Edge Computing and Social Internet of Things for large-scale smart environments development

Franco Cicirelli, Antonio Guerrieri, Giandomenico Spezzano, Andrea Vinci
CNR - National Research Council of Italy
Institute for High Performance Computing and Networking (ICAR),
Rende (CS), Italy
{cicirelli, guerrieri, spezzano, vinci}@icar.cnr.it

Orazio Briante, Antonio Iera, Giuseppe Ruggeri
University Mediterranea of Reggio Calabria
DIIES Department
Reggio Calabria, Italy
{orazio.briante, antonio.iera, giuseppe.ruggeri}@unirc.it

***Abstract*—Large-scale Smart Environments (LSEs) are open and dynamic systems typically extending over a wide area and including a huge number of interacting devices with a heterogeneous nature. Thus, during their deployment scalability and interoperability are key requirements to be definitely taken into account. To these, discovery and reputation assessment of services and objects have to be added, given that new devices and functionalities continuously join LSEs. In spite of the increasing interest in this topic, effective approaches to develop LSEs are still missing. This paper proposes an agent-based approach that leverages Edge Computing and Social Internet of Things paradigms in order to address the above mentioned issues. The effectiveness of such an approach is assessed through a sample case study involving a commercial road environment.**

Keywords*—*Large-scale Smart Environments; Social Internet of Things; Edge Computing; Agent Computing; Design Methodology.

I. Introduction

Large-scale Smart Environments (LSEs) are pervasive and distributed systems covering a wide geographical area [1], [2] and characterized by a huge number of possibly heterogeneous, interacting IoT devices. Their deployment aims to provide enhanced cyber-physical services to its users/inhabitants. Since new devices can be added to and removed from LSEs, they are recognized as high-dynamic systems. In addition, such systems are required to be capable to evolve their functionalities over time, according to changes in their finalities. As a consequence, they should support the possibility to add, update, and remove functionalities depending on the available devices and services. Within an LSE, besides *native* objects, which are directly deployed, owned, and managed by a specific LSE, other two kinds of entities are considered, namely *foreign* and *external* objects, which are better specified in the following. The first ones are objects that can enter and exit an LSE and do not belong to the LSE itself; private mobile devices are typical example of this type of objects [3]. The other ones are located outside the LSE, and are usually able to provide general purpose exploitable functionalities. In the considered open and dynamic scenario, issues such as entity discovery, trustworthiness, interoperability, data processing and management need to be addressed. Furthermore, it is of paramount importance to have methodological guidelines and tools to foster the development of LSEs by dealing with their complexity.

In this paper, an agent-based approach is proposed, which takes into account the highlighted issues. Aspects related to data management and processing are addressed by coupling the cloud computing paradigm [4] with the edge computing one [5], [6], implemented through the agent metaphor [7]. All of this provides an LSE with a huge computational power, exploitable for high-demanding processes, and a distributed elaboration capability supporting reactive and location-dependent processing. The other mentioned issues are addressed by leveraging the new Social Internet of Things (SIoT) computing paradigm [8], which fosters service discovery, resource visibility, object reputation assessment, and source crowding in the Internet of Things (IoT).

The paper also provides a set of guidelines for proper defining the roles and the interactions among all the entities involved in an LSE. In accordance with the proposed approach, as a reference platform to develop LSE, we adopt *iSapiens* [9], [10], which is an agent-based platform specifically thought for developing social and pervasive cyber-physical systems.

A preliminary version of our proposal has been presented in [2]. This paper goes further by (i) introducing a novel visual notation suitable for identifying the components of an LSE, (ii) describing a more comprehensive use case related to a Smart Commercial Road, and (iii) presenting a wider performance evaluation campaign to show the effectiveness of the proposed approach and to assess the performance of its social capabilities.

The remainder of the paper is organized as follows. After discussing some related works in Section II, the Social IoT is introduced in Section III. Section IV details the proposed

This is the post-print of the following article: "Edge Computing and Social Internet of Things for large-scale smart environments development " Article has been published in final form at: ieeexplore.ieee.org/document/8115262. DOI:10.1109/JIOT.2017.2775739

approach and summarizes the used reference platform. Section V describes the considered case study while the overall system performance in the reference scenario is assessed in Section VI. Finally, conclusions are drawn with an indication of future works.

II. Related Work

Smart Environments (SEs) and the IoT paradigm [11], [12] share the common vision of considering a pervasive presence in the environment of a variety of things/objects that are able to interact and cooperate with each other with the aim of creating new applications/services and reaching common goals [13]. In this context, the research and development challenges in creating a smart system are enormous [14], [13].

To date, many types of SEs [15] along with platforms and middlewares for their realization [16], [17], [18], [19] have been proposed. Such platforms often present limitations regarding interoperability and scalability towards LSEs, since they are usually designed to realize SEs and do not consider to jointly use multiple enabling technologies like cloud computing [4] and fog computing, and social capabilities [20]. Moreover, specific research-challenges in developing LSEs are also related to object discovery/use in a high-dynamic scenario where new devices can enter, move inside, or leave the system [8].

Many approaches to implementing SEs are known in the literature, and each of them takes into account specific issues in developing such systems. The interaction occurring among small bluetooth mobile devices are exploited in [21] to design a framework suited to implement intelligent SEs, allowing on-the-fly exploitation of the wireless devices available in the environment. This framework does not provide mechanisms to support proactivity in a deployed SE. The work in [22] focuses instead on extensibility. In particular, a robust framework is introduced, which favors system extensibility and supports the creation of specific context-aware applications. In [23] a distributed middleware is presented, capable of controlling resources located in physical spaces. Differently from other approaches, here an SE is built as a distributed system. The concept of *active spaces* is introduced with the goal of modeling programmable environments where all the considered devices are connected. A framework for developing context-sensing and context-aware applications is presented in [24]. The framework provides valuable abstractions usable for the design of various types of SEs. Authors of [17] propose a smart-home context-aware middleware. Their objective is the development of smart home environments by sharing contextual information among the entities of the system. Profiling issues are instead accounted for in the *Syndesi* framework [16], which is suited for the creation of personalized SEs that use wireless sensor networks. *Syndesi* is capable of profiling people so as to perform control actions on the basis of each obtained profile.

In order to move from SEs towards LSEs, it is possible to consider the Smart City scenario [25] which represents a hot topic in the current research landscape. Several platforms for the realization of Smart Cities have been already implemented. Anyway, also in this case, each of them only focuses on specific issues. As an example, the *Sentilo* platform is proposed in [26]. It is an open source IoT-based platform designed to manage sensors and actuators in a Smart City. The platform fosters openness and an easy interoperability among system components. *Sentilo* uses Cloud Computing and Big Data tools to collect, store, and elaborate data from distributed sensors so providing Smart City extensibility. Authors of [27] introduce an IoT middleware that has been implemented in the context of the EPIC (European Platform for Intelligent Cities) project. Such middleware has been designed to address issues regarding heterogeneity, interoperability, extensibility, and (re)configurability. In [28] a Smart City platform based on Big Data analysis to achieve extensibility is introduced. The platform relies on an architecture which provides three layers devoted to (i) collect, analyze, and filter data; (ii) aggregate data to infer knowledge; (iii) provide users with an interface to gather elaborated data. The Smart Connected Communities are instead introduced in [29]. The objective is the extension of the Smart City concept by considering specific issues such as livability, preservation, revitalization and sustainability of the urban areas to enhance. Another difference with respect to the Smart Cities is that a Smart Connected Community embraces not only a city area itself but also its neighborhood. An IoT-based multi-layer architecture for developing Smart Connected Community is proposed, the role and the importance of exploiting social capabilities is highlighted, but this is only considered as an open-issue deserving future work.

Despite the big research efforts devoted to the development of both platforms and middlewares exploitable to implement SEs and LSEs, a more moderate effort has been already made towards methodological guidelines and approaches to the design and realization of SEs/LSEs. The authors of [30] provide a methodology guideline for the realization of SEs by proposing a meta-model which captures both functional and data requirements. While this methodology can be scaled to LSEs, it is not conceived to consider both dynamic associations among SE components and social relations. The work in [31] introduces a three-layer architecture for SEs modeling. Here, an approach is introduced which allows configuring a smart home at design time. In [32]

a framework supporting the development of smart object systems is provided. Such a framework considers Unified Modeling Language (UML) metamodels to define different abstraction levels to support the analysis, design and implementation phases of a system.

Even though the above listed works propose very important solutions in modeling and implementing LSEs, they have the drawback that none of them consider dynamic associations among LSE components and social relationships. Moreover, to the best of our knowledge, the LSE solutions available in the literature either keep a centralized architecture [25], [33] or consider only a few devices usually scattered in a large area [34]. As novel and original contribution, this paper proposes an approach based on a joint exploitation of the distributed edge-computing paradigm along with the SIoT which takes into account the social and opportunistic relationships and interactions between entities involved in an LSE.

III. The contribution of the Social IoT

Social Internet of Things (SIoT) paradigm is a recent paradigm which has attracted the attention of the IoT research community thanks to the benefits deriving from the convergence of typical technologies and solutions of the IoT and the Social Networks domains [8]. Particularly suited to the purposes of this paper is the idea to leverage social network principles to facilitate the *management* of the IoT [8]. The insurgence of such a social network [35] has proven to foster resource visibility, service discovery, object reputation assessment, source crowding, and service composition in the IoT. When an object is seeking an information provided by the multitude of its peers, it might be easily overwhelmed by the complexity of the performed search. However, if "social-like" relationships built between the objects are used to limit the search only to those nodes with mutual social relations, exactly in the same way as humans search over their Social Network platforms, then the complexity and the time duration of the search could be drastically reduced [8]. Furthermore, as human social networks may be exploited to assess the trustworthiness and reputation of each member [36], [37], [38], [39], the social relationships established between objects can be also used to increase the trustworthiness in the exchange of each piece of information provided by the devices in the network. Just as humans leverage the opinion of their friends, the objects may leverage the experience and the opinion of other objects which share with them a social relation. It has been proven that such an approach allows isolating almost any malicious device in a network [40].
So far, five types of relationships have been defined for this new paradigm [8]:

1) *Ownership Object Relationship (OOR)*: relation bounding heterogeneous objects belonging to the same user;
2) *Parental Objects Relationship (POR)*: relation established between homogeneous devices produced by the same manufacturer and belonging to the same production batch;
3) *Co-Work Objects Relationship(C-WOR)*: relation set up between objects cooperating towards the provision of the same IoT application;
4) *Co-Location Objects Relationship (C-LOR)*: relation bounding devices that are always used in the same place;
5) *Social Object Relationship (SOR)*: relationship established between objects that come into contact, either sporadically or continuously, because their owners come in touch each other.

