Cite as:

Modoni, G. E., Veniero, M., Trombetta, A., Sacco, M., & Clemente, S. (2018). Semantic based events signaling for AAL systems. *Journal of Ambient Intelligence and Humanized Computing*, *9*(5), 1311-1325.

https://doi.org/10.1007/s12652-017-0534-0

PRE-PRINT Version

Noname manuscript No.
(will be inserted by the editor)

Semantic based events signaling for AAL systems

Gianfranco E. Modoni · Mario Veniero · Alberto Trombetta · Marco Sacco · Susanna Clemente

Received: date / Accepted: date

Abstract The synergistic cooperation of a set of different devices is still one of the major open issues in designing the new generation of collaborative AAL system. To contribute to fill this gap, this paper introduces Semantic Events Notifier (SEN), a lightweight prototype of semantic-based and agent-oriented middleware that aims to provide near real-time events notification among all the enabled devices of an AAL system. The idea behind this research work goes in the direction to find scalable technological solution to answer the continued growth of objects connected to the IoT network, in particular, within the domestic environment. SEN enables the involved Smart Objects to cooperate synergistically based on a shared semantic model, thus contributing to support various tailored services that assist elderly users or users with disabilities for a better and healthier life in their preferred living environment. Moreover, a prototype of the platform has been implemented and validated to prove the correctness of the approach. Finally, the evaluation of its quality of service requirements in terms of latency, efficiency, and scalability has been conducted in a real case study by means of a defined benchmark.

Keywords Semantic Middleware · AAL · Internet of Things · Semantic Web · Signaling service

G. E. Modoni
DiSTA - University of Insubria, Varese (VA), Italy
Institute of Industrial Technologies and Automation - National Research Council Bari, Italy
E-mail: gianfranco.modoni@itia.cnr.it

M. Veniero
abmedica spa - Cerro Maggiore (MI), Italy
E-mail: veniero.mario@abmedica.it

Alberto Trombetta
DiSTA - University of Insubria, Varese (VA), Italy
E-mail: alberto.trombetta@uninsubria.it

Marco Sacco
Institute of Industrial Technologies and Automation - National Research Council Milano, Italy
E-mail: marco.sacco@itia.cnr.it

Susanna Clemente
University of Salerno - Salerno, Italy
E-mail: sclemente@unisa.it

1 Introduction

Enhancing the efficiency of biomedical systems and healthcare infrastructures is one of the most challenging goals of modern society. Indeed, the longer life expectancy paired with the higher risk in old age to have a disability could increase the number of people who need care, with a consequent higher cost of health and social care (Broek, Cavallo, and Wehrmann 2010). Under these conditions, Ambient Assisted Living (AAL) technologies can play a significant role to enable new and more sustainable models of integrated care. For this reason, various research initiatives are currently ongoing to identify and experiment advanced technologies applied to AAL. One of the most prominent area of research in this field concerns the implementation of the Smart Home (SH), that aims to support users with disabilities for a better, healthier and safer life in their everyday living environment (Amiribesheli, Benmansour, and Bouchachia 2015) (Lotfi et al. 2012). Specifically, SH can provide ad-hoc services that allow to improve life quality for people who need permanent support, monitoring, among the others, important health parameters and providing the right medical care at the right time, while avoiding unnecessary healthcare costs and efforts.

Recent advances in the Internet of Things (IoT) can significantly support the implementation of the SH

paradigm. One of the interesting aspect of IoT protocols in AAL field is its potential to enable communication and cooperation between devices. As operations and overall behavior of IoT devices are heavily influenced by changes of state occurring within other connected devices, the SH paradigm cannot be realized without the availability of dynamic and flexible mechanisms of events notifications, that propagate these changes of state among the interested devices. The publish/subscribe interactions well adapt to the needs to synchronize the changing information among decoupled physical and virtual components distributed within an IoT platform, thus fully exploiting the potential of the Internet of Health (IoH) (Ali et al. 2015). However, they lack mechanisms to express both the data requests from the devices and the notifications events in a flexible and expressive way, obligating the subscriber to know the topic offered by the publisher and to be able to process natively the published messages. Indeed, to the best of the knowledge, solutions currently available in literature support only static mechanisms of selection of the data to be exchanged, which are based on predefined syntactical subjects (Fortino et al. 2013), while they do not consider different expressivity requirements normally needed in the definition of complex scenarios such in the AAL domain.

A major issue hindering the implementation of more expressive notification mechanisms is represented by the lack of interoperability among the various IoT devices (Atzori, Iera, and Morabito 2010), which can isolate significant data sets, emphasizing the existing problem of too much data and not enough knowledge (Sheth, Henson, and Sahoo 2008). Under these conditions the devices can lose, during their cooperation, the implicit information about the meaning of data (Gyrard, Bonnet, and Boudaoud 2014). The lack of interoperability is mainly due to the adoption of technologies with different communication interfaces, since devices are often produced from various vendors, which use different programming languages, operating systems, and hardware. Furthermore, the lack of widely accepted standards contributes to an already messy scenario. Another hurdle is the difference in the level of abstraction between the low-level raw data values acquired from sensors, the high-level models to represent analyzed and persisted data, and the requests expressed by the devices to receive information.

For these reasons, it is necessary to adopt a new model of collaboration among the various involved data sources, regardless their information representation formats. Various generic models of interoperability have been proposed up now by researchers. Among the others, it is proposed in (Sacco et al. 2014) an approach

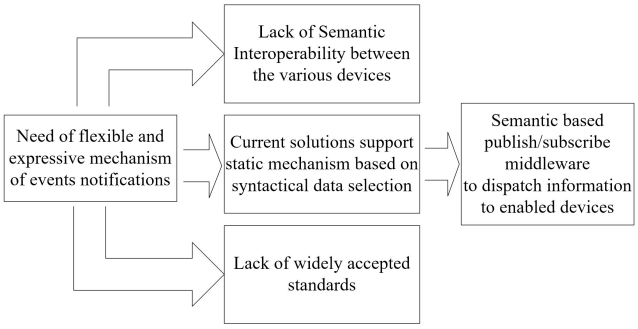

Fig. 1 Definition of the research problem

based on Semantic Web technologies (SWT) (Berners-Lee, Hendler, and Lassila 2001), which provides the definition of a reference model shared among all the devices. Basing on this reference model, this work analyzes and designs the Semantic Events Notifier (SEN), a publish/subscribe middleware that combines Semantic-Web and agent-based technologies to provide services for events notification within an ambient assisted living (AAL) system (Jammes and Smit 2005). The proposed middleware, which is one of the main outcomes of the ongoing Italian research project Design for All (D4A) (Sacco et al. 2014), has the overall aim to bring semantics to the subscription and notification specifications, thus providing more flexibility and expressiveness compared to the existing approaches.

Specifically, the SEN middleware enriches the information coming from different types of devices with semantic metadata, according to the reference model defined in the context of the D4A project, thus contributing to seamlessly integrate, aggregate and synchronize the various data sources. Moreover, it enables near real-time messages exchange among all the loosely coupled devices of an AAL system, expressing all the exchanged information (included the synchronization requests) under the form of a semantic model. Such a semantic-enabled solution allows a more accurate, flexible and adaptable characterization of the subscribers' needs and of the providers' capabilities, while it enhances the interoperability between all the involved devices. To test and validate the proposed platform, several demonstration scenarios have been identified, which are thought to represent habits and activities occurring on a regular basis in a domestic environment. A prototype has been also implemented and validated to prove the correctness of the approach. Finally, as for data intensive systems such as IoT middleware, the quality of service (QoS) requirements in terms of latency, efficiency, and scalability are stringent, an evaluation of these requirements has been conducted in a real case study by means of a defined benchmark, thus showing the impact of the

data storage, of the connected devices and of their requests.

The remainder of the paper is structured as follows. Sect. 2 reviews the literature related to this research study, whereas Sect. 3 examines the motivation for this research work. Sect. 4 illustrates the main characteristics of the middleware. Sect. 5 presents the motivating scenario, and describes the conducted experiments. Finally, Sect. 6 draws the conclusions, summarizing the main outcomes.

2 Related works

Various areas of the literature are particularly relevant to this research study. They are briefly reviewed below.

AAL and interoperability of its devices Despite todays available technological components of an AAL adopt a broad range of cutting-edge technologies, they are mostly closed systems with a limited ability to interact and to exchange data (Memon et al. 2014). They are based on closed-loop concepts, which focus on a specific purpose and are isolated from the rest of the world. Data provided by these IoT devices cannot be combined with each other since they are domain-specific and not interoperable (Miorandi et al. 2012). In fact, integration between heterogeneous elements is usually done at device or network level, and is limited to data gathering, but there is a lack of semantic interoperability across the heterogeneous IoT devices that comprise AAL system. This issue is worsened by the fact that developments based on the IoT paradigm are characterized by a large heterogeneity in terms of adopted technologies. To contribute to enhance IoT devices interoperability, various generic solutions have been proposed in literature. In (Gyrard, Bonnet, and Boudaoud 2014), it is proposed an approach based on SWT (Berners-Lee, Hendler, and Lassila 2001) to automatically combine, enrich and reason about Machine to Machine (M2M) communication data to provide cross-domain applications. The potential of the semantic-based approach has been also investigated in the context of Design For All project which aimed to enhance the semantic interoperability of different devices installed into the domestic environment, so that they can share and exploit the same information (Sacco et al. 2014). In such a scenario, SWT have been adopted to formally describe this information (including the many linking elements) in an ontology, which provides a holistic view of the smart home as a whole, considering the physical dimensions, the users involved, and their evolution over the time. This reference model is one of the basis for the implementation of the herein presented SEN. Other solutions that can integrate the heterogeneous data produced by the IoT devices are also offered by the leading commercial cloud providers such Microsoft which offers Microsoft IoT (*Microsoft IoT*) and Amazon which offers AWS IoT (*Apache ActiveMQ*). All these solutions contain different components for collecting, storing and analyzing data, enabling also a bi-directional communication between devices and the back end of the cloud. However, they do not implement mechanisms that allow connected devices to easily interact with other devices, thus signaling each-other their changes of state.

Message-oriented Middleware for IoT A significant change of state of a system can be seen as a specific event, fired when the change has occurred to be then signaled to the surrounding systems. An efficient service of events notification can be provided by a middleware application, which is a layer that acts as "software glue" between applications, operating system and network layers (Atzori, Iera, and Morabito 2010). A message-oriented middleware (MOM) can support loosely coupled inter process communication among distributed software components and thus well answers the need of large scale IoT system to support decoupled distributed interactions among the devices (Razzaque et al. 2016). Among the major features, a MOM provides synchronous and asynchronous communication mechanisms and parallel processing of messages, while supporting several levels of Quality of Service support (Curry 2004). One of its key-aspects is the concept of message queue for storing, transforming, and then forwarding messages, thus enabling asynchronous interaction, as well as a mechanism of First-In-First-Out (FIFO) queue. Apache ActiveMQ is one of the most popular open source messaging middleware (*ActiveMQ*) and supports the most well-known standard protocols for messaging. Namely, Apache ActiveMQ supports MQTT (*MQTT*), a lightweight messaging protocol built on top of TCP and characterized by low bandwidths as needed by IoT ecosystems, and Openwire (*ActiveMQ OpenWire*), the Apache ActiveMQ default message oriented protocol designed for performances optimization and supporting private peer-to-peer queues.

