1) *Core Resource Monitor* tracks availability of registered resources to ensure their availability; 2) *Core Resource Access Monitor* tracks information about resource popularity as seen by symbIoTe.

L1 platforms need to implement the symbIoTe Interworking Interface to be integrated into our ecosystem. To ease the integration process, we offer components which extend the platform's space with additional symbIoTe-specific functionality: *Resource Handler* registers selected resources with the Core. *Resource Access Proxy* receives requests for resource access from applications and translates them to platform-specific requests. It addition, it supports continuous queries and delivery of sensor data via WebSockets. *Monitoring* is intended to monitor resource health status and report it periodically to its counterpart within the Core Services, namely the Core Resource Monitor. *Authentication and Authorization Manager* enables a common authentication and authorization mechanism on the platform side, both for L1 and L2 interactions.

The remaining components within APP are the following: *Core Authentication and Authorization Manager* ensures that trusted platforms register resources with symbIoTe, while mapping resource access rights to proper credentials. *Core Security Handler* provides a set of generic security-related features required for the attribute based access control. Finally, *Core Bartering and Trading Component* integrates all bartering and trading functionalities for L2 Compliance that need to be centralized.

Platforms wishing to join federations need to be extended with additional functionality provided by the following components: *Federation Manager* offers SLA management between IoT Platforms to create a federation and is monitoring whether SLAs are being respected. *Bartering and Trading Manager* manages bartering and trading actions within established federations.

D. Dynamic Spaces and Roaming Devices (L3/L4)

Smart Spaces in the symbIoTe vision are generic environments (residence, campus, vessel, stadium, etc.) where one or more different local IoT platforms coexist. Such environments are typically associated to physical locations, ranging from wide spaces to small areas; a SPP defines abstract boundaries for the IoT services and platforms it embraces, and acts as a sort of gateway from local resources to the rest of the symbIoTe environment. Smart Spaces can host a multitude of devices with a goal to seamlessly connect, dynamically configure and automatically register devices with the Core Services; subsequently, the Search Engine and applications can transparently search for and use exposed resources. A SDEV relies on the functions provided by an SSP, in order to roam, associate, and be accessible from the symbIoTe Core or by any symbIoTe app. Any device capable of complying with the symbIoTe SSP interface for registration and resource access can be considered an SDEV.

To provide a truly interoperable approach at the SSP and SDEV level, we design a dynamic association and configuration process to allow any SDEV to become part of symbIoTe, provided that a minimal, device specific software shim is installed in the SSP. In this way, the SSP can annex both entire IoT platforms and individual devices which are not associated to any particular platform. An SSP can expose to the Core Services all the devices it has access to, regardless of which native IoT platform they belong to; therefore, SDEVs associated to a SSP can be exposed directly, without being "mediated" by any of the local platforms. Furthermore, symbIoTe-compliant platforms and services operating within a SSP are able to access all the resources associated to the SSP itself, provided that the necessary federated authentication and authorization policies between the relevant platforms are in place.

The interoperability function is thus fully deployed at the SSP level: when more than one IoT platform is active within a SSP, symbIoTe becomes a local resource interchange hub where interactions happen locally without a need to contact platform clouds. Entities visiting a symbIoTe SSP include both incoming apps (e.g. a user with a smartphone or tablet running a specific symbIoTe app) and incoming devices. In both cases, the incoming entity should be identified, authorized and given a way to access the rest of the SSPs facilities, while still keeping a consistent API. One of the challenges in managing visiting entities is to keep a fairly functional system even in case of temporary failure or degradation of Internet connectivity.

An L3 compliant SDEV is able to move from one SSP to another seamlessly, i.e. it is automatically reconfigured and re-annexed to a SSP. In particular, when visiting a "foreign" environment the SDEV will be able to use resources in the surrounding infrastructure, and offer its own resources to others, provided that the required SLAs are in place. In L3 mode, the SDEV is reconfigured as a new device each time it moves from one SSP to another. On the other hand, L4 compliance mandates that a SDEV connecting to a new SSP maintains the association with its "home" SSP, behaving as a roaming (as opposed to nomadic) device. This also implies that the L4 SDEV is always identifiable and traceable as it moves between SSPs.