Besides the above mentioned relationships, as in most human social networks, it is also possible to define special groups that include devices bounded by some common features or finalities. This is not a mere theoretical concept, as an open source platform called *SIoT Platform*[1] is available.

[1]Social Internet of Things (SIoT), www.social-iot.org

IV. An Approach for the creation of social large-scale smart environments

This section describes the proposed approach for the creation of LSEs. First, a categorization of the Smart Objects involved in an LSE is given, then the approach is detailed along with a visual notation for the design of LSEs. Finally, the iSapiens platform, which is the social and pervasive platform suggested for developing LSEs, is summarized.

A. A Smart Objects Categorization

All the Smart Objects which can be modeled in an LSE are grouped into three categories, namely *native*, *foreign* and *external* objects.

- *Native objects* are smart objects systematically deployed for the creation of a specific LSE, and are supposed to be interconnected with each other through a dedicated infrastructure/platform. It is assumed that all the native objects are managed by a single administrative entity, thus they are implicitly trusted.
- *Foreign objects* are objects independently conceived and deployed with respect to the creation of a specific LSE, but that share, either permanently or occasionally, their location with the LSE itself. Examples of foreign objects that get occasionally in touch with LSEs are smart-vehicles passing through the LSE, and smart-phones owned by people visiting the environment. Foreign objects that permanently coexist in the same space of the LSE can be SmartTVs, Smart Bulbs, or Nest Learning Thermostats owned by third-party and placed within the LSE; these can directly interact (sensing and actuation) with the enhanced physical environment.
- *External objects* are also independently conceived and deployed with respect to the creation of a specific LSE, but never share their location with the LSE itself and, thus, never get in touch with native objects. Examples of such a kind of objects are smart meteorological stations or smart traffic lights located outside the LSE. In particular, an external meteorological station can provide information about temperature and humidity, which can be usefully exploited to properly manage a native HVAC system of an LSE. Similarly, an external smart traffic light can be notified to limit the traffic entering the LSE area.

Foreign objects and *external objects* are supposed not to be directly connected to the same platform which manages the *native objects* belonging to an LSE. Indeed, the platform can be even unaware of their presence. Even if non-native objects can provide useful information for the management of an LSE, they are managed by third parties and their trustworthiness has to be assessed. For these reasons, foreign and external objects are exploited, discovered, and accessed by an LSE in an *Opportunistic/Ad Hoc* manner [41].

B. Description of the approach

Our proposal for the creation of LSEs is based on two pillars: (i) the Edge Computing paradigm [5], and (ii) the

Social Internet of Things (SIoT) [8]. In our vision, an LSE is constituted by a dynamic set of distributed computational nodes, directly spread in the environment. The computational nodes, which interact with each other, are located close to the devices and the data sources which they manage. Such an architecture is compliant with the Edge Computing paradigm, which fosters system reactivity and a reduction in the required communication bandwidth (as data are processed close to their sources, and only aggregated information is propagated across the system). Besides this Edge layer, a Cloud Computing layer is also considered. The latter is exploited when there is the need to execute tasks that are computational intensive or require a global view of the LSE.

At both layers, the deployment of LSEs can be achieved by using the agent metaphor [7]. The agent metaphor suits well with the modeling, simulation, and implementation of distributed, dynamic and complex systems [42], [43], [44]. This is because agents naturally support interaction among independent and autonomous entities [45], thus favoring emergent system properties which characterize distributed and complex systems. Moreover, agents can be dynamically created into or removed from a system, so as to support dynamism.

Finally, Edge Computing can be developed by using agents. This is possible since agents can choose to execute tasks on the computational node closest to the data sources.

Here, agents are used to implement the application logic of the LSE cyber-part. Anyway, in an LSE, cyber and physical parts need to interact with each other and with the external environment, where the LSE is located. Since *Native objects* are a-priory known, they can be mapped into the used cyber-space through a specific platform and are trusty. The interactions with the other kind of objects, namely foreign and external, is more difficult because (i) they have to be discovered, (ii) their trustworthiness should be assessed, and (iii) the data of interest have to be retrieved.

Interoperability issues can be overtaken thanks to the SIoT paradigm, that aims at managing IoT complexity by leveraging proactive objects interactions suitably built to mimic the human sociality model. While other interesting approaches such as [41] use the social networks of the humans to favor object interactions, SIoT exploits social-aware devices that are programmed to autonomously establish social relationships so to create a social network of objects. The usage of such a social network [35] favors resource visibility, service discovery, object reputation assessment, and service composition in the IoT. Moreover, social relationships among smart objects allow evaluating the trustworthiness of each object and to rate the information it provides.

The proposed approach promotes the use of an abstraction layer that hides the heterogeneity of the native objects (e.g., actuator and sensing devices) in terms of offered functionalities and communication protocols. So, the agents will be able to use a uniform API to access native objects.

Since in an LSE, interactions are very important and can happen among a very large number of entities, there is a basic need for defining and modeling how these entities can discover each other. Two different solutions are proposed: the former is based on a common registry (i.e., Yellow Pages), while the latter relies on discovery mechanisms exploiting opportunistic relationships among entities. The first solution allows the discovery of *Native objects* which, when installed in an LSE, are immediately included in a common register so that their presence can be known by everyone. The second solution has been conceived for the discovery, based on social properties, of *foreign* and *external objects*. While moving, mobile *foreign objects* can meet *native* and *external objects* so to create social ties with them and construct a social network of objects [8] which can be used both to discover new devices and to rate their trustworthiness [40].

In Figure 1, a multi-dimensional schema depicting the proposed design approach is reported. For each block, representing an entity, its possible location is shown (i.e., the *Cloud* or the *Edge*) together with the mechanisms which can be exploited to discover other entities (i.e., *direct knowledge*, *yellow pages*, or *social relations*).

Three main agent categories are defined:

- *Mirror agents* are devoted to map the functionalities exposed by the native objects into location independent functionalities offered by agents. In order to favor the modularity of applications, it is recommended to implement a specific agent for each functionality provided by a single device. Mirror agents run on the computational node, which is directly connected to the controlled devices they have to manage. Mirror agents can enhance the wrapped device by enforcing, for example, access policies and/or negotiation procedures for guaranteeing exclusive or time-based use of such devices.
- *Boundary agents* manage the interactions with SIoT enabled *foreign* and *external objects* and third-party provided Internet services. Boundary agents are designed to connect such (not-native) entities to the agents world.
- *Concept agents* are conceived to model the main functionalities of an LSE by taking into account modularity, separation of concerns and maintainability issues. Concept agents aims at separating different issues related to (i) the definition of a set of desired goals, (ii) the acquisition of the related knowledge, and (iii) the planning of a set of actions for achieving them. Concepts foster reasoning by exploiting domain related notions. They abstract from physical measures and actuation on the controlled environment. For instance, in a Smart Home application, a goal concept named “Indoor Wellness” can coordinate the knowledge concept of “Indoor Climate” and the plan concept “Air Conditioning Controller” in order to ensure the healthiness in the house. The relationships among concepts create a concept hierarchy having at its top level the Goal agents and at the bottom Mirror and Boundary agents. Higher level concepts are built by exploiting the behaviors of the low-level ones.

C. Visual Notation

In this section, we introduce a simple yet effective visual notation suitable for identifying the components of an LSE, described in the previous section, and the interactions among

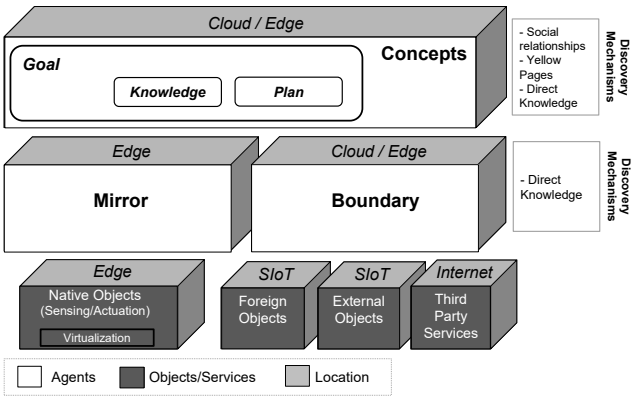


Fig. 1. A multi-dimensional schema of the proposed design approach

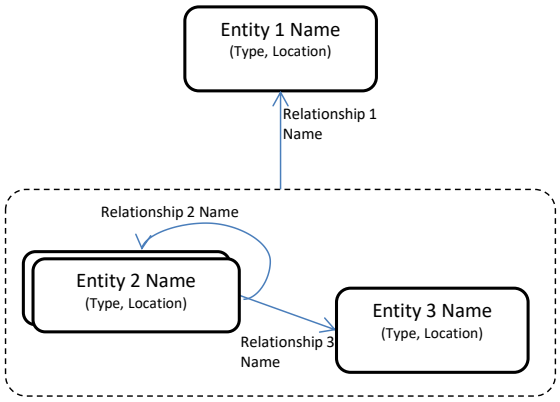


Fig. 2. Entities and relationship diagram summary.

them. Figure 2 shows an example, which summarizes all the notation elements.

An entity type of the system is represented by a rounded rectangle. All entity types must have attributes stating their name and role (e.g., a Concept, a Mirror, or a Boundary entity). An agent entity also has an attribute describing the location (edge or cloud) of the related agent.

A directed labeled arrow between two entity types models an interaction requirement among instances of them. An interaction requirement is defined as a triple $< StartEntity, EndEntity, RelationLabel >$, and means that an instance of *StartEntity* needs to interact with a set of instances of *EndEntity* defined by the relation label. A relation label can be either a social or a Yellow Pages query, or a direct knowledge relation. If a relation label refers to a social query, it is highlighted in an italic-bold font. An empty label is used for a direct knowledge relation. A self-loop models an interaction requirement among instances of the same entity. In this case, the entity itself is depicted as a stack of rounded rectangles.