Semantic-based Event models A current limitation of existing MOM solutions available in literature is the need for the subscriber to know the topic specified by the publisher and to be able to process natively the published messages. In fact, such solutions support static mechanisms of selection of the data to be exchanged, which are based on predefined syntactical subjects (Fortino et al. 2013). These static configurations hinder the possibility to fully exploit the functionalities of a MOM-based approach in flexible and agile scenarios such as

in the AAL domain. To contribute to fill this gap, recent researches are endeavoring to enhance these configurations, bringing semantics to event specifications and notifications. It is expected that semantic event specifications have more flexibility and expressiveness compared to syntactical ones (Moser et al. 2009) (Gao, Ali, and Mileo 2014). In this regard, various semantic models have been proposed up now to represent a comprehensive event model that include event detection, filtering and notification. One if this work is (Moser et al. 2009), which introduced an approach that uses ontologies to facilitate semantic event correlation derived from semantic equivalence, inherited meaning, and relationships between different entities. Another approach based on Event ontology model is the Complex Event Service (CES) ontology (Gao, Ali, and Mileo 2014) which aimed to describe event services and requests. This ontology is the basis of the Automated Complex Event Implementation System (ACEIS), which plays the role of a middleware between sensor data streams and smart city applications. ACEIS integrates IoT streams and compose the relevant data to answer the users' requirements. The latter are expressed as event requests based on semantic IoT stream descriptions and are transformed by the system into a set of federated stream reasoning queries, enabling a semantic complex event processing over distributed service networks. In (Taylor and Leidinger 2011) it is proposed the use of ontologies to specify and recognize complex events that arise as selections and correlations (including temporal correlations) of structured digital messages, typically streamed from multiple sensor networks. In (Patri et al. 2016) it is introduced an approach that detect events from sensor data using a shape-based time series classification exploiting a machine learning algorithm. A knowledge driven approach is also the basis of ("A knowledge based resource discovery for Internet of Things."), which proposed an approach called Context Aware Sensor Configuration Model (CASCOM) to simplify the process of configuring IoT middleware platforms. Authors demonstrated how IoT resources can be described using semantics in such a way that they can later be used to compose service work-flows. Unlike SEN approach where information consumers are devices (things), CASCOM addresses human user, specifically non-technical personnel, that can be enabled to retrieve the data they required. A novel method to filter information contained in an Event model is introduced in (Bastinos and Lavbic 2015). Indeed, unlike the previous mentioned approaches, the latter allowed to compose the queries performed on the Event model exploiting standard semantic languages (SPARQL), that can increase the expressivity and reasoning power of event filtering. Similarly, to the framework introduced in (Bastinos and Lavbic 2015), the infrastructure acting as basis for the herein presented SEN allows the consumers to specify subscriptions in the form of SPARQL queries. However, unlike the framework illustrated in (Bastinos and Lavbic 2015), the consumers do not have to leverage an Event model to specify in the composed queries the events to which they want to subscribe, but they can directly explicit the changing of interest that have been applied on a specific knowledge base. This approach avoids the complexity of defining and managing an additional layer of abstraction (the Event model). In fact, the publisher simply updates the knowledge base domain and these changes are then notified directly to the consumer if he subscribed, while the publisher should not worry to trigger the event, as it happened in the approach reported in (Bastinos and Lavbic 2015).

Agent-based systems The implementation of SEN is based on the vision that devices can be modelled as agents that synergistically cooperate. In such an approach an agent proxies a specific function or device and then can cooperate with other agents to proactively collect data and update the current state of the system (Adis 2003). Some recent researches have already proposed the agent-based approach to design solutions of middleware. For example, case studies presented in (Tapia et al. 2010) point out that mobile agents can be successfully adopted to build context awareness systems by exploiting input from the users and their surrounding environment. On a similar vision is inspired the architecture proposed by the research project UBIWARE (Katasonov et al. 2008). The latter exploits software agent technologies to realize a middleware for the IoT, which allows the creation of a self-managed system comprising a set of distributed and heterogeneous components of different nature. Through this approach, the information related to each resources condition and their interactions are monitored and then persisted into a storage. As for the SENs design, the UBIWARE research project leveraged the IEEE FIPA agent system model (*IEEE Foundation for Intelligent Physical Agents Sta ndards Committee*). However, the solutions proposed in UBIWARE and SEN differ in how they store the systems resources. Indeed, while the first adopts a local storage for each agent, SEN uses a shared KB stored on a centralized repository, with the idea that this approach can allow to discover some important knowledge at the level of the whole system.

Middlewares Quality of Service To guarantee that applications meet their Quality of Service (QoS) require-

ments in terms of performance and scalability, it is essential that the platforms on which they are built are tested and evaluated using specific benchmarks (Liu et al. 2016). However, if a benchmark is to be useful and reliable, it must fulfill several fundamental requirements. First, it must be designed to stress platforms in a manner representative of real-world messaging applications. It must exercise all critical services provided by platforms and must provide a level playing field for performance comparisons. Different metrics can be used to facilitate a standard and systematic performance evaluation of a semantic application when dealing with huge flows of near real-time information coming from many sources; two of the most commonly used metrics are query duration and load time (Modoni, Sacco, and Terkaj 2014). Other useful metrics include disk space requirements, memory footprint and deletion duration. Based on the above metrics, several benchmarks (e.g. LUBM (Guo and Heflin 2005), Berlin Benchmark (Bizer and Schultz 2009)], BICEP Benchmark (Marcelo, Mendes, and Marques 2013), etc.) have been formalized and published. Nevertheless, all of them focus on the ontology related performances, but none of them referring to the emergence of information and to their dispatching in an IoT centered application. For this reason, a specific testing and more suitable benchmark framework was designed, to assess QoSs SEN.

3 Motivations and requirements

While todays homes are environments where various devices perform separate and isolated tasks, future homes can become systems of distributed and interconnected smart objects working together in a reliable and predictable manner (Perumal et al. 2008). An enabler of the SH is a middleware that can support devices to cooperate among themselves, acquiring, handling and sharing the knowledge about the home inhabitants (Koskela and Vnnen-Vainio-Mattila 2004) (Harper 2003). The middleware allows to abstract and hide the complexities of hardware or software components involved within the system. It also enables a proper devices interaction and enhances their capability to exchange information providing a seamless view on high-quality data extracted through a common and generalized interface. The following motivating scenario highlights the benefits of such an effective middleware for AAL. Maria is an elderly woman that lives alone in a remote house. She has a small motor impairment of her upper limbs. Moreover, she suffers from some heart-related disorders for which she must take a specific medicine when her heart rate along with blood pressure are found to be high. To monitor her physical and physiological conditions and support her daily activities, various kind of devices were recently installed in Marias house. The installed technological components comprise embedded and wearable sensors to check Marias heart rate, pulse rate, and blood pressure, environment sensors, a laptop, and a smartphone. If Maria is in the condition that a medicine should be administered, this event must be notified the nurse that take care of Maria through a specific alarm sent by means of a smartphone. Thus, it is desirable a smart mechanism that monitors the Marias physiological condition parameters, captures the event and eventually notifies the alarm towards interested consumers (such as the smartphone). For her frailty, Maria also needs to live in a healthy environment where physical dimensions such as temperature, humidity and CO2 should always be within a specific range of values. Whenever any of these measures, performed with different sensors deployed in the domestic environment, goes out of the expected range boundaries, a proper action is needed to be taken. A significant example of action can be automatically opening the window and automatically closing it when the monitored environment values return to normal. Also in this case a mechanism of signaling can be enabler of a smart interaction between the environment sensors and the window. From the above simple example, it clearly emerges that the development of such a middleware is challenging for the complexity of the interactions that must be realized among the different devices. In the following, a set of its requirements are elicited to define key aspects empowering the development of such application.

R1: Prompt notification of a state change among devices and services Middleware must propagate contextual state messages coming from physical and virtual sensors to keep the other interested components updated about interesting events. Notification are distributed as messages to other devices, so that the latter can perform further evaluations and actions, where needed. Middleware is responsible to both filter the content of each notification message and discriminate the notification target on a per-interest-subscription model. Messages content should be provided to be both automatically interpretable and machine readable. Events notification, as well as their dispatching to other components should adopt a PUSH strategy (providers announce the availability of new information) (Devanbu and Stubblebine 2000), thus realizing an event-based processing model based on publish-subscribe mechanisms.

R2: Subscription to per-consumer relevant information The capability to subscribe to relevant information is

another functional requirement for the middleware. Since an integrated semantic knowledge base includes information from several data sources, it can provide a wider scope for the basis of event filtering. Consumers should be able to subscribe to events of interest simply specifying the kind of information they are interested in without bounding to specific knowledge about the producers capabilities.

R3: Processing efficiently huge IoT stream Efficient interaction through communication technologies between consumers and producers is another significant challenge that needs to be addressed within SH. The growing quantity of devices and services that are connected to the Web makes it more difficult to deal with the communication issues among the several actors of an AAL platform (devices, services, and so on). Thus, the middleware must be supported by a scalable architecture, that reduces memory and computational cost of a massive amount of real-time data produced by devices.

R4: Supporting private interactions The middleware must implement a private peer-to-peer messaging politic. Thus, each message can be received by one single consumer, exactly that one subscribed to its content. No information will be lost. Each information authored by a sensor will be retained and made at any time available to any consumer notifies its interest to process that information.

4 Overview of SEN

SEN is a multi-agent and semantic based server providing near real-time signaling capabilities to all the networked enabled devices involved in an AAL system. It allows to distribute new emerging information to interested devices deployed in the real environment, which can register themselves to be notified about the changing of the state of one or more interesting elements of shared knowledge.

The approach behind the SEN assumes interacting devices being loosely coupled agents, synergistically cooperating as a multi-agent system to fulfill specific goals. The evaluation of the emergence of interesting information is proactively performed by a central agency, recognizing an update of the knowledge-base, and notifying eventual updates in subscribers interested information. Under these conditions, this architecture adheres to the Enterprise 2.0 Social Software (E2.0) model (McAfee 2009) (McAfee 2016), supporting social and networked applications to concurrently access and modify a shared knowledge domain.

The applicability of the proposed approach is based on a set of assumptions. Initially, a semantic-based approach has been selected to explicitly define semantics of devices data by taking as a reference a semantic model that all the devices share. The reference could be an existing semantic model or a new one designed from scratch. Moreover, data acquired from the devices are annotated and semantically enriched, according to the selected reference model. Finally, the semantic knowledge base is handled by a proper Triple Store (Modoni, Sacco, and Terkaj 2014) exposing the information through a SPARQL endpoint. The following section explores in details these assumptions.

A key feature of SEN is its capability to express both interests (subscriptions) and following alerts about changed information under the form of semantic data. Requests are expressed as SPARQL 1.1 Query Language (Harris, Seaborne, and hommeaux 2013) queries and include the expected query result according to the SPARQL 1.1 Recommendations produced by the SPARQL Working Group (*SPARQL Working Group.* 2009), allowing each consumer to manage returned information in a best-fit method (Modoni, Veniero, and Sacco 2016). Namely, the returned information are notifiable by means of JSON (Seaborne 2013b), CSV or TSV (Seaborne 2013a), XML (Dave and Broekstra 2008) or RDF (Prud'hommeaux and Seaborne 2013). This way, connected devices can cooperate synergistically by means of a semantic model through which the involved devices can share information, while the middleware supplies the central point which dispatches information through mechanisms that are transparent to their clients.

SEN leverages an effective and efficient semantic event-driven model that supports design and development activities, reducing the development cost, while also increasing interoperability, quality and portability. Such event-based solution is also considered more efficient in terms of performance compared to more common polling-based model. Being consumers not constrained to poll semantic data through continuous queries and are informed when something changes, the SEN can significantly reduce the bandwidth cost of busy spin semantic queries, as well as the required workload both at the client application and knowledge base sides.