IV. Implementation Status

The symbIoTe consortium is developing open source software implementing the introduced architecture and features available at https://github.com/symbiote-h2020. Both the Core Services and platform components are designed and implemented using the microservices architecture, having in mind the scalability and distributed characteristics of the architecture. A base for current system implementation is provided by the Spring framework (Spring Boot, Spring Cloud) to support component configuration, services discovery (Eureka middle tier load balancer) and tracing (Zipkin distributed tracing system). We currently focus on Core Service components and L1 compliance features, while the initial platforms becoming symbIoTe complaint are openUwedat and OpenIoT. Further details are available at the symbIoTe web site http://symbiote-h2020.eu/.

V. Conclusion

The paper presents the general concepts of the symbIoTe architecture and its interoperability aspects aiming to design and build a flexible interoperability and mediation framework for IoT platforms. The framework simplifies the process of cross-platform application development, introduces novel concepts related to IoT platform federations, as well as flexible methods for integration of smart space infrastructure and smart devices within symbIoTe-enabled environments. symbIoTe does not strive to become another IoT "superplatform": It does not store any sensor-generated data outside of IoT platform boundaries, but rather acts as a mediator between applications and platforms ensuring secure and uniform access to platform resources through well-defined interfaces. The symbIoTe architecture is built around a layered stack in accordance with the oneM2M functional architecture which currently focuses primarily on semantic interoperability. symbIoTe goes a step further to propose features relevant to organizational interoperability which are currently not covered by state-of-the-art solutions: These are related to platform federations, resource bartering and trading as well as device roaming.

Acknowledgment

This work is supported by the H2020 symbIoTe project, which has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No 688156. The authors would like to cordially thank the entire symbIoTe consortium for their valuable comments and discussions.

References

[1] J. Soldatos et al., "OpenIoT: Open source internet-of-things in the cloud," in *LNCS 9001: Interoperability and Open-Source Solutions for the Internet of Things - International Workshop*, 2014, pp. 13–25.

[2] E. Kovacs, M. Bauer, J. Kim, J. Yun, F. L. Gall, and M. Zhao, "Standards-based worldwide semantic interoperability for IoT," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 40–46, December 2016.

[3] oneM2M, "M2M Functional Architecture," Technical Specification, 2015. [Online]. Available: http://www.onem2m.org/images/files/deliverables/TS-0001-Functional_Architecture-V1_6_1.pdf

[4] H. van der Veer and A. Wiles, "Achieving Technical Interoperability - the ETSI Approach," ETSI Whitepaper No. 3, 2008.

[5] IERC, "IoT Semantic Interoperability: Research Challenges, Best Practices, Recommendations and Next Steps," Position Paper, 2015.

[6] S. Sciancalepore, M. Pilc, S. Schrder, G. Bianchi, G. Boggia, M. Pawlowski, G. Piro, M. Plociennik, and H. Weisgrab, "Attribute-based access control scheme in federated iot platforms," in *2nd Workshop on Interoperability and Open-Source Solutions for the Internet of Things (InterOSS-IoT 2016)*, LNCS To appear in 2017.

[7] J. Schabauer, G. Schimak, G. Dünnebeil, and M. Litzenberger, "openUwedat - a toolbox solution for integrated air quality and traffic monitoring," in *EnviroInfo Dessau 2012, Dessau, Germany, August 29-31, 2012.*, 2012, pp. 695–706.

[8] A. Antonic, V. Bilas, M. Marjanovic, M. Matijasevic, D. Oletic, M. Pavelic, I. Podnar Zarko, K. Pripuzic, and L. Skorin-Kapov, "Urban crowd sensing demonstrator: Sense the zagreb air," in *Proceedings of the 22th International Conference on Software, Telecommunications and Computer Networks (SoftCOM), 2014*, 2014.

[9] M. Jacoby, A. Antonic, K. Kreiner, R. LŁapacz, and J. Pielorz, "Semantic interoperability as key to iot platform federation," in *2nd Workshop on Interoperability and Open-Source Solutions for the Internet of Things (InterOSS-IoT 2016)*, LNCS To appear in 2017.