A set of entities can be grouped within a dashed box. An arrow between a group and an entity is equivalent to a set of arrows which links all the group component with the entity itself, having the same direction and relation label. Similarly, an arrow between two groups is equivalent to a set of arrows linking all the entities of the first group with all the entities of the second one.

D. iSapiens: a Platform for Pervasive and Social LSEs

Among the possible platforms and frameworks suitable for the creation of LSEs (see Section II), we take as a reference the iSapiens platform[2] [10], [9], as it owns a set of features which perfectly comply with the proposed approach. iSapiens is a distributed agent-based platform for the development of pervasive and social cyber-physical systems, like smart homes and smart cities. Figure 3 shows the iSapiens architecture.

An iSapiens-based smart environment includes a set of networked computing nodes, distributed in the controlled environment. Each node is close and directly connected (by wires or wirelessly) to the physical resources, e.g. sensor, actuators, or more complex smart objects, which it manages, according to the edge computing paradigm. The edge computation layer is paired with a cloud/internet service layer, for the offline execution of computational demanding tasks [46]. Each computational node hosts the iSapiens middleware, which contains an Agent Server, and a Virtual Object Container.

The Agent Server permits the execution and the management of software agents, and supports the communication among them, achieved through the exchange of asynchronous messages. The platform provides two mechanisms for the creation and management of relationships among agents, which are: (i) *acquaintance messages*, used to establish direct knowledge among agents, and (ii) a Yellow Pages service, which is exploitable to manage agent search and discovery in the distributed system. An Agent Server can be also executed on the cloud layer, if required.

The iSapiens platform provides mechanisms for the exploitation of the features offered by the SIoT platform (see Section III), so as to achieve social capabilities.

The SIoT platform furnishes functionalities for: (i) "Relationship management", i.e., to establish, update, and remove objects relationships, (ii) "Service discovery" permitting the search for objects which provide specific services, (iii) "Service composition", allowing objects to interact with each other, and (iv) "Trustworthiness management" for the assessment of the information provided by other members. Whenever a new iSapiens-enabled object is available, it is also registered to the SIoT platform, and its social profile is created.

An SO is responsible to constantly update the object profile on SIoT by periodically updating all the relevant information about the object it represents. To carry out its tasks, an SO exploits the information provided by the *Social Senser (SS)*, which is installed on the real device and periodically overhears the transmission over the Wi-Fi and Bluetooth interfaces to search for other nearby social-enabled devices. This information, once forwarded by the social object to the SIoT Platform, is used to determine mutual contacts between social objects and, eventually, to create a social tie between those objects. An SO is also responsible to look for a given data, by analyzing

[2]http://domus.icar.cnr.it/isapiens/isapiensAThome.html

the information from friends in the social object graph, and to send queries to the SIoT platform or answering to specific data requests. All the SOs residing in a specific node are managed by a *Social Objects Container*.

All the devices connected to an iSapiens node are abstracted as *Virtual Objects* (VOs).

VOs are managed by the Virtual Object Container of the iSapiens middleware. The VO abstraction permits hiding the device heterogeneity, in terms of communication protocols and drivers required to manage the physical devices, by providing agents with an API for the transparent access to the functionalities of the *native objects*.

It is worth noting that all the components introduced, either a whole computing node, an agent, a virtual object, or a physical device, can be dynamically added, removed, or updated. Furthermore, by exploiting both yellow-pages and acquaintance messages, the relationships among agents can be also dynamically updated. All the presented features allow iSapiens to suitably develop large-scale, extensible and pervasive applications.

V. Case Study: Safety and Security in a Commercial Road

Here, a case study showing the effectiveness of the introduced approach is proposed. In particular, we consider a commercial road augmented by an ICT system capable to react to people physical malaise, malicious intrusions, and fire event detection within the *commercial activities (CAs)*. The overall goal of the proposed LSE is the improvement of both the safeness and the security within the instrumented area.

The considered area is made up of a set of buildings hosting several commercial activities such as shops, restaurants, and cafés. Most of the CAs are equipped with security and safety systems that include SIoT enabled devices. A set of CAs exploits a Building Management System (BMS) developed by using iSapiens whose role is addressing both security and safety issues.

More specifically, two iSapiens-based subsystems can be highlighted:

- the *security subsystem* has been conceived to detect burglaries and intrusions into commercial activities through a set of sensors scattered in the CAs themselves. These sensors detect the presence of people, the opening of doors/windows, and the glass breaking of entrances. Moreover, intrusions and aggressions can be signaled through a SafetyApp deployed on the mobile phones owned by the CA workers. The system reacts by starting deterrent actions, such as turning on sirens or flashing alarms, and asking the neighborhood (dynamically discovered through SIoT) to switch on their flashing alarms;
- the *safety subsystem* has been conceived to react to malaises, fire events, or nearby explosions. In particular, a physical malaise is signaled through the SafetyApp. The fire alerting component aims at detecting the occurrence of fire in a CA or in its neighborhood, and at guiding the people in a commercial activity to the closest available emergency exit. The explosion-alerting component has the purpose to detect possibly dangerous explosions in the nearby area. Each commercial activity is equipped with smoke, temperature, and humidity sensors, that are required to detect fires, and with a set of exit path lights, used to signal the occurrence of a dangerous event and to highlight a safe exit path. Moreover, during the evacuation, all the doors can be unlocked.

A common feature of both subsystems is the possibility of searching for the closest caregivers, such as doctors, guards or trusted people, through the SIoT component, when reacting to dangerous events.

In the remainder of the Section, we describe how the proposed approach is used for the design of the case study. An analysis of SIoT and iSapiens capabilities is provided in Section VI.

We assume that each iSapiens-enhanced CA is equipped with an iSapiens node consisting of a single-board computer like a Raspberry Pi 3, while sensors and actuators are connected through a ZigBee network. All the computing nodes run the same set of agents, so they are functionally equivalent. Besides the network of edge computing nodes, a remote cloud node is also considered so as to offer shared services.

Figure 4 shows the design of the system performed by means of the visual notation introduced in Section IV-C, where rounded rectangles model entities within the system, and arrows model interaction requirements. The labels of the arrows represent queries issued for discovering the entities engaged in an interaction. A bold italic label identifies SIoT queries, while a plain one identifies queries issued on the yellow pages. In the Figure, *Mirror* and *Boundary* agents are not reported for simplicity, as they do not add new functionalities to the considered system. All the agent marked with the "edge" label are considered as running at least in one computational node of each CA, while the "cloud" labeled agents run on the cloud.

The description of the agents used to implement the system is reported in Table I whereas Table II describes the devices involved and their categories (native, foreign, external, and third party service).

The design of the system exploits the approach described in Section IV. For each agent in the system, its location and the abstraction level it belongs to are specified. In particular, the system objectives are pursued by the *Safeness* and *Security* agents by a proper coordination of their associated *plan* and *knowledge* agents.

SIoT plays a key role in the discovery process and in supporting interactions between the iSapiens system and the external and foreign devices. In particular, SIoT is exploited (i) to interact with external smoke, temperature and humidity devices to assess the presence of fire events near a certain CA; (ii) to detect a possible dangerous explosion in the considered commercial area by exploiting the microphones from the SIoT-enabled mobile phones; and (iii) to reach the closest caregivers, as guards or doctors, in case dangerous events occur. It is worth noting that all the communications between LSE entities and native objects are mediated by iSapiens, by exploiting either messages or the virtual objects' abstraction. In a complementary manner, all the communications towards

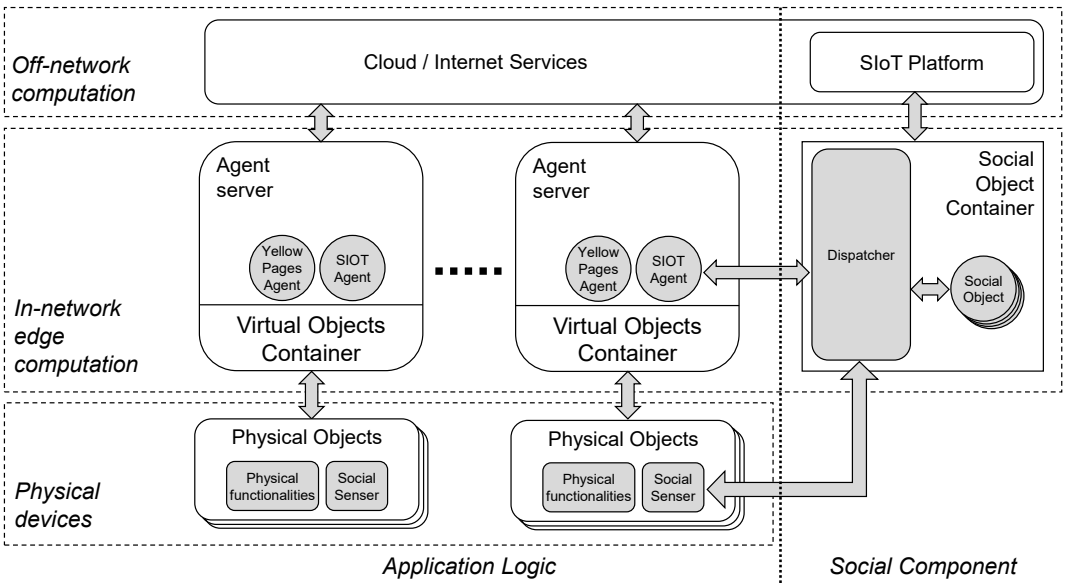


Fig. 3. *iSapiens* platform: decomposition in computational layers.