4.1 The implementation

The design of SEN leverages the specifications of the Foundation for Intelligent Physical Agents (FIPA) (FIPA 2008), which include a full set of computer software standards for specifying how agents should communicate and interoperate within a system. Namely, it has

been considered the FIPA Subscribe-like interaction protocol (IP) specification (FIPA 2003) that defines messages to be exchanged, as well as their sequencing following a Request-Reply Enterprise Integration Pattern (Hohpe and Woolf 2004). The FIPA-compliant implementation of the SEN relies on the JADE (Bellifemine, Caire, and Greenwood 2007), a middleware for the implementation of distributed and cooperating multi-agent systems.

SEN is also empowered by a messaging system, supporting both the publish/subscribe pattern and message queue models. The first allows the specific receivers (subscribers) to express interest in one or more information expressed as SPARQL queries and receive only messages that are of interest. The second enables an asynchronous inter-process communications protocol. In the proposed implementation, the selected messaging system is the Apache ActiveMQ (*ActiveMQ*), an industrial state of the art open source messaging platform providing API in several popular industrial languages. Through the messaging system, clients can subscribe specifying their interests profile, the content type of the expected response and the authorization credentials for the repository (namely, username and password). The communication protocol of ActiveMQ has been set to Openwire (*ActiveMQ OpenWire*). Indeed, unlike other more well-known protocol in the field of IoT such as MQTT, Openwire is capable to support private queues which is one of the essential requirement of SEN.

To retain information authored by each sensor (both physical and virtual), they are stored into a Triple Store keeping track of the evolutions occurring in the domestic environment. This component also provides reasoning capabilities to automatically deduce implicit knowledge from the explicitly asserted facts (Modoni et al. 2016), driven by concepts and relationships interpretation entailments. Moreover, a valid Semantic Repository (SR) is capable to handle large amount of semantic data, also in the form of Big Data (Modoni, Sacco, and Terkaj 2014). The managed data comprise:

- the domain ontology, as representation of the knowledge about home environment;
- the ontological population, compliant to the domain ontology;
- the derivation rules needed to properly entail the implicit knowledge.

To implement the Semantic Repository, it was adopted Stardog (*Stardog*), a commercial solution of Triple Store that well answers the requirements of an intensive data application as the case of the SEN (Modoni, Sacco, and Terkaj 2014). Moreover, Stardog was deployed to a cloud-based (Armbrust et al. 2010) platform, which guarantees the horizontal scalability of the overall architecture.

4.2 Components

Fig. 2 depicts the overall structure of SEN, focusing on its semantic information, integration and dispatching capabilities. Referring the more general contexts of Enterprise 2.0 Semantic Knowledge Information systems (Hinchcliffe 2007) (Cook 2008) customized for Ambient Assistive Living (AAL) environments, SEN is in charge of providing actors interacting with the common knowledge base three fundamental capabilities: a) authoring, allowing information producers to, indeed, author newly explicit information to be shared globally, b) retrieval, as the ability to access and extract information from the knowledge base, and c) signaling, as the capability of being (near) real-time notified about emerging interesting information. This latter can be considered one of SENs fundamental feature, w.r.t. the requirements of the middleware.

The platform attempts to deal with the horizontal scalability. Because of their strict dependency, and being in this paper mainly focused on efficient signaling for real-time context aware applications, the diagram outlines the components in charge of supporting it: Update Manager (UM) and Semantic Broker (SB), on its turn made up by the Subscription Manager (SM) and the Messaging System (MS) supported by the on behalf of multi-agent System. Each physical or virtual sensor, after having gathered the information which oversees, affects the knowledge base hosted on the shared semantic repository by updating or deleting semantic assertions. This is done through a web service exposed by UM exploiting the Jersey Framework (*Project Jersey*) and wrapping up the SPARQL endpoint exposed by the internal Semantic Repository to allow the activation of the MS. On the other hand, information consumers (smart services and sensors) subscribes to the SB, providing their semantically modeled interests profile.

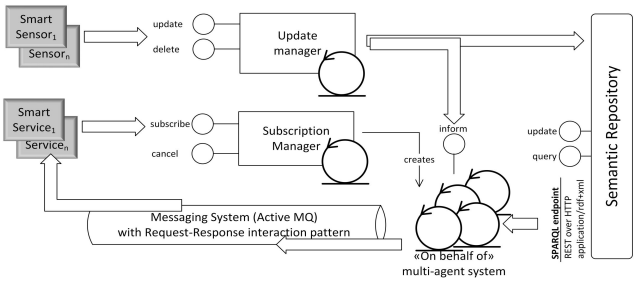

Fig. 2 Overall architecture of SEN

SM is the component which is always listening on the queue that manages the new subscriptions, leveraging the Apache ActiveMQ (*ActiveMQ*) messaging system. Whenever SM receives a subscription request from a network client, it activates server-side an agent (the ClientAgent) which takes care of the client interests. Namely, SM records this interest activating an on-behalf-of agent in charge of signal emerging new information to the consumer. If such agent already exists, SM simply notifies the new consumers interest. In each moment, the consumer can unsubscribe by cancelling the request. Each time a sensor authors new knowledge, the UM informs the SB of the occurred event so that, in a continuous query processing fashion, the SB can evaluate emerging information and notifies it to the consumer through the MS.

4.3 Interaction model

To interact with SEN, standardized subscription and cancellation protocols were defined to which both data providers and consumers must adhere. Fig. 3 shows through an UML sequence diagram the interactions among the involved entities.

All clients start the interaction with SEN through a subscription message containing the description of the information of interest, the desired response format, an eventually minimum refresh rate in milliseconds (if it desires to be kept up to date even if no information changes), a unique identifier of the request and a reference of the client to which forward the discovered information. The transmitted query should refer the shared knowledge domain described by the semantic model held by the repository. The server processes the subscription request and decides whether to accept it. If it is rejected, the repository sends to the client the rejection condition, ending the interaction. If accepted, the server creates a new on-behalf-of agent or forwards the received subscription to the already living client curator agent. The latter evaluates in its turn the query sending back an information message containing the result of the executed query, compliant with the chosen response format. In case of SELECT, CONSTRUCT or DESCRIBE query forms, the reply is bounded to a not-empty result.

After the transmission of the subscription, the client waits for subsequent notifications generated by its curator agent. This happens whenever the knowledge base is changed and the desired information emerges producing a result that is different from that previously transmitted or on the elapsing of the minimum refresh rate interval, if this is the case.

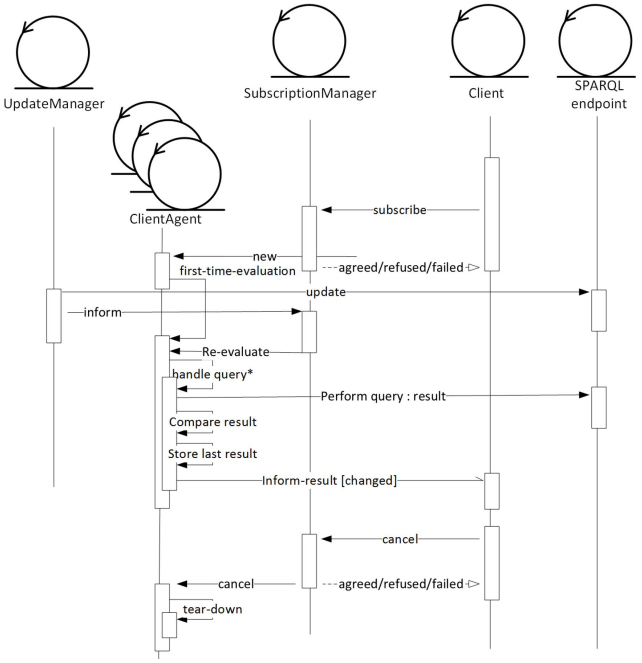

Fig. 3 SEN interaction model

The server continues to broadcast type messages as long as one of the following conditions happen:

1. The client deletes the subscription request by a cancellation request;
2. An error occurs causing the server to be no longer able to communicate with the client or to process queries.

5 The experimental case

5.1 Design For All project

Design For All (D4A) is a research project co-funded by the Italian Ministry for Education, University and Research within the cluster of initiatives for Technologies for Ambient Assisted Living. It aimed to enhance the semantic interoperability of different devices installed into the domestic environment, so that they can share and exploit the same information (Sacco et al. 2014). Semantic Web technologies have been adopted to formally describe this information (including the many linking elements) in an ontology, which provides a holistic view of the smart home as a whole, considering the physical dimensions, the users involved, and their evolution over the time. It also allows the use of reasoning tools, able to derive new knowledge about the concepts and their relationships, thanks to inferencing rules specified in the Semantic Web Rule Language (SWRL). The semantic model developed, called the

Virtual Home Data Model (VHDM), provides a consistent representation of several knowledge domains; it is composed of three main modules: a) the Physiology model, to keep track of users medical conditions over time; b) the Smart Object Model, which provides a description of the relationships between appliances and related functionalities; c) the Domestic Environment, which includes information on thermo-hygrometric conditions and air and light quality.

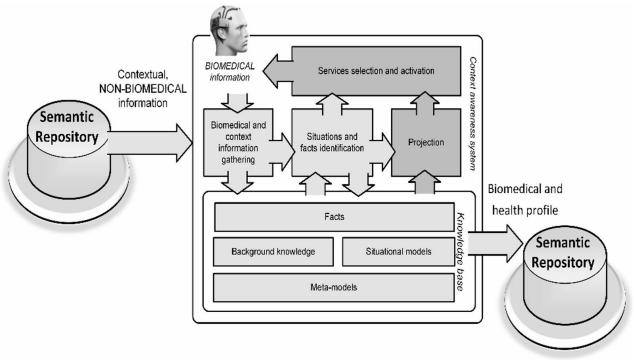

Fig. 4 Situation identification systems architecture

5.2 The conducted experiments

One of the defined experimental settings is a context aware Situation Identification System (SIS). The main goal of SIS is the identification of ongoing situations involving one or more observed users, that are relevant from a cognitive point of view in the reference scenario, and the selection of suitable services to be activated basing on the users contextual and situational profile.

The main capabilities of the system are:

- Environment perception and sensing by gathering, analyzing and semantically annotating brain signals, as well as several other bioelectrical and medical information. All the information is merged with contextual information acquired by ubiquitous home-automation sensors;
- Comprehension of the evolving scenario, extracting of all those features useful to identify facts about the observed subjects having a relevance from a cognitive perspective in the reference scenario (i.e. subject characterization as well as relations established with other elements of the scene and currently assessed situations). Contextual profiling of the observed user, together with the identification of his status and behavioral patterns are the main results of this activity;
- Projection of the understood scenario with the identification of possibly occurring situations;
- Activation of services needed by the current contextual and situational user profile to positively affect the environment.

As can be seen in Fig. 4, the general architecture of SIS recalls the Endsleys model for Situation Awareness (Endsley and Garland 2000) and is based on a specific knowledge base feeding and sustaining situational models both in identification and processing situations. These models and their underlying knowledge are, in their turn, described according to an application ontology focused on modeling and reasoning on medical and behavioral features related to the observed subject. This architectural approach has proven its flexibility and effectiveness in several applicative contexts, ranging from airport security (Furno, Loia, and Veniero 2010) to harbor security (Clemente, Loia, and Veniero 2014) and ambient intelligence (Furno et al. 2011).

SIS retrieves contextual information coming from smart objects deployed in the environment such as internal and external temperature, light levels, devices activation, users intentions as feedback of executed tasks, etc. Standard bioelectrical and biomedical information, on the other hand, are gathered by means of standard medical, CE certified, Bluetooth sensors such as blood glucometers, thermometers, sphygmomanometers and heart rate monitors, made smart through a machine-to-machine gateway able to put medical information in the loop. Finally, non-stationary electroencephalographic signals are acquired through an ad-hoc made electroencephalographic wireless helmet, built and owned by abmedica. Gathered signals are analyzed to extract useful features related to the ongoing physiological phenomenon, mental states, closed/open eyes condition, stress level and so on.