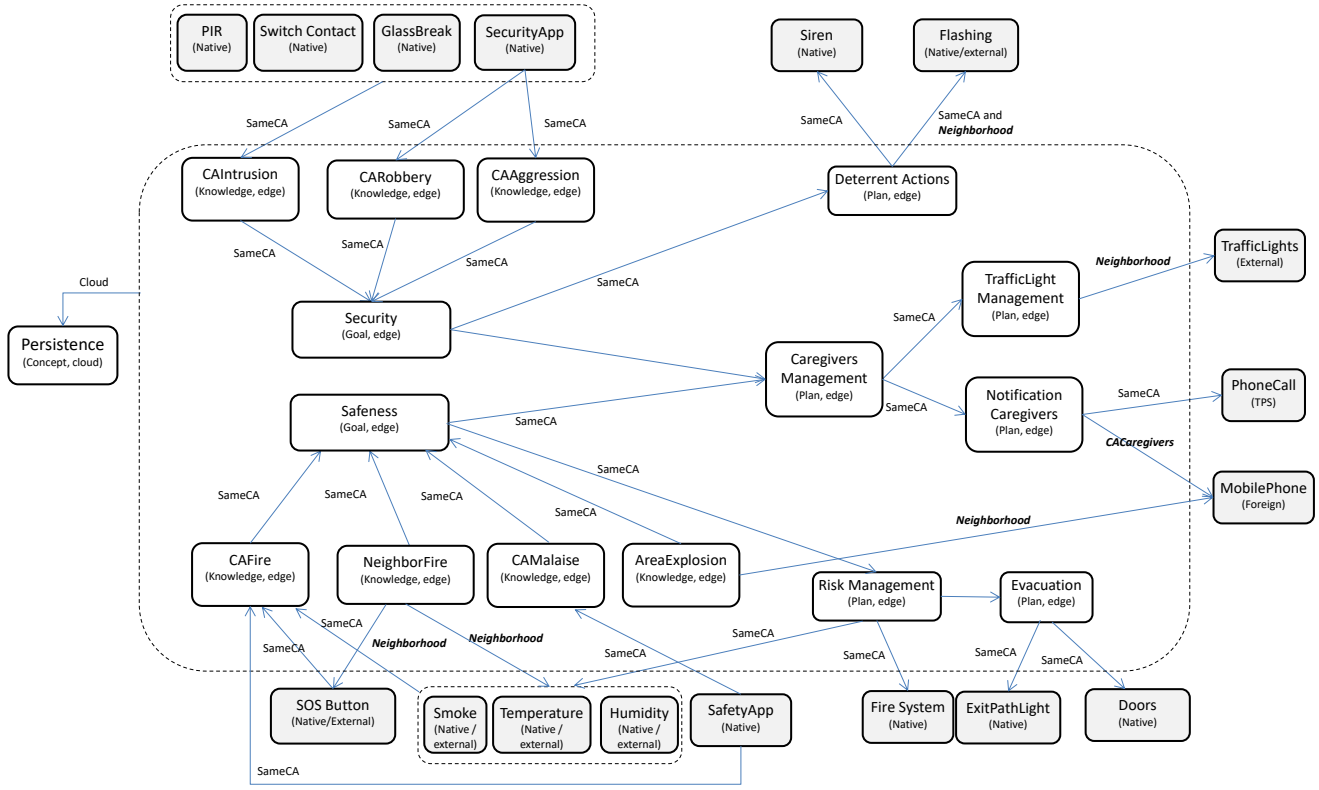


Fig. 4. Entities and interactions involved in the considered case study for the Building Management System of the Commercial Activities.

TABLE I. A DESCRIPTION OF THE AGENTS CONSIDERED IN THE CASE STUDY.

Agent Name	Agent Type	Description
CAIntrusion	Knowledge	Such agent monitors the environmental data provided by the sensors connected to the same Commercial Activity (see the SameCA label on the arrow). Moreover, an intrusion can be signalled through the SecurityApp owned by the commercial activity workers. A detected intrusion event is signaled to the Security agent running on the same node.
CARobbery	Knowledge	Such agent is designed to receive alerts from the SecurityApp running on the mobile phone owned by the Commercial Activity workers. A detected robbery event is signaled to the Security agent running in the same CA.
CAAggression	Knowledge	Such agent is designed to receive alerts from the SecurityApp running on the mobile phone owned by the Commercial Activity workers. A detected aggression event is signaled to the Security agent running in the same CA.
CAFire	Knowledge	Such agent monitors the environmental data provided by the sensors deployed in the same CA (see the SameCA label on the arrow). A detected event of fire is signaled to the Safeness agent, which manages the same CA.
NeighborFire	Knowledge	All the NeighborFire agents gather data from neighbor SIoT enabled devices to estimate a possible event of fire in the neighborhood. In the case of fire, they notify the Safeness agent in the same CA.
AreaExplosion	Knowledge	All the AreaExplosion agents gather noise level data from foreign SIoT enabled mobile phones to infer a possible explosion or roar event close to the mobile phone itself. If it is the case, they notify the Safeness agent in the same CA.
CAMalaise	Knowledge	Such agent is designed to receive alerts from the SafetyApp running on the mobile phone owned by the Commercial Activity workers. A detected malaise event is signaled to the Safeness agent running in the same CA.
Caregivers Management	Plan	The agent is activated by both the Security and the Safeness agents. It manages the NotificationCaregivers agent to notify the proper caregivers and the TrafficLightManagement agent to regulate the traffic in the area.
Notification Caregivers	Plan	NotificationCaregivers agents take care to notify all the caregivers of specific commercial activities through SIoT (e.g. guards, ambulances, or doctors) and, in the case it is necessary, they make PhoneCalls to firefighters, police, or hospitals.
TrafficLight Management	Plan	It manages the traffic light system in the neighborhood of the commercial activity to favor the caregivers arrival.
FireManagement	Plan	The FireManagement agents are activated by the Safeness agents residing in the same CA. They control the Fire System taking into account the data provided by fire-related sensors deployed in the CA. Moreover, they can drive the Evacuation agents.
Evacuation	Plan	The Evacuation agents are activated by the FireManagement agents. The Evacuation agents control both the ExitPath-Lights to drive people outside a commercial activity and doors' locks.
DeterrentActions	Plan	It manages the Siren and the Flashing alarm in its CA and, if needed, it can reach, through SIoT, the nearby Flashing Alarms and request for their activation.
Safeness	Goal	It takes decisions about the safeness of a specific situation, and, on need, can coordinate the CaregiversManagement and FireManagement agents in order to react to a fire event, to a nearby explosion, and to the malaise of a person in the CA.
Security	Goal	It takes decisions about the dangerousness of a specific situation, and, on need, can coordinate the CaregiversManagement and DeterrentActions agents in order to react to intrusions, robberies, or aggressions occurring within a CA.
Persistence	Concept	All the agents send their status changes to this agent, which takes care to manage the log of all the CAs.

TABLE II. DESCRIPTION OF THE DEVICES AND THE SERVICES INVOLVED IN THE CASE STUDY.

Device/Service Name	Device/Service Type	Device/Service Description
PIR	Native	The Passive Infrared sensors detect movements in a CA
Switch Contact	Native	They are used to detecting the status (opened/closed) of doors and windows
Glass Break	Native	These sensors are used to detecting glass breaking of windows and doors
SecurityApp	Native	It is used by CA workers to signal to the Security subsystem about intrusions, robberies, and aggressions
Siren	Native	It is used for discouraging purposes by the local security system
Flashing	Native/External	It is used for discouraging purposes by the local security system. Moreover, flashing devices in the neighborhood can be activated through SIoT
Traffic Lights	External	They regulate the traffic in the commercial road.
Mobile Phone	Foreign	They are all the SIoT enabled mobile devices belonging to the people attending the commercial area. Such devices are also used to detecting nearby explosions or roars (e.g. gunshots)
Phone Call	ThirdPartyService	This service is used to contact public emergency services such as firefighters, police, or hospitals
Doors	Native	They can be unlocked so as to favor the evacuation of a CA
Exit Path Light	Native	They can be used to signal the most convenient path to the exit
Fire System	Native	It is used to help in the extinguishing or preventing the spread of fire in a CA
SafetyApp	Native	It is used by CA workers to signal to the Safeness subsystem about a possible malaise of a person in the CA
Humidity, Temperature, Smoke	Native/External	They are used together to detect fire events within a CA or in its neighborhood
SOS Button	Native/External	It is used to signal a possible fire event in a CA

external and foreign objects occur instead by exploiting the capabilities offered by the SIoT platform. A common implicit ontology between iSapiens and non-native objects is assumed.

With respect to the considered case study, the social behavior of the system, along with its performance, will be evaluated in Section VI.

VI. PERFORMANCE EVALUATION

Although the idea of using information provided by *foreign* and *external objects* seems promising, it still remains to assess how effective a discovery procedure based on SIoT can be. Also, it must be evaluated (i) the performance achieved by an information collection mechanism, which leverages the social ties among the objects, and (ii) the advantages deriving from the inclusion of the *foreign* and *external objects* into the LSE through SIoT to enrich the sensing capabilities of an iSapiens based system. We, therefore, run a series of tests to shed light on these aspects. Specifically, in our simulation campaign we consider a city block of $2[Km^2]$, depicted in Figure 5, which roughly corresponds to the city center of Reggio Calabria (IT). This area includes the main institutional buildings, over 100 commercial activities and recreational places, such as shops, restaurants, and theaters. Therefore, it matches well to an LSE

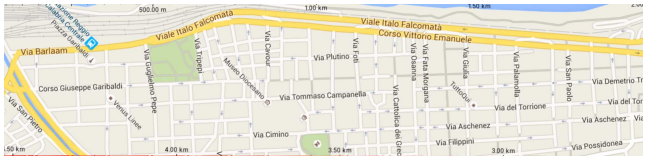


Fig. 5. The area considered in our simulations.

like the one described in section V.

We assume that 10 out of the 100 places hosting a commercial activity are managed by a Building Management System (BMS) based on iSapiens. We further consider that 4 BMSs are managed by a single administrative entity (e.g. a franchising firm); thus they represent the *native objects* in our scenario. The remaining 6 BMSs belonging to third parties represent the *external objects*. Obviously, we consider both, the *native* and the *external objects*, standing in fixed positions at the commercial or recreational venues in the area of interest. Also, we assume that the *native objects* have a mutual *Ownership Object Relationship (OOR)* [8].

By using the well-known SWIM software [47] we simulate N, $N \in [50, 100, 150, 200, 250, 300]$, people moving within the area described above along with their personal devices that represent the *foreign objects* in our scenario. For the sake of simplicity, we consider each person bringing only one personal device always with her. While people move, their carried personal devices enter in contact (we assumed a sensing range of 10 $[m]$) and begin to create their SIoT network. By post-processing the output of SWIM through custom software written in *MATLAB*®, we monitor how the social network between the *native*, the *external*, and *foreign objects* evolved. Specifically, according to [35] we consider that two objects establish a *Social Object Relationship (SOR)* [8] after having experienced one or more contacts for a cumulative contact time of at least 10 $[min]$.