During its continuous reasoning process, SIS feeds back to the semantic repository part of the inferred contextual and situational profile, namely the observed users current state and health profile, thus concurring to enrich the D4A commonly agreed instantaneous profile of the observed user allowing other parties (virtual reality software applications, adaptive user interfaces, and so on) to leverage also on that information to evaluate the premises to execute specific actions to support user (Sacco et al. 2014).

Fig. 5 depicts the general experimental settings realized in the context of D4A validation. On the left are represented defined sensors in the role of information producer: a) medical sensors (thermometer, sphygmomanometer, oximeter and heart rate monitor, EEG helmet), together with the information gathering application; b) environment physical sensors (such as light, temperature, humiture, etc.) and virtual sensors, i.e. a comfort evaluation sensor aggregating physical sensors

values in a comfort index, once again fed back to the shared knowledge; c) the SIS system, acting both as information consumer and as information publisher(in the role of virtual sensor), providing inferred instantaneous state and health profile of the observed user. On the right, adaptive interfaces and fitness supporting appliances are provided.

All these third-party appliances (sensors, actuators, and services) depend on a prompt discovery of information about the observed user, allowing them to synchronize their internal beliefs and to (near) real-time react to changes occurred in user environment, or user conditions. Given the complex nature of handled information, as well as the potentially huge number of information consumers, a busy-wait PULL-based information discovery model, can rapidly saturate both computational and network resources, understandably soon degrading the overall system performances.

Thus, the efficient PUSH-oriented discovery mechanism provided by SEN was adopted to allow information consumer to promptly discover information relevant for adaptation and control activities.

5.3 The semantic model and the SPARQL queries

According to the aforementioned setting (Fig. 5) in what follows are listed the information provided by the smart sensors:

- Environmental temperature, humiture and lighting. The sensors, a DS18B20 Temperature Sensor module, a DHT11 Humiture sensor and a Photoresistor module, all of them from Sunfounder and controlled by a Raspberry PC, provide information about temperature, light level and humiture percentage of the room where the user is located. The related semantically annotated information is provided with a frequency of 0.017Hz, allowing the system to read environmental information approximately once a minute;
- Environmental comfort evaluation. A virtual sensor gathering environmental sensors data to provide an aggregated index assessing the living comfort. This sensor acts both as information consumer in that it depends on data instantaneously collected from smart sensors, and as a smart sensor, proving back to the AAL application ecosystem the inferred comfort level;
- Biomedical and bioelectrical data. Measurements are performed before and during the user performs an exercise on a bicycle ergometer, whose activity is monitored by a fitness evaluation system. Namely, measurements relate to temperature, blood pressure (both systolic and diastolic), heart rate. Attention and stress level are evaluated by the EEG helmet. In this case, measurements are performed only before and after the exercise, to avoid artefacts and noise induced by the occurring movement.

Environmental conditions gathered by smart sensors are updated through a delete and replace approach, leveraging the adopted Triple Store transactional support to ensure information consistency.

A different storing approach was selected for different information types and sources. Contextual environment conditions, such as light levels, temperature, and so on, are stored on an instantaneous-only basis, because in the experimental setting the corresponding historical series was recognized to be useful only to the comfort evaluation sensor. On the other hand, biomedical information is authored and retrieved through a more complex assertions set, in that subject historical clinical condition was considered somehow relevant for more than a personalization or adaptive system, as well as for design tools leveraging them.

Thus, focusing on light level and temperature and humiture sensors and considering the bedroom as reference environment, information is updated through the following SPARQL Update commands:

```
PREFIX env: <http://www.itia.cnr.it/
    Domestic_Environment_ABox#>
PREFIX envr: <http://www.itia.cnr.it/
    Domestic_Environment_TBox#>
DELETE WHERE {
    env:Room_Bedroom
        envr:hasInteriorTemperature ?t .
};
INSERT DATA {
    env:Room_Bedroom
        envr:hasInteriorTemperature
            temperature .
}
PREFIX env: <http://www.itia.cnr.it/
    Domestic_Environment_ABox#>
PREFIX envr: <http://www.itia.cnr.it/
    Domestic_Environment_TBox#>
DELETE WHERE {
    env:Room_Bedroom
        envr:hasInternalLuminance ?l .
};
INSERT DATA {
    env:Room_Bedroom
        envr:hasInternalLuminance
            luminance
}

PREFIX env: <http://www.itia.cnr.it/
```

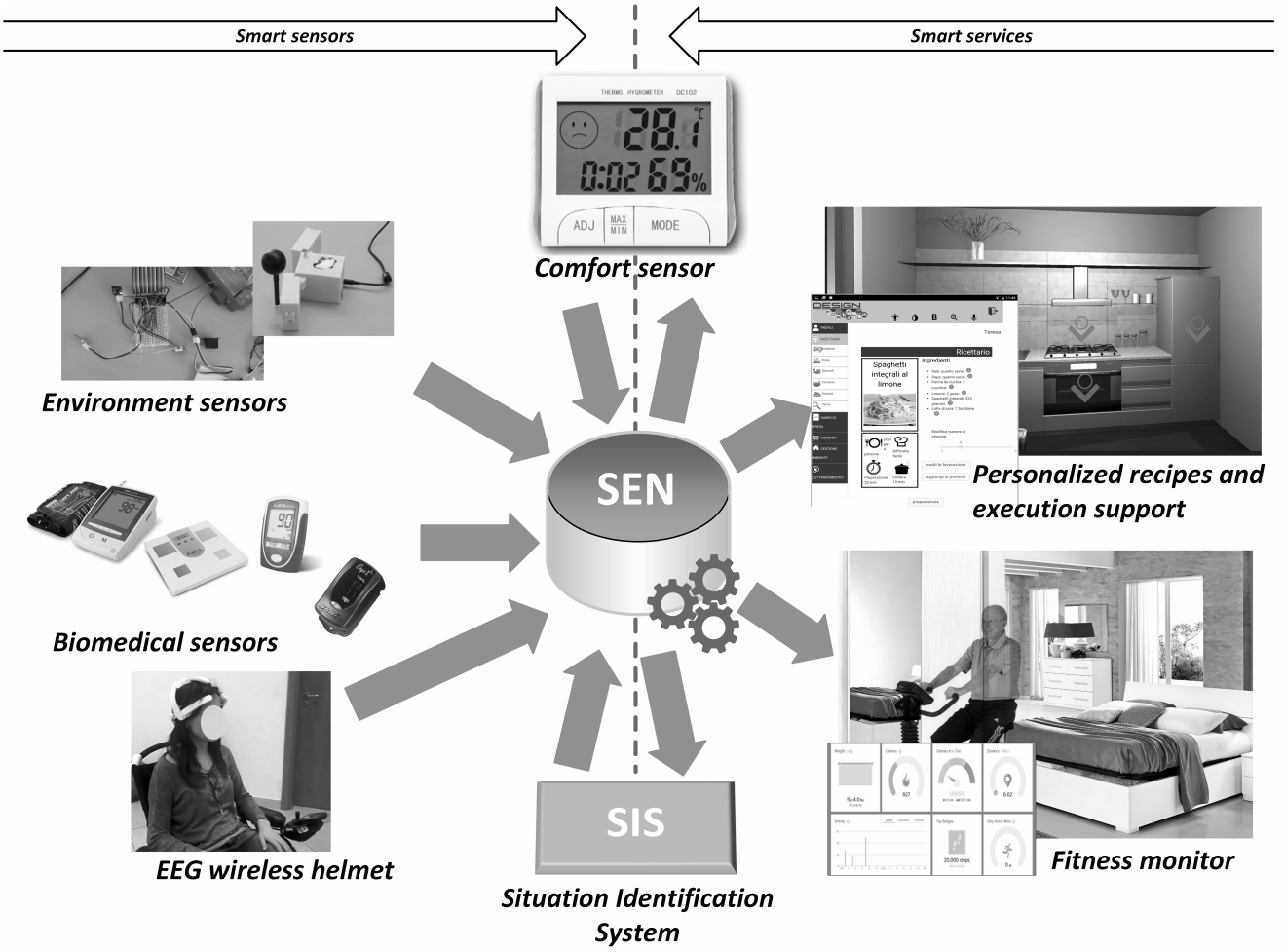

Fig. 5 Experimental setting using SIS

```
    Domestic_Environment_ABox#>
PREFIX envr: <http://www.itia.cnr.it/
    Domestic_Environment_TBox#>
DELETE WHERE {
    env:Room_Bedroom
        envr:hasInternalHumiture ?h .
};
INSERT DATA {
    env:Room_Bedroom
        envr:hasInternalHumiture
            humiture
};
```

where *temperature*, *luminance* and *humiture* are the instantaneous floating-point got values, respectively in Celsius degrees, lumen and humiture percentage. This way, the shared knowledge base constantly is aware of the latest instantaneous value for each sensor.

Analogously, an example of how smart services can register for the discovery of these information by means of the following SPARQL Query profile is the following:

```
PREFIX env: <http://www.itia.cnr.it/
    Domestic_Environment_ABox#>
PREFIX envr: <http://www.itia.cnr.it/
    Domestic_Environment_TBox#>
SELECT ?t, ?l, ?h
WHERE {
    env:Room_Bedroom
        envr:hasInteriorTemperature ?t .
    env:Room_Bedroom
        envr:hasInternalLuminance ?l .
    env:Room_Bedroom
        envr:hasInternalHumiture ?h .
}
```

Any more complex query definition according to the D4A shared ontology can be used to discover complex emerging information, once again a delete and replace base.

```
PREFIX rdf: <http://www.w3.org/1999/02/
    22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/
```

```
    01/rdf-schema#>
PREFIX common: <http://www.itia.cnr.it/
    Common_Box_D4A#>
PREFIX usr: <http://www.itia.cnr.it
    /Design4All_UserOntology#>
PREFIX msrTBOX: <http://www.itia.cnr.it/
    SO_measurement_descriptor_TBox#>
PREFIX msrABOX: <http://www.itia.cnr.it/
    SO_measurement_descriptor_ABox#>
PREFIX vsignABOX: <http://www.itia.cnr
    .it/Basic_Vital_Sign_Ontology_ABox#>
DELETE WHERE {
    usr:Donald
        usr:hasAssociatedMeasurement ?m
};
DELETE WHERE {
    ?m rdf:type
        msrTBOX:PhysiologicalMeasurement
};
DELETE WHERE {
    ?m msrTBOX:hasAssessmentDate ?o
};
DELETE WHERE {
    ?m msrTBOX:hasDescription ?o
};
DELETE WHERE {
    ?m msrTBOX:hasMeasurementValue ?o
};
DELETE WHERE {
    ?m msrTBOX:refers_to
        vsignABOX:VS_Trouble_condition
};
INSERT DATA {
    usr:Donald
        usr:hasAssociatedMeasurement
            msrABOX:Measure_e08
};
INSERT DATA {
    msrABOX: Measure_e08 rdf:type
        msrTBOX:PhysiologicalMeasurement
};
INSERT DATA {
    msrABOX: Measure_e08
        msrTBOX:hasAssessmentDate
            "2017-03-12T22:53:10"
};
INSERT DATA {
    msrABOX: Measure_e08
        msrTBOX:hasMeasurementValue
            value
};
INSERT DATA {
    msrABOX:Measure_e08
        msrTBOX:refers_to
            vsignABOX:vitalsign
};
```

where vsignABOX:*vitalsign* can be one of the following:

- VS_Systolic_blood_pressure and VS_Diastolic_blood_pressure, respectively, instantaneous systolic/diastolic blood pressure;
- VS_Glucose_concentration, the measured blood glucose concentration;
- VS_Pulse_rate, the current pulse rate;
- SpO2_concentration, the instantaneous oximetry;
- VS_Axillary_temperature, the axillary temperature of the user.