A. *Inclusion of the* foreign *and* external objects *in the LSE*

This set of simulations aims at investigating how the *foreign* and *external objects* are included in an LSE by means of the social ties they create with the *native* objects. To carry out our study, we assume that the *native objects* are switched on for the first time at $t = 0$ and that from this moment onward, they begin to develop their social network by experiencing one or more contacts with the *foreign objects*. Since all the *native objects* are already connected by OORs, once a *foreign object* has sealed a social tie with a *native object*, it is included in the social network of all of them. Obviously, an *external object* by definition can never directly get in touch with a *native* one; however it can establish a social tie with a *foreign object* already in the social network of a *native* one, and hence be included in it. Figure 6 shows how the number of *foreign* and *external objects* included in the social network of the *native objects* changes over time following the system start-up. There is no matter on what is the number of objects in the simulation scenario because, after a while, they are all included in the social network of the *native objects*. In a greater detail, Figure 7 shows the time required to include *foreign* and *external*

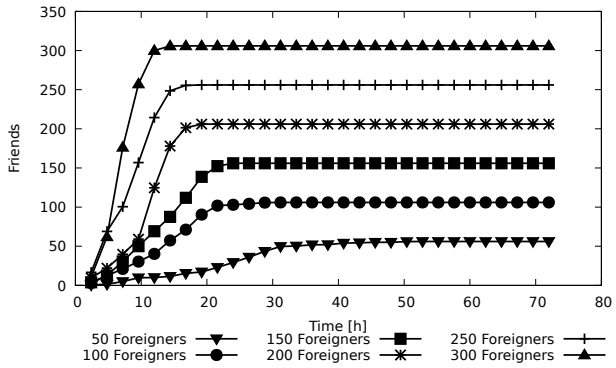


Fig. 6. Number of *foreign* and *external objects* included in the social network of the *native objects* after the system start-up, by varying the simulation time

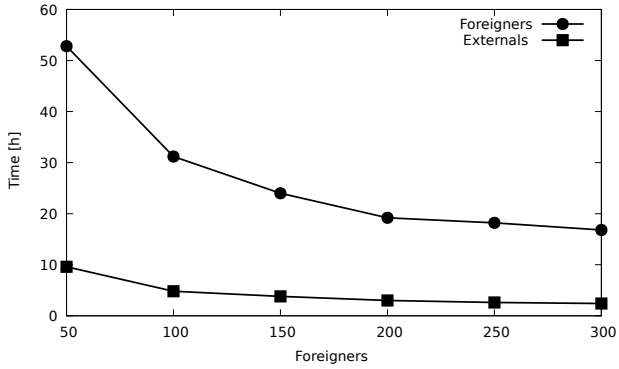


Fig. 7. Time to include in the social network of the *native objects* the totality of the *foreign* and *external objects*.

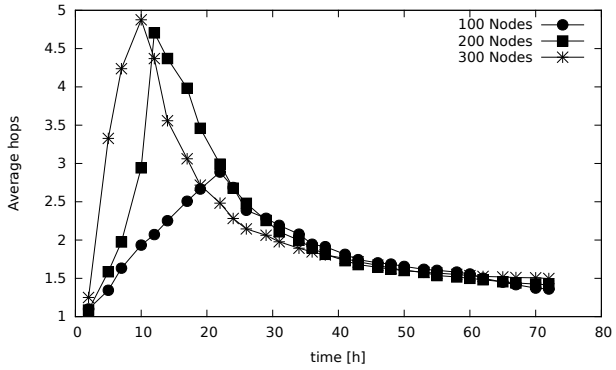


Fig. 8. Average distance in the social network between *foreign* and *external objects* with respect to their closest *native object*

objects in the social network of the *native objects*. In Figure 7 a pretty fast inclusion process is observed. All the *external nodes* are added to the social network in less than 10 hours while the inclusion of all the *foreign objects* can require up to about 52 hours. Also, the greater the number of *foreign objects*, the faster is the inclusion process. Figure 8 reports the average distance in the social network between *foreign*

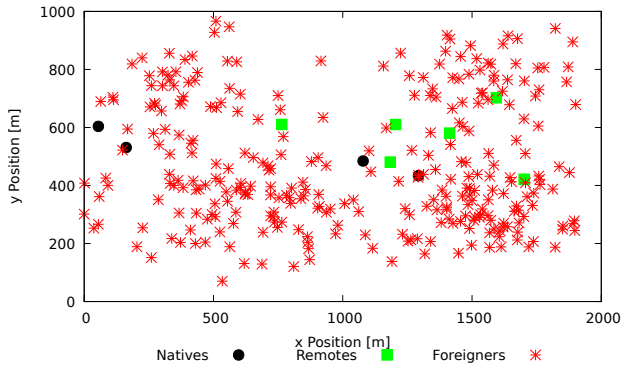


Fig. 9. An example of the *foreign objects* positioning.

and *external objects* with respect to their closest *native* one. It provides a justification of the different inclusion rates observed when varying the number of *foreign objects*. Initially, *native objects* have a few direct ties with other objects and most of the devices in their social network are connected through a chain of acquaintances. The higher the number of *foreign objects*, (i) the longer are the chains that can be made and (ii) the higher is the number of objects that can be quickly included in the social network. As time goes by, the *native objects* create an increasing number of social ties with foreign objects; as a consequence, the chain is shortened and the average hops in the social network decrease to about 1.5.

B. Sensing the LSE by leveraging the social capabilities

The second objective of our validation campaign was to assess the real effectiveness of SIoT in extending the sensing capabilities of the *native objects* based on iSapiens. To this aim, we focus our attention on the detection of a loud noise similar to the one caused by an explosion. We assume that all the devices in the scenario are equipped with a microphone and that they can reveal a loud noise. We made no assumption about the detection algorithm, except that it is effective only when the executing device falls within a radius of r meters from the noise, with $r \in [25, 50, 75, 100, 125], [m]$. Once the loud noise is observed, its detection is notified through the social network by following the chain of acquaintances departing from the detecting object. Thus, if a *native* object is in the social network of the detecting object, it will be also notified about the detection. In order to carry out our analysis, we suppose that at a given time $t^* > 0$ a loud noise is generated in a random position within the area of interest. We further assume that, at the same time t^*, *foreign objects* are randomly located in the simulation area. More specifically, they are located around a set of points of attraction (e.g. shops, restaurants, squares), as exemplary shown in Figure 9.

For each value of t^* we generate 10^5 loud noises and consider them detected when they fall within the sensing range r of one or more objects in the social network of the *native objects*.

The results are shown in Figure 10.(a). As a term of comparison, we also report the detection probability observed without any support from SIoT. Soon after its start-up, the social network of the *native objects* is small and the contribution of the *foreign object* in detecting the noise is reduced. However, as time goes by, more nodes are included in the social network and the detection probability quickly increases. It gets very close to 1 when a high number of *foreign objects* is considered. Obviously, the greater the number of *foreign objects*, the greater is the probability of detection.

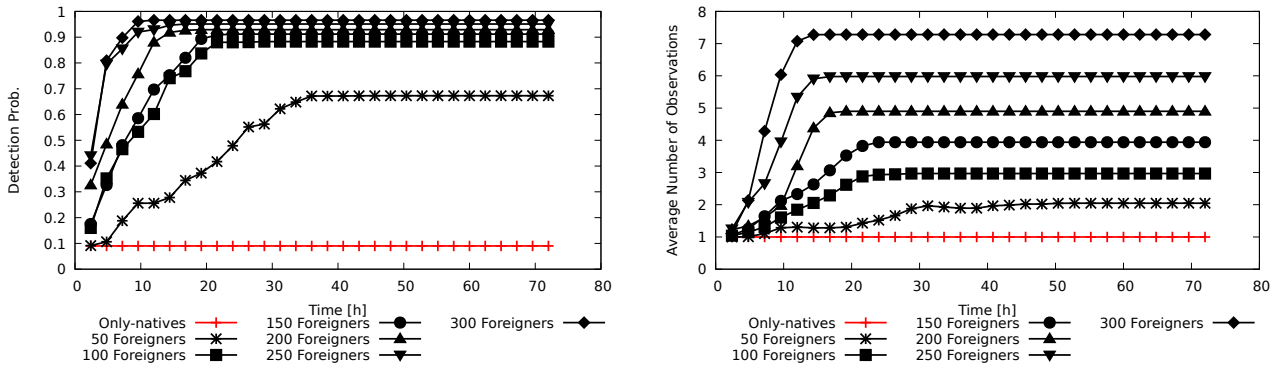
Relying on multiple observations of the same phenomenon may be very useful in several applications. As an example, multiple observations can be used to reject false alarms or to get a better estimation of the phenomenon itself. In our sample case, a loud noise can be heard by multiple objects in the social network and the *native objects* can use the multiple observations to smartly take a decision on the actions to undertake (e.g., by using ambient intelligence algorithms). Figure 10.(b), shows the average number of detections for each loud noise. Thanks to SIoT, this value can rise up to over 7 concurrent observations when increasing the number of *foreign objects*. Again, as a term of comparison, we reported the average number of observations obtained by considering only the *native objects* without the support of SIoT. In this case, we registered no more than a single detection of the noise.

Figure 11.(a) shows the detection probability obtained when considering 300 *foreign objects* and a variable Detection Range $[25, 50, 75, 100, 125]\, m$. As a reference, Figure 11.(b) reports the same probability by considering only the native objects and an equal detection range. As it is argued by comparing the two figures, whatever the sensing range is, the use of SIoT greatly improves the detection probability.

Whenever a loud noise is observed by a *foreign* or an *external object*, the detection is notified to the closest *native object* through the social network by following the chain of acquaintances departing from the detecting object. Propagating the information from an object to the next along the chain requires a signaling exchange between the involved objects and the SIoT platform[3]. Thus the propagation process takes some time to be completed. After a long run of measures on a real device we obtained an average latency for each signaling exchange of $351.5\,[ms]$. We, therefore, roughly estimate the detection latency as the number of hops separating the detecting object from the closest native one timed the average latency. Of course, this estimation does not take into account many aspects such as the always-changing communication channel, the traffic congestion, or the performance of the server hosting the SIoT platform. Nonetheless, it provides some information on a couple of interesting trends.

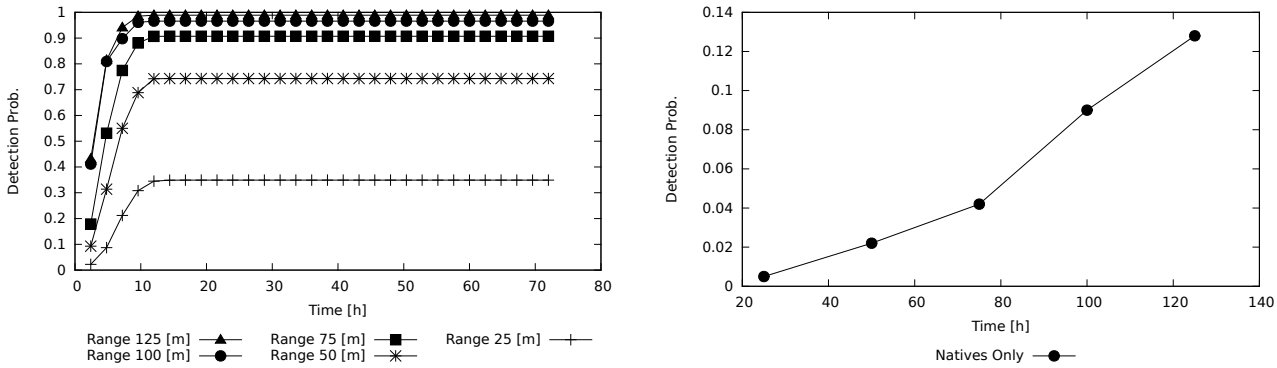
Figure 12, shows the results obtained in the case of 300 *Foreign Objects*. One observes that the shorter the detection range, the higher is the latency. This trend can be explained by considering that, with a short detection range, the probability that the noise is directly detected by a native object or one of its direct friends is low. On the contrary, an event is often detected by an object that is many hops away from the closest

[3]Social Internet of Things (SIoT) Platform. [Online] http://platform.social-iot.org



a) b)

Fig. 10. (a) Detection Probability, and (b) number of objects that detect the event, when varying the number of *foreign objects* in the scenario and the time. Detection Range 100 $[m]$.



a) b)

Fig. 11. Detection Probability considering: (a) 300 *foreign objects* and a variable Detection Range $[25, 50, 75, 100, 125]$ m, (b) only *native objects* and variable Detection Range $[25, 50, 75, 100, 125]$ m

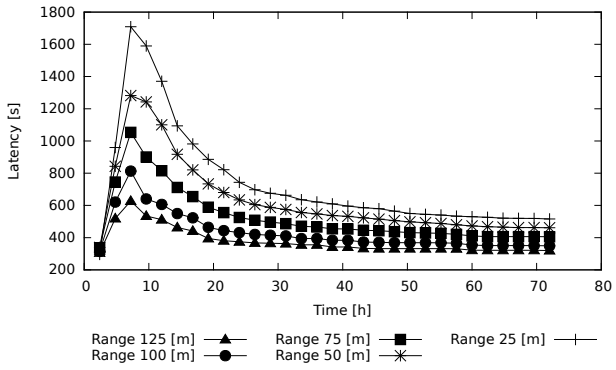


Fig. 12. Detection latency 300 *foreign objects* and a variable Detection Range $[25, 50, 75, 100, 125]$ m.

native one.

The second interesting trend is observed when focusing on the detection latency variation with the time elapsed since the *native objects* power up. In this case, the latency strongly depends on the average distance in the social network between the detecting object and the closest *native object* shown in Figure 8 and described in section VI-A. We may distinguish three different phases characterizing the variation of the Discovery Latency. In the first phase, just after the native objects power up, the latency is low. In this phase a few social relations have been already established and the chains of acquaintances are short. Under those conditions, the probability of detection is very low; however, in the case of a successful detection, the noise is usually detected by a *native object* or by an object connected to a *native object* by a very short chain of acquaintances, very often made by a single link. In the second phase, with the passing of time, new objects are included in the social network through long chains of acquaintances, and, although the detection probability increases, very often the chain of acquaintances connecting the detecting object with the closest native is quite long. Hence, the detection latency increases. Finally, during the third phase, more and more objects create direct social ties with the *native objects*. Thus, the average length of the chain of acquaintances connecting the detecting object with the closest *native* shortens more and more and the Discovery Latency decreases accordingly. However, this trend is softened by an increase in the sensing range. In

facts, with a larger detection range, the probability that a loud noise is detected by one of the few objects close to a *native* object obviously increases.

C. Detection Probability varying the Foreign objects placement scheme

The last set of simulations are finalized to verify that the performance described in section VI-B are not biased by the adopted placement scheme of the *foreign objects*. So far, we have assumed that *foreign objects* are randomly placed in the simulation area around a set of attraction points (e.g. shops, restaurants, squares). In the remainder of the paper we label this scheme as *"Attraction points"* scheme.

In the last simulation set, two additional placement schemes are considered. The first one, namely "*Random*" scheme, randomly places the *foreign objects* in the simulation area according to a uniform distribution. The second one places the *foreign objects* close to the main street which is parallel to the x axis. We label this scheme as *"Street"*. Figure 13 reports some exemplary outcomes of the object positioning schemes described.

When repeating our experiments by considering all the three placement schemes we have described, the performance slightly changes. As an example, Figure 14 shows the Detection Probability when considering 150 *foreign objects* by varying the node positioning and the detection range in (a) $75[m]$, and (b) $125[m]$. As observed in Figure 14, varying the positioning scheme implies a slight change in the performance metrics and the main trends we have highlighted so far remain essentially unaffected.

D. Hardware Utilization and Signaling Overhead

The proposed approach has been conceived as part of the activities of the DOMUS project[4]. Within the project, a proof of concept implementation of the proposed system is being developed to showcase the effectiveness of our proposal. So far, the main components of the proposed system, such as the *Social Object Container*, the *Social Senser*, and the *Virtual Object Container* have been implemented. A small scale testbed covering a wing of our campus have been already deployed. Hence, we are now able to present a first analysis of the resources required to implement the proposed approach on real devices. Firstly, we estimated the resources required to make a commercial device behave like a social object. It is worth recalling that the *iSapiens* platform achieves this goal by instantiating two software modules (See Section IV-D) for each object: the *Social Senser (SS)* and the corresponding *Social Object (SO)*. The *SS* is installed on the real device and periodically, every $T_{Disc}[s]$, overhears the transmission over the Wi-Fi and Bluetooth interfaces to search for other nearby social-enabled devices. The information collected during each discovery procedure is forwarded to the corresponding *SO* that resides in the edge and that is responsible to manage the SIoT Platform on behalf of the actual device. Of course, the resources required to execute the SS and the SO depend on the T_{Disc}. A smaller T_{Disc} implies a more precise discovery process but it has also a higher cost in terms of required resources. Figure 15 shows the *SS* and *SO* average CPU usage varying T_{Disc}. The values shown in Figure 15 are obtained by measuring the resource utilization of the two software modules running on different hardware platforms (e.g. raspberries, laptops, smart-phones) under the normal operational conditions. To make comparable the measure collected on different platforms, we expressed the CPU utilization in MHz by multiplying the percentage of utilization of a given CPU by the Clock of that CPU. We characterized the CPU usage for T_{Disc} varying from $15\,[m]$ to $[1ms]$. It is interesting to observe that the CPU usage of the *SO* remains low, in the order of units of MHz whatever T_{Disc}. Conversely, for what concerns the *SS*, the CPU usage is low, in the order of tens of MHz up a T_{Disc} of about $1\,[s]$, the usage increases abruptly up to about $1\,GHz$. We investigated this behavior on our prototypes and the clues we collected so far indicate that this increase is due to the fact that for small values T_{Disc} becomes comparable to the duration of the overhearing process and hence the *SS* tends to be always active thus using all the available resources. According to the formerly reported analysis we chose to configure the devices in our testbed to operate with $T_{Disc} = 5\,[s]$ to keep the resource consumption on the devices at acceptable levels. The data exchanged between the *SS* and the *SO* under the normal operational conditions constitute a Signaling Overhead required by our approach to allow the devices to act as social objects. Thus the second measurement campaign we conducted was aimed at characterizing this overhead traffic. To this aim, we utilized the well known protocol analyzer Wireshark[5] to capture all the packets exchanged between each pair of *SS* and *SO*. As shown in Figure 16, the signaling traffic is periodic being made by an average of $12\,pkt/min$ and generates an average rate of about $123b/s$. Such an amount of overhead traffic is small enough not to affect the network performance.

In order to assess the computational resources needed by the iSapiens middleware, we run a set of tests that consider different configuration settings in a distributed scenario. Tests were executed by varying the total number of agents in the system and the way such agents were deployed (see Table III). Each test was executed for a period of ten minutes, and the reported data are the averaged output of five executions. The computational node under observation is a Raspberry Pi 3 Mod.B, having a Quad-core ARMv8, 1.2GHz CPU and 1GB of RAM. All the deployed agents execute the same *cycle*, which consists in reading a value from a Virtual Object, propagating that value to all the other agents, and then sleeping for $2s$. This behavior was chosen because it is typical for a general monitoring application, where sensed data have to be shared with other applicative agents. Table III highlights the total CPU cost of a computational node and the cost of the execution of a single agent *cycle* in the same node. In the case no agents run in the system, we can evaluate the CPU cost of the iSapiens platform which is named as *baseline CPU consumption*. For each system configuration, the CPU

[4]The DOMUS Project site: http://www.distrettodomus.it/

[5]https://www.wireshark.org/

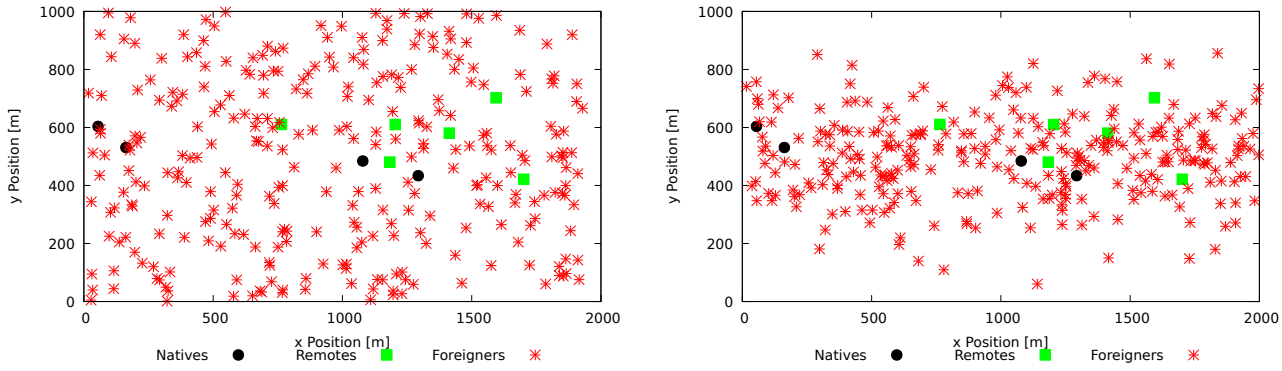


Fig. 13. Exemplary outcomes of the object positioning scheme we adopted: (a) *"Random"*, (b) *"Street"*, along the main street that traverses all the area at $y = 500[m]$.