For each of the authored information, the simplest way for a smart service to discover the instantaneous picture of the user, is by issuing the following SPARQL Query.

```
PREFIX usr: <http://www.itia.cnr.it/
  Design4All_UserOntology#>
PREFIX msrTBOX: <http://www.itia.cnr.it/
  SO_measurement_descriptor_TBox#>
PREFIX msrABOX: <http://www.itia.cnr.it/
  SO_measurement_descriptor_ABox#>
PREFIX vsignABOX: <http://www.itia.cnr
  .it/Basic_Vital_Sign_Ontology_ABox#>
SELECT ?user ?v
WHERE {
    ?user
        usr:hasAssociatedMeasurement ?m .
    ?m
        msrTBOX:hasMeasurementValue ?v.
    ?m
        msrTBOX:hasAssessmentDate ?date .
    ?m
        msrTBOX:refers_to
            vsignABOX:vitalsign
}
ORDER BY DESC(?date) LIMIT 1
```

where *vitalsign* is one of the previously described biomedical information.

5.4 The performance evaluation framework

Performance and scalability are often high priority concerns when selecting a middleware, especially in data intensive application. Moreover, they are among those Information Systems features that can be evaluated and assessed through quantifiable criteria, though it is necessary first to understand how to define and implement tests that accurately model the expected workloads while stripping away uncontrollable elements. On

the other hand, the proposed middleware has been implemented in the form of a semantic-oriented Continuous Event Processing system. Under this hat, considering the AAL target context, it is recognizable as a good representative of an event-driven application with real-time, mission critical, performance-sensitive constraints, generating huge amounts of data and requiring very short response times to support adaptation and service to user needs. This way a systematic evaluation of its performances, related to its enabling technologies is required. Various benchmarks have been formalized and developed up now in literature; most of them depend on the specific application scenario and are focused on a given functional and performance domain requirements set. According to (Gyrard et al. 2015), we choose to define a self-made framework that considered the need of automated and timely answers, taking into account the underlying technologies and scenario, while allowing to parametrize performance indicators affecting behavior of both producers and clients.

We agreed to evaluate middleware performances in terms of its ability to notify changes applied to the KB under several conditions. The speed is mainly affected by five major dimensions:

- the size of the knowledge (in triples count) to work on;
- the complexity of the entailment supporting inference and deduction;
- the size of the derived deduction model (in triples count);
- the number of working information producers and the update rate;
- the number of subscripted consumers and the complexity of required knowledge graphs.

When dealing with information emergence identification or querying, the first three issues mainly affect the size of the model making up the knowledge base, and consequently the time required to find the triples graph matching the information modeled by the discovery profile. Indeed, the complexity of the entailment introduces a further issue w.r.t. performance evaluation, bound to the time required to be applied to an eventually huge KB before executing any query. This, on its turn, depends on the chosen algorithm and the size of the explicit assertions making up the KB itself. Therefore, we decided to remove this element to obtain a simpler model.

The benchmark consists in processing simultaneously several parameters affecting the systems workload:

- $K \subseteq \mathbb{N} - \{0\}$, the size of the knowledge base to work on, in terms of triples count;
- $S \subseteq \mathbb{N} - \{0\}$, the number of working sensors;
- $C \subseteq \mathbb{N} - \{0\}$, the number of subscripted consumers (smart services or sensors) under a set of queries;
- $Q \subseteq \mathbb{N} - \{0\}$, the number of subscriptions for each consumer.

Because the size of knowledge base is the main reference parameter, we combined described parameters by means of the following weighted expression:

$$\alpha\xi_{k_M}(k)\cdot(1+\beta\xi_{s_M}(s))\cdot(1+\gamma\xi_{c_M}(c))\cdot(1+\delta\xi_{q_M}(q)) \quad (1)$$

where

- $\alpha \in]0 \dots 1]$, $\beta, \gamma, \delta \in [0 \dots 1]$ are constants representing the relevance of each parameter in the problems space. Constraining α to be a positive value we recognize that the size of the knowledge base is the main affecting parameter;
- $k_M \in K - \{1\}$, $s_M \in S - \{1\}$, $c_M \in C - \{1\}$, and $q_M \in Q - \{1\}$ are, respectively, $maxK$, $maxS$, $maxC$ and $maxQ$.

Finally, said P one of the identified performance affecting parameters values set, for any $\rho_M > 1$ and $\rho \geq 1$, $\xi_{\rho_M} : P \to [0 \dots 1]$ is a normalization function defined as

$$\xi_{\rho_M}(\rho) = \frac{ln(\rho)}{ln(\rho_M)} \quad (2)$$

ξ is aimed to balance the effects of different increment schemes foreseen for each basic parameter. Note that $0 \leq \xi_{\rho_M}(\rho) < 1$ for any pair ρ, ρ_M.

The reference load factor is given by the size of the knowledge base, while the other parameters are considered magnifying factors, with a specific magnification impact. In our experiment, we set $\alpha = 1$, $\beta = 0.5$, $\gamma = 1$ and $\delta = 0.7$. With this assumption, we want to stress the fact that the main affecting parameters are the size of the knowledge base and the number of subscripted consumers. Indeed, sensors influences the number of signaling activations depending on uncontrollable scheduling issues. In this sense, more authoring activities can be masked by a single following signaling, being thus collapsed in a single action. At the same time, the number of subscripted queries affects response time because of the serialized management approach adopted to serve a specific subscriber needs.

Considering the described parameters, each of them can assume values in a different range. Furthermore, each of them may have a generally different increasing step, and different impact on the overall performance when ρ approaches ρ_M on higher order values. For instance, triples number grows very fast, but the difference of the impact on performances of a given triples

number and the immediately next increase may be negligible. At the same time, the difference of the impact on performances between a low triples number and a high one is generally very significant. On the other hand, this does not generally hold when considering one of the remaining dimensions. These thoughts lead us to consider we need to consider some tailoring when normalizing over the several measured dimensions. The increase step is introduced in the normalization function, together with a specialized logarithms root for each of considered parameter, allowing to specialize the growing rate of logarithm while enhancing the differences between low and high values.

def. 1 *Given P a performance parameter, calling i_P the increase step for P, and r_P the root logarithm used to normalize values over P, $\xi^P_{\rho_M} : \mathbb{N} - 0 \to [0 \dots 1]$ is a normalization function defined as*

$$\xi^P_{\rho_M}(\rho) = \frac{log_{r_P}(\frac{\rho}{i_P} + 1)}{log_{r_P}(\frac{\rho_M}{i_P} + 1)} \tag{3}$$

where ρ_M is the maximum value of ρ in P.

In the conducted experiment we set

- $k_M = 10^6$, $i_K = 5 \cdot 10^4$ and $r_K = 1.2$ (thus selecting a more rapidly increasing logarithm for triples number normalization than the other ones, to enhance differences with them);
- $s_M = 10^1$, $i_S = 1$ and $r_S = e$;
- $c_M = 5 \cdot 10^2$, $i_C = 5$ and $r_C = e$;
- $q_M = 5$, $i_q = 1$ and $r_Q = e$.

Finally, it naturally results that

$$0 \leq \alpha\xi^K_{k_M}(k)(1 + \beta\xi^S_{s_M}(s))(1 + \gamma\xi^C_{c_M}(c))(1 + \delta\xi_{q_M}(q)) \leq (1 + \beta)(1 + \gamma)(1 + \delta) \tag{4}$$

Eq. 3 concurs to define the load of the system, following the definition below.

def. 1 *given k_M, s_M, c_M, and q_M respectively representing the upper limit to the size of the knowledge base, the number of working sensors, the number of subscripted consumers, and the number of subscripted queries per consumer, the load of SEN is given as the function $load : K \times S \times C \times Q \to [0..1]$ defined as*

$$load(k, s, c, q) = \frac{\alpha\xi^K_{k_M}(k)(1 + \beta\xi^S_{s_M}(s))(1 + \gamma\xi^C_{c_M}(c))(1 + \delta\xi_{q_M}(q))}{(1 + \beta)(1 + \gamma)(1 + \delta)} \tag{5}$$

In def. 1 the logarithmic scale allows to consider a slowly increasing load, especially when dealing with great input sizes, to flatten local differences.

Fig. 6 depicts the loads surface obtained varying performance parameters. As mentioned earlier, the purpose of the benchmark is to evaluate the ability of the middleware to process increasing loads while providing quick answers with low latency. Latency is commonly agreed as the time interval between a stimulation and a response in a system (or, from our perspective, the time delay between a change in the knowledge base and the notification of emerging information to the client).

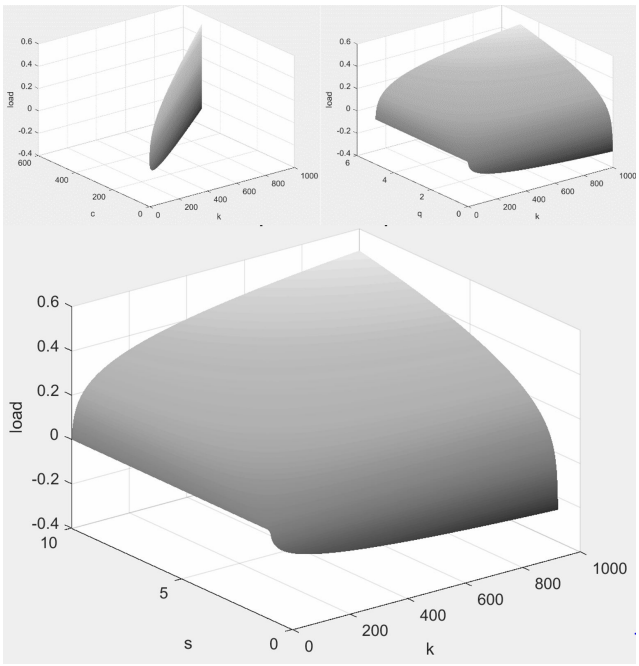

Fig. 6 SEN load focusing on triples count as main parameter (right coordinate, with a 10^3 unit) and, from top to bottom, respectively, subscribed consumers, queries per consumer and sensors as left coordinate

To evaluate performances, here we interpret latency as the stimulus turnaround time or *stimulus processing turnaround*, i.e. the average time elapsed between a new information gathering and reporting the result back to all involved subscribed consumers, under a well-defined load. To filter noise induced by several uncontrollable factors (network latency, operating system scheduler, and so on), for each instance of a performance evaluation, trial was repeated several times and the resulting latencies averaged using the harmonic mean ($latency_m$), thus giving a measure that is more robust in the presence of outlier values than other statistical means such as mean or median. W.r.t. the described experimental setting, the trials count was set to 10.

Finally, load under a given configuration t=<k,s,c,q> and the average latency are combined to define a synthetic evaluation parameter named *performance score* and denoted by p^t_{SEN}, as follows.

def. 1 *provided a trial configuration* $t =< k, s, c, q >$ *for the set of trials T, and given* $load_t$ *and* $latency_{m,t}$*, respectively representing the load induced by the trial configuration t and the average latency of stimulus processing turnaround, the score reached by SEN is given by* $p^t_{SEN} : \mathbb{R}^+_0 \to \mathbb{R}^+_0$ *defined as*

$$p^t_{SEN} = load_t \cdot \frac{max_T(latency_{m,t})}{latency_{m,t}} \tag{6}$$

where $max_T(latency_{m,t})$ *is the maximum* $latency_{m,t}$ *measured on all possible trials.*

Note that p^t_{SEN} has essentially a comparison purpose aimed to understand if SEN performances have a constant trend, w.r.t. to load affecting parameters.

To perform the evaluation a test framework has been developed as illustrated in Fig. 7.