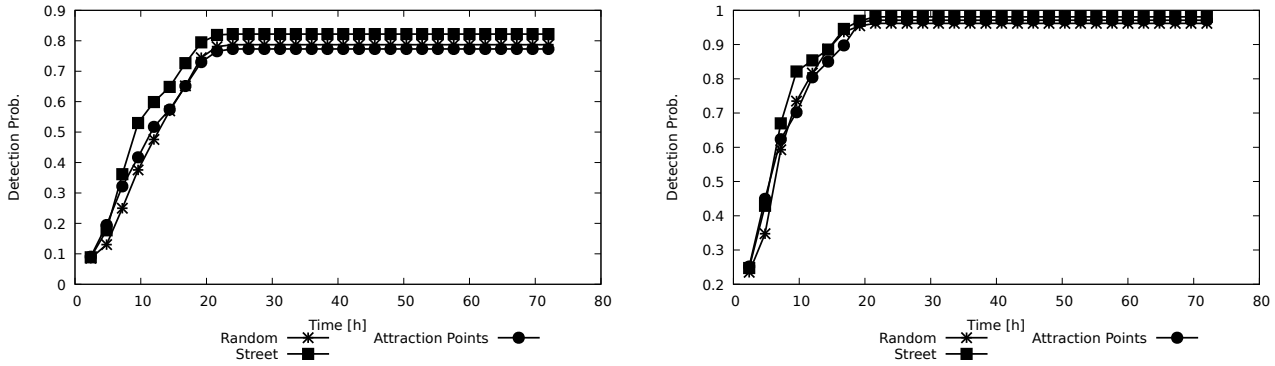


Fig. 14. Detection Probability considering 150 *foreign objects* varying the node positioning and the detection range in, (a) 75 $[m]$, and (b) 125 $[m]$

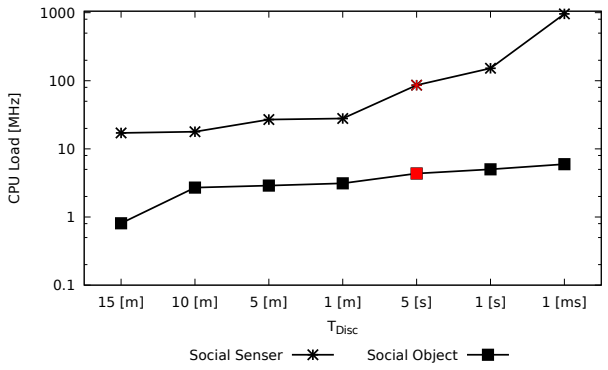


Fig. 15. *Social Senser* and *Social Object* average CPU usage

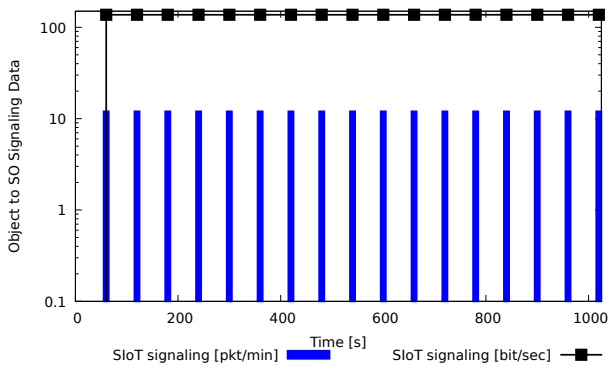


Fig. 16. *Social Senser* to *Social Object* signaling overhead after the *Object* power up a t=0s, analysis interval 60s, sensing interval 5s.

cost per cycle is computed by subtracting the CPU baseline to the total CPU cost on the node, and dividing it by the number of cycles executed by all the agents. From the table it emerges that the platform suits with the execution on low-power computational nodes, like the considered one. In case of all system agents running on a single node, the cost per cycle not significantly increases with respect to the growing number of agents and communications, that, in this case, are completely local. Counter-wise, when agents are split on multiple nodes and, thus, remote communication is required, then the Hz per cycle experience a significant increase as the number of agents and the remote communication increase. This means that, while designing an iSapiens application, it is of

TABLE III. RESOURCE UTILIZATION ON AN iSAPIENS NODE

#agents in the system	#agents on a node*	#cycles per agent	CPU (Hz) (node*)	CPU (Hz per cycle) (node*)
0	0	0	1200	—
1	1	300.3	2400	3.99
2	2	300.1	4800	5.99
4	4	301.1	8400	5.98
8	8	299.7	19200	7.55
16	16	298.9	42000	8.55
2	1	285.3	4800	12.61
4	2	226.2	8400	15.91
8	4	115.8	13200	25.90
16	8	47.3	21600	53.91

*Raspberry Pi 3 mod.B (Broadcom 2837 Quad-Core ARMv8 1.2GHz, 1GB RAM)

utmost importance to reduce as such as possible the remote communications. This is obviously eased by the adoption of an Edge Computing paradigm.

The results we got so far demonstrate that the proposed approach is compatible with off the shelf devices. We are now planning a wider deployment on a city block scale that will involve a large number of devices. We count on this wide-area deployment to obtain a greater insight on the proposed approach and to be able of studying the social interaction between the involved objects on a long time horizon in operational conditions.

VII. CONCLUSIONS

This paper has presented a novel approach to face the main challenges in the implementation of Large-scale Smart Environments such as the dynamism, the wide area coverage, the device heterogeneity, and the openness. Agent-based Edge Computing and Social Internet of Things are the pillars of the proposed approach. They respectively enable distributed and scalable computing and dynamic discovery of entities based on opportunistic relations. The iSapiens platform has been suggested as reference platform because its characteristics fit the introduced approach. The effectiveness of the proposed approach has been carefully assessed by describing a comprehensive use case related to a Smart Commercial Road and by carrying out an extensive simulation campaign to rate the effectiveness of SIoT in fostering object discovery and information collection processes in LSEs.

Ongoing and future work is devoted to:

- implement a field trial to get an experimental evaluation of the iSapiens platform at both agent and social level so as to enlarge the presented testbed to cover a whole city block in order to involve a larger number of devices and to generate a wider social structure. The goal is to provide a deeper investigation about the side-effects of on-the-fly entrance of new services and the resources needed;
- include a library of ready-to-use constructs to allow an explicit modeling of the behavior of all the entities introduced in the design approach;
- provide a library of ready-to-use interaction patterns which can be exploited for defining communication among entities;
- permit the specification of application dependent ontologies to be used for supporting communication among native and not-native objects.

We have realized a proof of concept implementation of the system and we have planned to enlarge our testbed to cover a city block scale so to involve a larger number of devices and to generate a wider social structure. As a consequence, a deep investigation about the side-effect of on-the-fly entrance of new services is considered to be carried out as future work.

ACKNOWLEDGMENT

This work has been partially supported by the "Smart platform for monitoring and management of in-home security and safety of people and structures" project that is part of the DOMUS District, funded by the Italian Government (PON03PE_00050_1).

REFERENCES

[1] Y. Oh, J. Han, and W. Woo, "A context management architecture for large-scale smart environments," *IEEE Communications Magazine*, vol. 48, no. 3, pp. 118–126, March 2010.

[2] F. Cicirelli, A. Guerrieri, G. Spezzano, A. Vinci, O. Briante, A. Iera, and G. Ruggeri, "An edge-based approach to develop large-scale smart environments by leveraging SIoT," in *Proc. of the 14th IEEE Intl. Conf. on Networking, Sensing and Control (ICNSC2017)*, May 2017.

[3] G. Aloi, G. Caliciuri, G. Fortino, R. Gravina, P. Pace, W. Russo, and C. Savaglio, "Enabling iot interoperability through opportunistic smartphone-based mobile gateways," *J. Netw. Comput. Appl.*, vol. 81, no. C, pp. 74–84, Mar. 2017. [Online]. Available: https://doi.org/10.1016/j.jnca.2016.10.013

[4] M. Daz, C. Martn, and B. Rubio, "State-of-the-art, challenges, and open issues in the integration of internet of things and cloud computing," *J. of Network and Computer Applications*, vol. 67, pp. 99 – 117, 2016.

[5] P. Garcia Lopez, A. Montresor, D. Epema, A. Datta, T. Higashino, A. Iamnitchi, M. Barcellos, P. Felber, and E. Riviere, "Edge-centric computing: Vision and challenges," *SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 5, pp. 37–42, Sep. 2015.

[6] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proc. of the First Edition of the MCC Workshop on Mobile Cloud Computing*, ser. MCC '12. New York, NY, USA: ACM, 2012, pp. 13–16. [Online]. Available: http://doi.acm.org/10.1145/2342509.2342513

[7] N. R. Jennings, "On agent-based software engineering," *Artificial Intelligence*, vol. 117, no. 2, pp. 277 – 296, 2000.

[8] L. Atzori, A. Iera, G. Morabito, and M. Nitti, "The Social Internet of Things (SIoT) - When Social Networks Meet the Internet of Things: Concept, Architecture and Network Characterization," *Comput. Netw.*, vol. 56, no. 16, pp. 3594–3608, Nov. 2012.