Initially, the user specifies the work parameters k, s, c and q, together with its preferred relevance or, alternatively, uses the standard benchmark configuration to create a test setup (Step 1). Then, the Benchmark Generator module generates a *simulated environment* (Step 2) made up by the desired instances of both producers (smart sensors) and consumers (smart services), the former notifying the desire information to SEN, and the latter subscribing to its *signaling services*. The generated environment (both sensors and services) is then activated (Step 3) to start performances measurements. During the execution lifetime, each Stimulus-Reaction Correlation (SRC) is logged by SEN and transferred (*performances data flow*) to the Performances Collector. SCRs allow to both track input-output relations and tear down latency correlations. Finally, performances are aggregated and reported to the user (Step 4) as tables and diagrams needed to assess performances. All tools provide an easy to run environment and thus they require very little effort to be executed.

5.5 Perfomances evaluation

The main significant slice of over one thousand performance tests has been reported in figures from Fig. 8 to Fig. 16, where both the knowledge base size and the registered consumers change over the respective experimental ranges.

Other reports have been omitted here because the other dimensions do not add a relevant value, being analogous to the formers. More precisely, figures from Fig. 8 to Fig. 14 present the details of performance tests executed under several conditions and evaluated according to eq. 6 compared to the system load defined by eq. 5. Each figure reports on the left the system load (the lower graphic) and the evaluated p^t_{SEN} (the upper graphic). On the right side, instead, is reported the $latency_{m,t}$ in milliseconds, as the whole stimulus-receipt round-trip-time (including the time required by the access to the knowledge base, that, in its turn, averages around 60-100ms). Both sides adopt the size of the knowledge base in triples count as abscissa that ranges in $5 \cdot 10^4 \leq kleq 10^6$.

Fig. 15 and Fig. 16, instead, show, respectively, an aggregated report allowing to point out how SEN performances scales better upon the growing of systems load. As it can be seen, the system scores better under high loads, generally following its growth rate. Moreover, when the size of the knowledge base goes towards huge loads, the systems responds positively with a constantly growing score jumps significantly upwards. This seems to be true under any stress condition, as shown by Fig. 15. At the same time, Fig. 16 points out how the execution time tends to stabilize driven by the number of concurrent consumers expecting a reply.

6 Conclusions

This paper introduced a semantic signaling middleware that dispatches requested information to the enabled devices connected to an AAL system. Thus, it can contribute to enable a smarter interaction of these devices which can cooperate through the integration of the knowledge about home environment based on a common semantic model. The proposed middleware is particularly relevant in the field of AAL, in the perspective of moving away from more traditional assistive technologies towards an approach that considers the full range of human diversity.

Future developments will mainly address four goals. The first goal will concern the implementation of a Trust model to enable reliable and secure interactions between trustworthy entities. For example, in a scenario where the window should be automatically opened when the monitored environment goes out of the expected range boundaries, the smart window must be sure that environmental sensors are trustworthy to execute the action of opening. In this regard, a potential solution of Trust model is based on the evaluation of the devices reputation (Bossi, Braghin, and Trombetta 2014), so that only the devices that are granted can publish critical information. The second goal will regard the optimization of the queries execution, through a smarter

organization of the TBOX based on a subdivision in different graphs. This way, when a change is triggered, the system will reload only the queries involving the graphs affected by the changes, and, at the same time, the system could provide a real upper bound to the memory consumption induced by per-client agents activation. Moreover, while the current implementation assumes that there is a (TBOX) semantic model shared between the different devices, the future implementation will investigate the capability to allow each device to have its own semantic model, leveraging modules for ontological alignment of different models (Cudr-Mauroux 2013) (Modoni et al. 2016).

Finally, the middleware will be also used and validated in other fields different from AAL (e.g. to support a Cyber Physical System in manufacturing (Modoni, Sacco, and Terkaj 2016)). In fact, even SEN has been conceived for an AAL system, it is agnostic to the meta model of the used semantic data, and for this reason its transfer technology in other fields is lightweight and can done with little effort.

Acknowledgements This work has been co-funded by the Ministry of University and Research of Italy within the cluster Tecnologie per gli ambienti di vita Technologies for Ambient Assisted Living initiatives, with the overall objective of increasing the quality of life in the domestic environments using the modern technologies.

References

ActiveMQ. online. URL: `http://activemq.apache.org`.

ActiveMQ OpenWire. online. URL: `http://activemq.apache.org/openwire.html`.

Adis, W. (2003). "Quality of service middleware". In: *Industrial Management and Data Sys- tems* 103.1, pp. 47–51.

Ali, M. I. et al. (2015). "A semantic processing framework for IoT-enabled communication systems". In:

Amiribesheli, M., A. Benmansour, and A. Bouchachia (2015). "A review of smart homes in healthcare". In: *J AIHC* 6.4, pp. 1–23.

Apache ActiveMQ. online. URL: `https://aws.amazon.com/iot-platform/`.

Armbrust, M. et al. (2010). "A view of cloud computing". In: *Communications of the ACM* 53.4, pp. 50–58.

Atzori, L., A. Iera, and G. Morabito (2010). "The internet of things: A survey." In: *Computer networks* 54.15, pp. 2787–2805.

SPARQL-based framework for semantically-based event processing. (2015). Maribor Slovenia.

Bellifemine, F., G. Caire, and D. Greenwood (2007). *Developing multi-agent systems with JADE*. Vol. 7. John Wiley & Sons.

Berners-Lee, T., J. Hendler, and O. Lassila (2001). "The Semantic Web." In: *Scientific American* 284.5, pp. 34–43.

Bizer, C. and A. Schultz (2009). "The berlin sparql benchmark". In: *International Journal on Semantic Web and Information Systems* 5.2, pp. 1–24.

Multidimensional reputation network for service composition in the internet of things (2014). 2014 IEEE International Conference.

Broek, G. van den, F. Cavallo, and C. Wehrmann (2010). "AALIANCE ambient assisted living roadmap". In: *IOS press* 6.

Clemente, S., V. Loia, and M. Veniero (2014). "Applying cognitive situation awareness to collision avoidance for harbour last-mile area safety". In: *Journal of Ambient Intelligence and Humanized Computing* 5.5, pp. 741–745.

Cook, N. (2008). *Enterprise 2.0: how social software will change the future of work*. Gower Publishing Ltd.

Cudr-Mauroux, P. (2013). "Loose ontological coupling and the Social Semantic Web". In: *J AIHC* 4.3, pp. 349–356.

Curry, E. (2004). "Message-oriented middleware." In: *Middleware for communications*, pp. 1–28.

Dave, B. and J. Broekstra (2008). "SPARQL query results XML format. W3C Recommendation 15." In: URL: `https://www.w3.org/TR/2013/REC-rdf-sparql-XMLres-20130321/`.

Devanbu, P. T. and S. Stubblebine (2000). "Software engineering for security: a roadmap". In: *Proceedings of the Conference on the Future of Software Engineering, ACM*.

Endsley, M. and D. J. Garland (2000). "Theoretical underpinnings of situation awareness: A critical review." In: *Situation awareness analysis and measurement*, pp. 3–32.

FIPA (2008). "Fipa communicative act library specification." In: URL: `http://www.fipa.org/specs/fipa00037/SC00037J.html`.

— *IEEE Foundation for Intelligent Physical Agents Sta ndards Committee*. Tech. rep. FIPA. URL: `http://www.fipa.org/`.

— (2003). "Subscribe Interaction Protocol Specification". In: URL: `http://www.fipa.org/specs/fipa00035/`.

Fortino, G. et al. (2013). "An agent-based middleware for cooperating smart objects." In: *Highlights on Practical Applications of Agents and Multi-Agent Systems*, pp. 387–398.

Furno, D., V. Loia, and M. Veniero (2010). "A fuzzy cognitive situation awareness for airport security". In: *Control and cybernetics* 39.4, pp. 959–982.

Furno, D. et al. (2011). "Towards an agent-based architecture for managing uncertainty in situation awareness". In: IEEE Symposium on Intelligent Agent (IA). Paris.

Gao, F., M. I. Ali, and A. Mileo (2014). "Semantic discovery and integration of urban data streams." In: vol. 1280. Proceedings of the Fifth International Conference on Semantics for Smarter Cities.

Guo, Y. and J. Heflin (2005). "LUBM: A benchmark for OWL knowledge base systems". In: *Web Semantics: Science, Services and Agents on the Wo-rld Wide Web* 3.2, pp. 158–182.

Gyrard, A., C. Bonnet, and K. Boudaoud (2014). "Enrich machine-to-machine data with semantic web technologies for cross-domain applications." In: Internet of Things (WF-IoT). IEEE World Forum on.

Gyrard, A. et al. (2015). "Cross-domain Internet of Things application development: M3 framework and evaluation." In: Future Internet of Things and Cloud (FiCloud). 2015 3rd International Conference.

Harper, R. (2003). "Inside the smart home: Ideas, possibilities and methods." In: *Springer London*, pp. 1–13.

Harris, S., A. Seaborne, and E. Prud hommeaux (2013). *SPARQL 1.1 query language.* Tech. rep. W3C recommendation 21.10. URL: `https://www.w3.org/TR/sparql11-query/`.

Hinchcliffe, D. (2007). "The state of Enterprise 2.0." In: *The Enterprise Web 2.*

Hohpe, G. and B. Woolf (2004). "Enterprise integration patterns: Designing, building, and deploying messaging solutions." In: Addison-Wesley Professional, pp. –.

Jammes, F. and H. Smit (2005). "Service-oriented para digms in industrial automation." In: *IEEE Transactions on Industrial Informatics* 1.1, pp. 62–70.

Katasonov, A. et al. (2008). "Smart Semantic Middleware for the Internet of Things." In: *ICINCO-ICSO* 8, pp. 169–178.

Koskela, T. and K. Vnnen-Vainio-Mattila (2004). "Evolution towards smart home environments: empirical evaluation of three user interfaces". In: *Personal and Ubiquitous Computing* 8.3-4, pp. 234–240.

Liu, M. et al. (2016). "Semantic Agent-Based Service Middleware and Si- mulation for Smart Cities". In: *Sensors* 16.12.

Lotfi, A. et al. (2012). "Smart homes for the elderly dementia sufferers: identification and prediction of abnormal behavior". In: *J AIHC* 3.3, 205–218.

Marcelo, R. N., P. B. Mendes, and P. Marques (2013). "Towards a standard event processing benchmark." In: Proceedings of the 4th ACM/SPEC International Conference on Performance Engineering (ICPE '13). (ICPE 13), pp. 307–310.

McAfee, A. P. (2009). *Enterprise 2.0: New collaborative tools for your organization's toughest challenges.* Harvard Business Press.

— (2016). "Enterprise 2.0: The dawn of emergent collaboration." In: *MIT Sloan management review* 47.3, p. 21.

Memon, M. et al. (2014). "Ambient assisted living healthcare frameworks, platforms, standards, and quality attributes". In: 14.3, pp. 4312–4341.

Microsoft. *Microsoft IoT.* Tech. rep. Microsoft. URL: `https://www.microsoft.com/en-us/internet-of-things/`.

Miorandi, D. et al. (2012). "Internet of things: Vision, applications and research challenges." In: *Ad Hoc Networks* 10, pp. 1497–1516.

Modoni, G. E., M. Sacco, and W. Terkaj (2014). "A survey of RDF store solutions." In: International ICE Conference on Engineering, Technology and Innovation (ICE).

— (2016). "A Telemetry-driven Approach to Simulate Data-intensive Manufacturing Processes." In: *49th Procedia CIRP-CMS* 57, pp. 281–285.

Modoni, G. E., M. Veniero, and M. Sacco (2016). "Semantic Knowledge Management and Integration Services for AAL Applications". In: Proceedings ForItAAL2016,Lecture Notes in Electrical Engineering.