[9] F. Cicirelli, A. Guerrieri, G. Spezzano, A. Vinci, O. Briante, and G. Ruggeri, "iSapiens: A Platform for Social and Pervasive Smart Environments," in *Proc. of the IEEE World Forum on Internet of Things (WF-IoT2016) - Social Internet of Things Special Session*, 2016.

[10] F. Cicirelli, A. Guerrieri, G. Spezzano, and A. Vinci, "An edge-based platform for dynamic smart city applications," *Future Generation Computer Systems*, vol. 76, pp. 106 – 118, 2017.

[11] L. Atzori, A. Iera, and G. Morabito, "Understanding the Internet of Things: definition, potentials, and societal role of a fast evolving paradigm," *Ad Hoc Networks*, vol. 56, no. Supplement C, pp. 122–140, 2017.

[12] L. Yang, W. Li, M. Ghandehari, and G. Fortino, "People-centric cognitive internet of things for the quantitative analysis of environmental exposure," *IEEE Internet of Things Journal*, vol. PP, no. 99, pp. 1–1, 2017.

[13] O. Vermesan, P. Friess, P. Guillemin, H. Sundmaeker, M. Eisenhauer, K. Moessner, F. L. Gall, and P. Cousin, *Internet of Things Strategic Research and Innovation Agenda*, 2013.

[14] G. Fortino, A. Guerrieri, W. Russo, and C. Savaglio, "Middlewares for smart objects and smart environments: overview and comparison," in *Internet of Things Based on Smart Objects*. Springer, 2014, pp. 1–27.

[15] D. Cook and S. Das, *Smart Environments: Technology, Protocols and Applications*. Wiley-Interscience, 2004.

[16] O. Evangelatos, K. Samarasinghe, and J. Rolim, "Syndesi: A Framework for Creating Personalized Smart Environments Using Wireless Sensor Networks," in *Proc. of the 2013 IEEE Intl. Conf. on Distributed Computing in Sensor Systems*, ser. DCOSS '13. Washington, DC, USA: IEEE Computer Society, 2013, pp. 325–330.

[17] H. Vahdat-nejad, K. Zamanifar, and N. Nematbakhsh, "Context-Aware Middleware Architecture for Smart Home Environment," *Intl. Journal of Smart Home*, vol. 7, no. 1, 2013.

[18] F. Cicirelli, G. Fortino, A. Giordano, A. Guerrieri, G. Spezzano, and A. Vinci, "On the design of smart homes: A framework for activity recognition in home environment," *Journal of medical systems*, vol. 40, no. 9, p. 200, 2016.

[19] M. Amadeo, O. Briante, C. Campolo, A. Molinaro, and G. Ruggeri, "Information-centric networking for M2M communications: Design and deployment," *Computer Communications*, vol. 89, pp. 105–116, 2016.

[20] A. Socievole and S. Marano, "Exploring user sociocentric and egocentric behaviors in online and detected social networks," in *2012 2nd Baltic Congress on Future Internet Communications*, April 2012, pp. 140–147.

[21] A. Savidis and C. Stephanidis, "Distributed interface bits: dynamic dialogue composition from ambient computing resources," *Personal Ubiquitous Comput.*, vol. 9, no. 3, pp. 142–168, May 2005. [Online]. Available: http://dx.doi.org/10.1007/s00779-004-0327-2

[22] J. E. Bardram, R. E. Kjær, and M. Pedersen, "Context-Aware User Authentication - Supporting Proximity-Based Login in Pervasive Computing," in *UbiComp 2003: Ubiquitous Computing*, ser. Lecture Notes in Computer Science, A. Dey, A. Schmidt, and J. McCarthy, Eds. Springer Berlin Heidelberg, 2003, vol. 2864, pp. 107–123. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-39653-6_8

[23] M. Román, C. Hess, R. Cerqueira, A. Ranganathan, R. H. Campbell, and K. Nahrstedt, "A Middleware Infrastructure for Active Spaces," *IEEE Pervasive Computing*, vol. 1, no. 4, pp. 74–83, Oct. 2002. [Online]. Available: http://dx.doi.org/10.1109/MPRV.2002.1158281

[24] A. K. Dey, G. D. Abowd, and D. Salber, "A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications," *Hum.-Comput. Interact.*, vol. 16, no. 2, pp. 97–166, Dec. 2001. [Online]. Available: http://dx.doi.org/10.1207/S15327051HCI16234_02

[25] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of Things for Smart Cities," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, Feb 2014.

[26] M. Bain, "Sentilo - Sensor and Actuator Platform for smart Cities," https://joinup.ec.europa.eu/community/eupl/document/sentilo-sensor-and-actuator-platform-smart-cities, 2014, accessed: 2017-04-21.

[27] P. Ballon, J. Glidden, P. Kranas, A. Menychtas, S. Ruston, and S. Van Der Graaf, "Is there a need for a cloud platform for european smart cities?" in *eChallenges e-2011 Conf. Proc., IIMC Intl. Information Management Corporation*, 2011, pp. 1–7.

[28] Z. Khan, A. Anjum, K. Soomro, and M. A. Tahir, "Towards cloud based big data analytics for smart future cities," *Journal of Cloud Computing*, vol. 4, no. 1, p. 2, 2015.

[29] Y. Sun, H. Song, A. J. Jara, and R. Bie, "Internet of things and big data analytics for smart and connected communities," *IEEE Access*, vol. 4, pp. 766–773, 2016.

[30] F. Cicirelli, G. Fortino, A. Guerrieri, G. Spezzano, and A. Vinci, "Metamodeling of Smart Environments: from Design to Implementation," *Advanced Engineering Informatics (ADVEI)*, vol. 33, pp. 274 – 284, 2017.

[31] G. Lehmann, A. Rieger, M. Blumendorf, and S. Albayrak, "A 3-layer architecture for smart environment models," in *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE Intl. Conf. on*. IEEE, 2010, pp. 636–641.

[32] G. Fortino, A. Guerrieri, W. Russo, and C. Savaglio, "Towards a Development Methodology for Smart Object-Oriented IoT Systems: A Metamodel Approach," in *Systems, Man, and Cybernetics (SMC), 2015 IEEE Intl. Conf. on*, Oct 2015, pp. 1297–1302.

[33] L. Sanchez, L. Muñoz, J. A. Galache, P. Sotres, J. R. Santana, V. Gutierrez, R. Ramdhany, A. Gluhak, S. Krco, E. Theodoridis, and D. Pfisterer, "SmartSantander: IoT experimentation over a smart city testbed," *Comput. Netw.*, vol. 61, no. C, pp. 217–238, 2014.

[34] F. Gil-Castineira, E. Costa-Montenegro, F. Gonzalez-Castano, C. Lpez-Bravo, T. Ojala, and R. Bose, "Experiences inside the ubiquitous oulu smart city," *Computer*, vol. 44, no. 6, pp. 48–55, June 2011.

[35] H. Asl, A. Iera, L. Atzori, and G. Morabito, "How often social objects meet each other? Analysis of the properties of a social network of IoT devices based on real data," in *2013 IEEE Global Communications Conf. (GLOBECOM)*, Dec 2013, pp. 2804–2809.

[36] P. D. Meo, K. Musial-Gabrys, D. Rosaci, G. M. L. Sarnè, and L. Aroyo, "Using centrality measures to predict helpfulness-based reputation in trust networks," *ACM Trans. Internet Technol.*, vol. 17, no. 1, pp. 8:1–8:20, Feb. 2017. [Online]. Available: http://doi.acm.org/10.1145/2981545

[37] J. A. Golbeck, "Computing and applying trust in web-based social networks," Ph.D. dissertation, College Park, MD, USA, 2005, aAI3178583.

[38] J. Golbeck and J. Hendler, *Accuracy of Metrics for Inferring Trust and Reputation in Semantic Web-Based Social Networks*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 116–131. [Online]. Available: https://doi.org/10.1007/978-3-540-30202-5_8

[39] S. Valenzuela, N. Park, and K. F. Kee, "Is There Social Capital in a Social Network Site? Facebook Use and College Students' Life Satisfaction, Trust, and Participation," *Journal of Computer-Mediated Communication*, vol. 14, no. 4, pp. 875–901, 2009. [Online]. Available: http://dx.doi.org/10.1111/j.1083-6101.2009.01474.x

[40] M. Nitti, R. Girau, and L. Atzori, "Trustworthiness management in the social internet of things," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 5, 2014.

[41] B. Guo, D. Zhang, Z. Wang, Z. Yu, and X. Zhou, "Opportunistic IoT: Exploring the harmonious interaction between human and the Internet of Things," *Journal of Network and Computer Applications*, vol. 36, no. 6, pp. 1531 – 1539, 2013.

[42] D. B. Lange and M. Oshima, "Seven good reasons for mobile agents," *Communications of the ACM*, vol. 42, no. 3, pp. 88–89, 1999.

[43] G. Fortino, R. Gravina, W. Russo, and C. Savaglio, "Modeling and simulating internet-of-things systems: A hybrid agent-oriented approach," *Computing in Science Engineering*, vol. 19, no. 5, pp. 68–76, 2017.

[44] F. Cicirelli and L. Nigro, "Control centric framework for model continuity in time-dependent multi-agent systems," *Concurrency and Computation: Practice and Experience*, vol. 28, no. 12, pp. 3333–3356, 2016, cpe.3802. [Online]. Available: http://dx.doi.org/10.1002/cpe.3802

[45] C. Savaglio, G. Fortino, and M. Zhou, "Towards interoperable, cognitive and autonomic IoT systems: An agent-based approach," in *2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)*, Dec 2016, pp. 58–63.

[46] E. Cesario and D. Talia, "Distributed data mining patterns and services: an architecture and experiments," *Concurrency and Computation: Practice and Experience*, vol. 24, no. 15, pp. 1751–1774, 2012.

[47] S. Kosta, A. Mei, and J. Stefa, "Small World in Motion (SWIM): Modeling Communities in Ad-Hoc Mobile Networking," in *2010 7th Annual IEEE Communications Society Conf. on Sensor Mesh and Ad-Hoc Communications and Networks (SECON)*, June 2010.