Modoni, G. E. et al. (2016). "Enhancing factory data integration through the development of an ontology: from the reference models reuse to the semantic conversion of the legacy models". In: *International Journal of Computer Integrated Manufacturing*, pp. 1–17.

Moser, T. et al. (2009). "Semantic event correlation using ontologies." In: *On the Move to Meaningful Internet Systems, 2009.* Vol. 5871. Springer, Berlin, Heidelberg, pp. 1087–1094.

MQTT. online. URL: `http://mqtt.org`.

Patri, O. P. et al. (2016). "Sensors to Events: Semantic Modeling and Recognition of Events from Data Streams". In: *International Journal of Semantic - Computing* 10.4, pp. 461–501.

Perera, C. and A. V. Vasilakos. "A knowledge based resource discovery for Internet of Things." In: *Knowledge based Systems* 109, pp. 122–136.

Perumal, T. et al. (2008). "Interoperability for Smart Home Environment Using Web Services". In: *International Journal of Smart Home* 2.4, pp. 1–16.

Project Jersey. online. URL: https://jersey.java.net/.

Prud'hommeaux, Eric and Andy Seaborne (2013). *SPARQL query language for RDF*. Tech. rep. W3C. URL: https://www.w3.org/TR/rdf-sparql-protocol/.

Razzaque, M. A. et al. (2016). "Middleware for internet of things: a survey." In: *IEEE Internet of Things Journal* 3.1, pp. 70–95.

Sacco, M. et al. (2014). "Supporting the Design of AAL through a SW Integration Framework: The D4All Project." In: *International Conference on Universal Access in Human-Computer Interaction*, pp. 75–84.

Seaborne, A. (2013a). *SPARQL 1.1 Query results CSV and TSV formats*. Tech. rep. W3C. URL: https://www.w3.org/TR/2013/REC-sparql11-results-csv-tsv-20130321/.

— (2013b). *SPARQL 1.1 Query Results JSON Format*. Tech. rep. W3C. URL: http://www.w3.org/TR/2013/REC-sparql11-results-json-20130321.

Sheth, A., C. Henson, and S. & Sahoo (2008). "Semantic Sensor Web." In: *IEEE Internet Computing* 12.4, pp. 78–83.

SPARQL Working Group. (2009). Tech. rep. W3C. URL: https://www.w3.org/2009/sparql/wiki/Main_Page.

Stardog. online. URL: http://stardog.com/.

Tapia, D. I. et al. (2010). "Agents and ambient intelligence: case studies". In: *J AIHC* 1.4, pp. 85–93.

Taylor, K. and L. Leidinger (2011). "Ontology-driven complex event processing in heterogeneous sensor networks." In: *The Semanic Web: Research and Applications*. Vol. 5871. Springer, pp. 285–299.

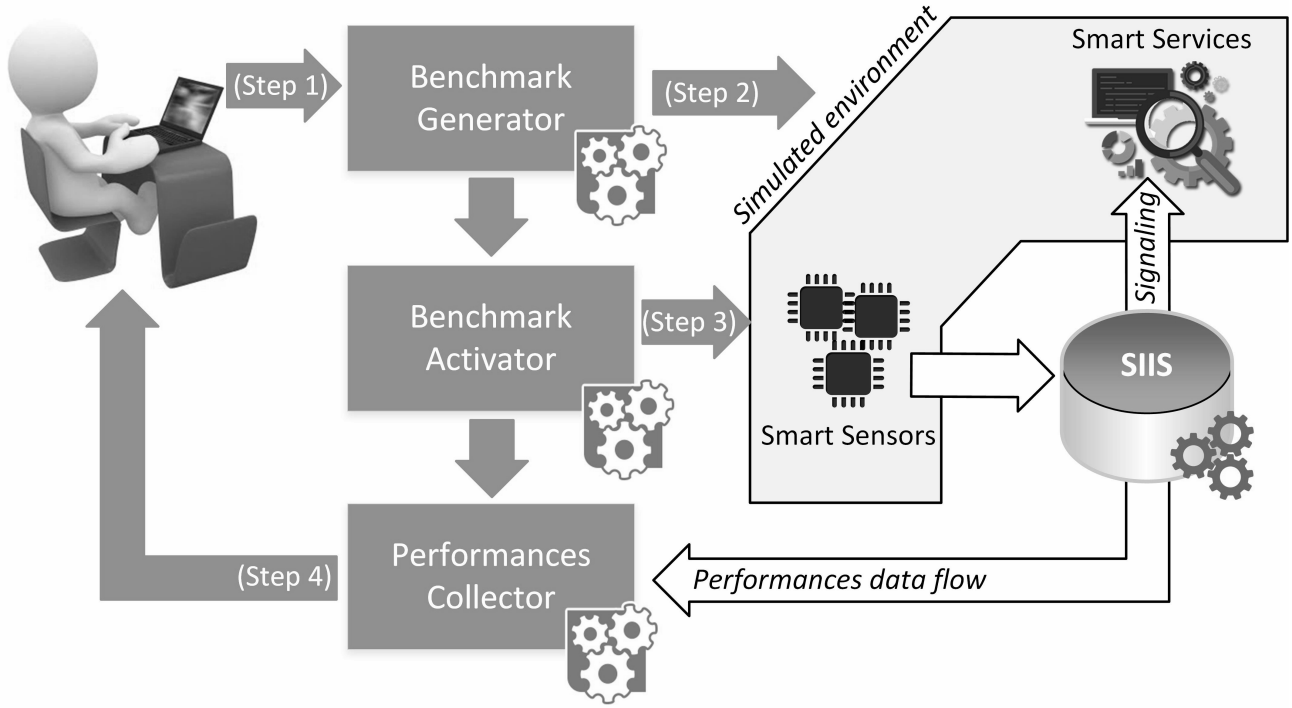

Fig. 7 Testing framework

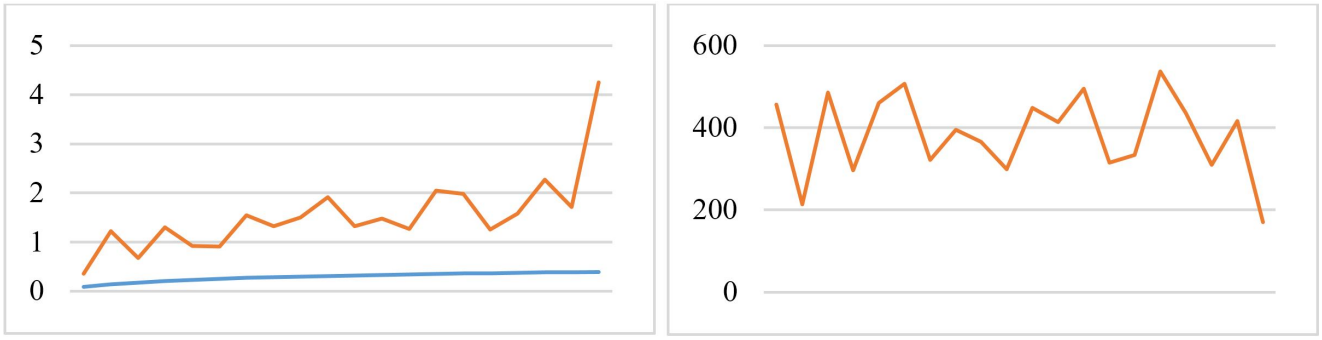

Fig. 8 p^t_{SEN}, $load_t$ (left) and $latency_{(m,t)}$ (right), $t =< k, 1, 1, 1 >$

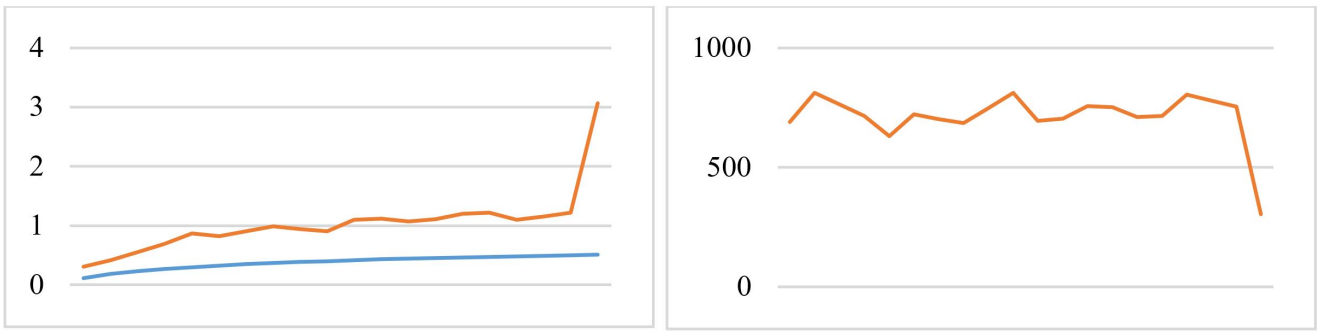

Fig. 9 p^t_{SEN}, $load_t$ (left) and $latency_{(m,t)}$ (right), $t =< k, 1, 6, 1 >$

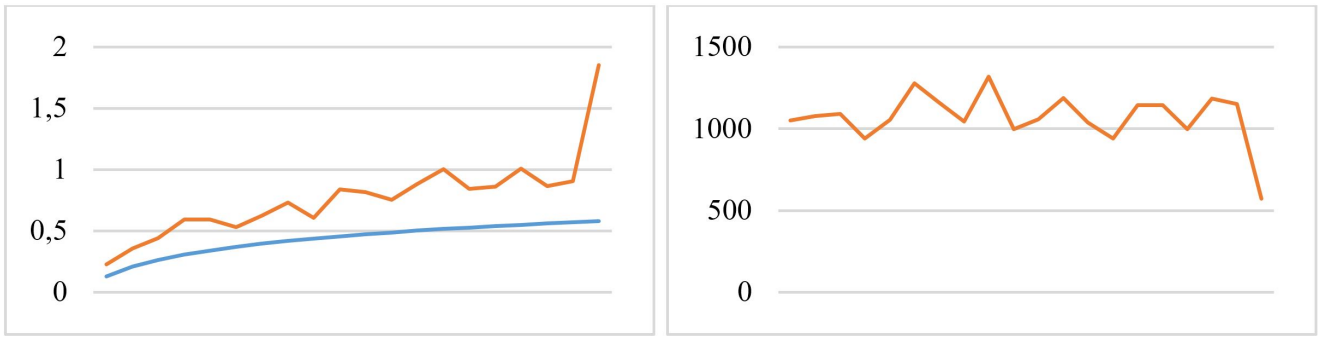

Fig. 10 p^t_{SEN}, $load_t$ (left) and $latency_{(m,t)}$ (right), $t =< k, 1, 11, 1 >$

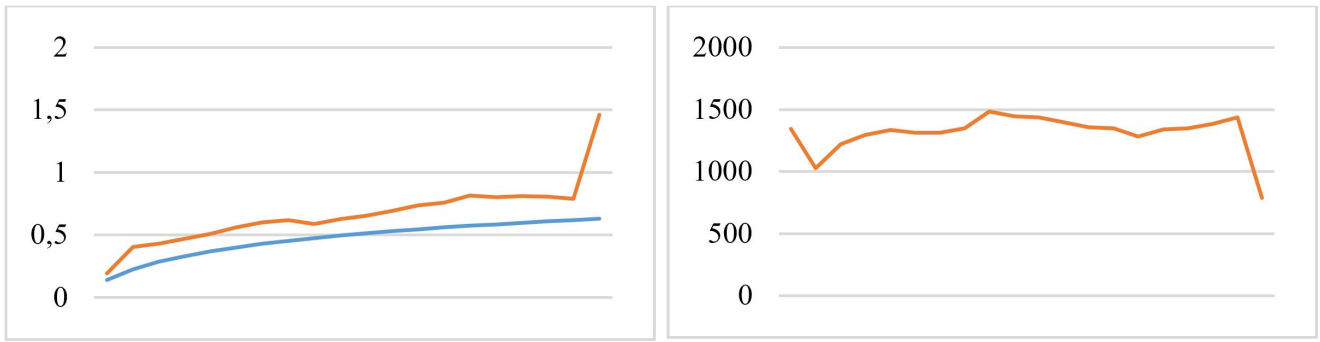

Fig. 11 p^t_{SEN}, $load_t$ (left) and $latency_{(m,t)}$ (right), $t =< k, 1, 16, 1 >$

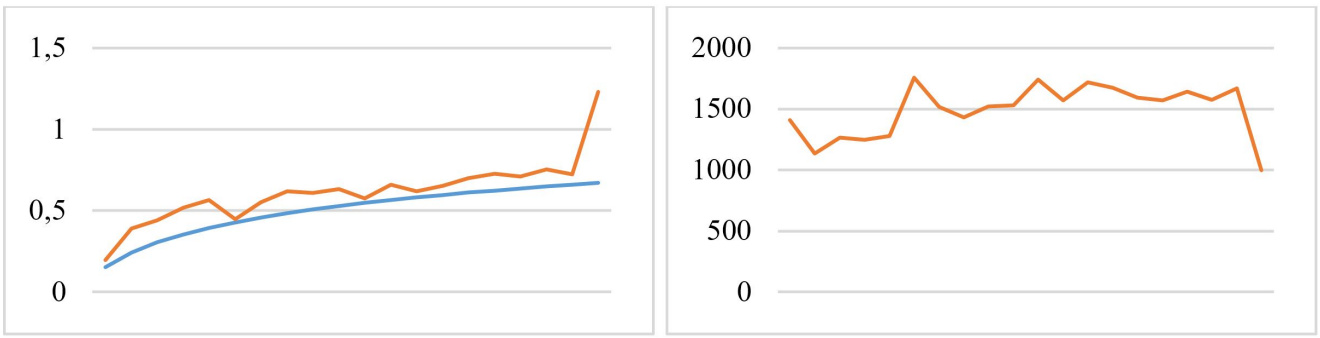

Fig. 12 p^t_{SEN}, $load_t$ (left) and $latency_{(m,t)}$ (right), $t =< k, 1, 21, 1 >$

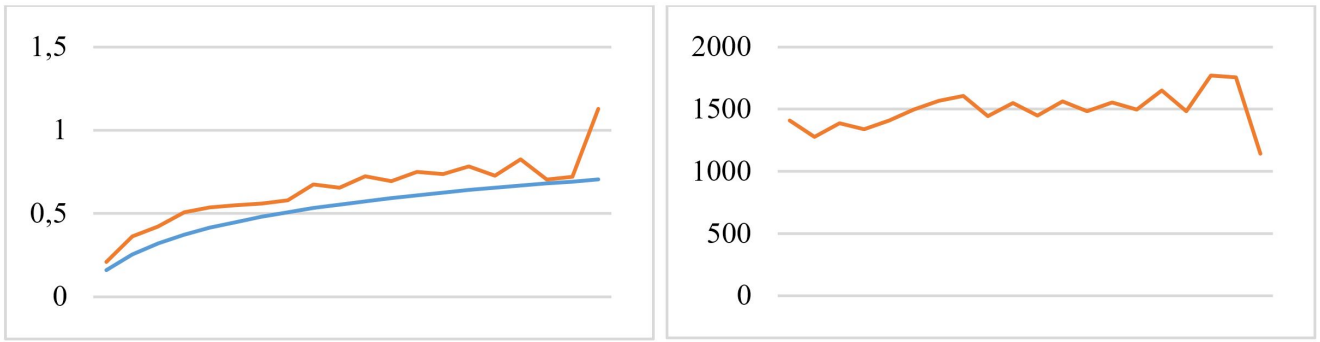

Fig. 13 p^t_{SEN}, $load_t$ (left) and $latency_{(m,t)}$ (right), $t =< k, 1, 26, 1 >$

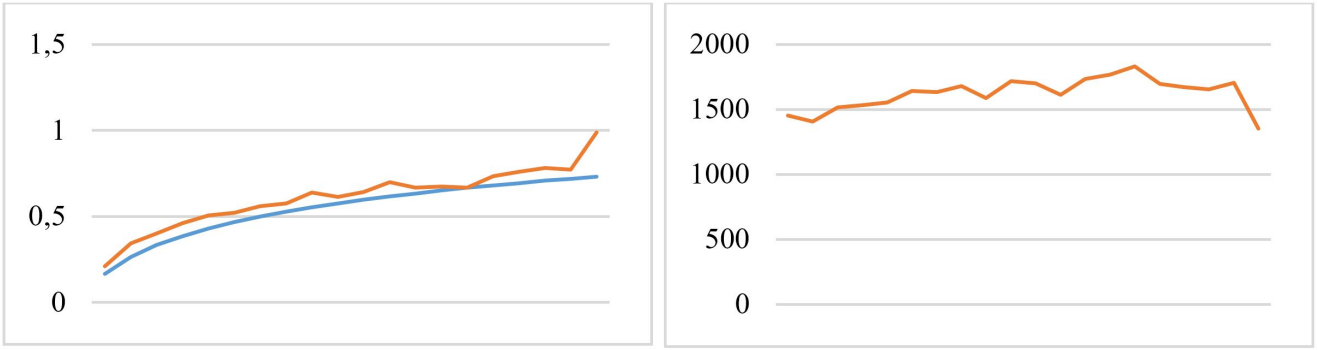

Fig. 14 p^t_{SEN}, $load_t$ (left) and $latency_{(m,t)}$ (right), $t =< k, 1, 31, 1 >$

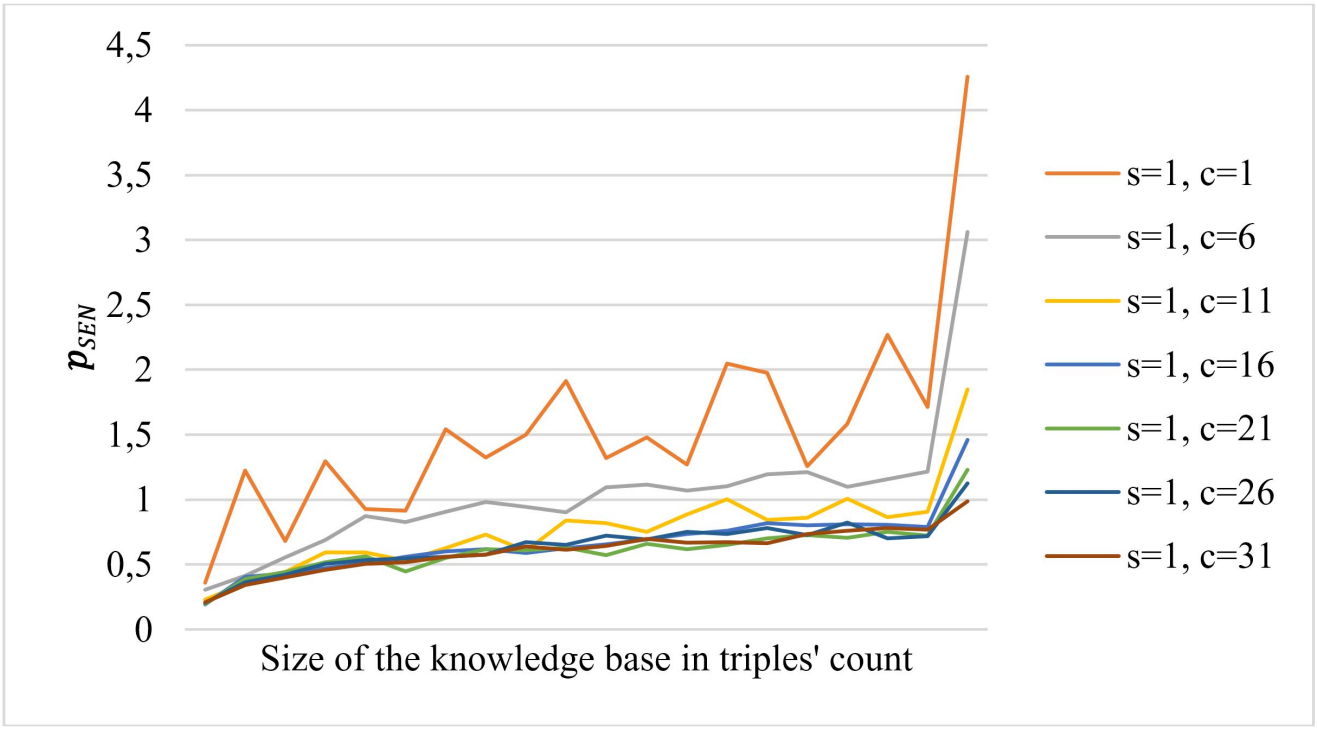

Fig. 15 p^t_{SEN}, with $t =< k, 1, c, 1 >$ where $5 \cdot 10^4 \leq k \leq 10^6$ and $1 \leq c \leq 31$

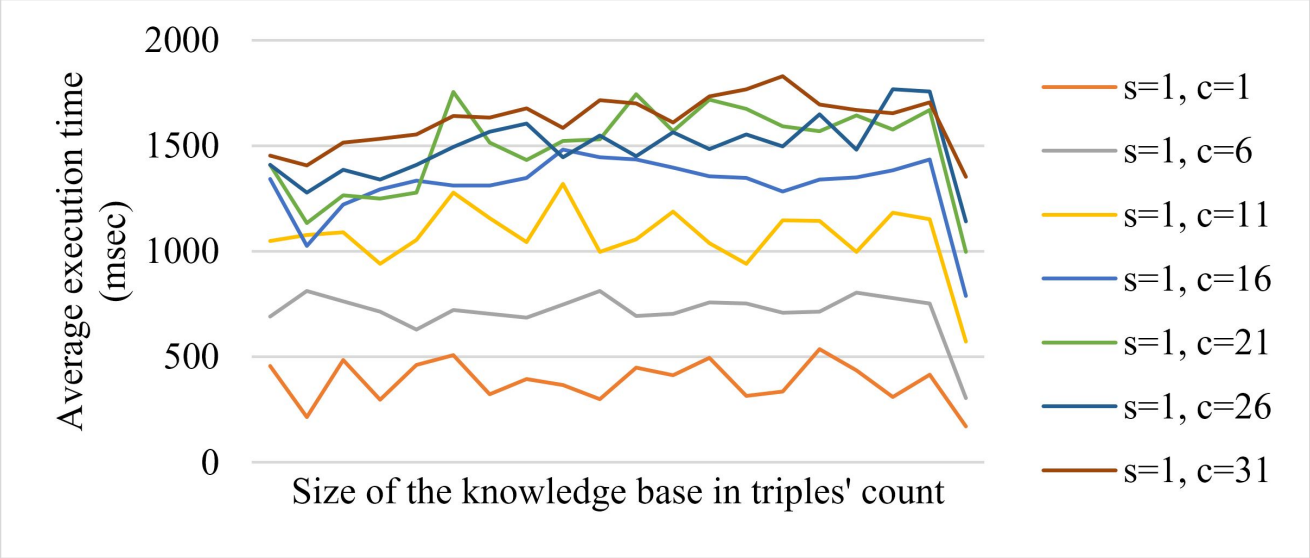

Fig. 16 Average execution time under several conditions

Cite as:

Modoni, G. E., Veniero, M., Trombetta, A., Sacco, M., & Clemente, S. (2018). Semantic based events signaling for AAL systems. *Journal of Ambient Intelligence and Humanized Computing*, *9*(5), 1311-1325.

https://doi.org/10.1007/s12652-017-0534-0

PRE-PRINT Version