2010 Fourth International Conference on Sensor Technologies and Applications

Smart City: An Event Driven Architecture for Monitoring Public Spaces with Heterogeneous Sensors

Luca Filipponi and Andrea Vitaletti
SAPIENZA Universitá di Roma
filipponi@dis.uniroma1.it
vitale@dis.uniroma1.it

Giada Landi and Vincenzo Memeo
Nextworks
g.landi@nextworks.it
v.memeo@nextworks.it

Giorgio Laura and Paolo Pucci
Elsagdatamat
Giorgio.Laura@elsagdatamat.com
Paolo.Pucci@elsagdatamat.com

***Abstract*—In this paper, we present the Smart City Architecture developed in the context of the ARTEMIS JU SP3 SOFIA project. It is an Event Driven Architecture that allows the management and cooperation of heterogeneous sensors for monitoring public spaces. The main components of the architecture are implemented in a testbed on a subway scenario with the objective to demonstrate that our proposed solution, can enhance the detection of anomalous events and simplify both the operators tasks and the communications to passengers in case of emergency.**

***Keywords*-Event Driven Architecture; Smart Spaces; Heterogeneous Sensors.**

I. Introduction

The SOFIA project is a three-year ARTEMIS project [1] started in January 2009 and involving partners from four different EU countries. The project is centered on the concept of smart environment, intended as an ecosystem of interacting objects - e.g., sensors, devices, appliances and embedded systems in general - able to self-organize, offer federated services and process or provide complex data. The main goal of the SOFIA project is to create a semantic interoperability platform and a selected set of vertical applications to form smart environments based on embedded systems. The key factors in these smart environments will be an open and distributed information storage and common search procedures for all the embedded systems, implemented with different specific technologies. In this vision, local mash-up applications will rely on open data and will use a range of devices. Two driving ideas reside behind this approach: (i) a connection between the real physical world and the information world that will enrich the user experience and (ii) an Interoperability Open Platform (IOP) that will promote innovation and will guarantee the future evolution of smart environments, both from a scientific/technological point of view and in terms of business. With these concepts, SOFIA fosters an Internet-like revolution in physical space, aiming to make ”embedded information” in the physical world available for smart services - connecting the physical world with the information world. Outcomes of the project encompass (i) new user interaction paradigms for interacting in smart environments, (ii) a common multi-vendor interoperability solution between many new and legacy heterogeneous devices and embedded systems, and (iii) application development schemes, ontologies and tools that can mobilize new developers for smart environments. The project addresses three application areas or “verticals” which represent different kinds of space in terms of scale, potential applications and services: smart personal spaces (e.g., car), smart indoor spaces (e.g., home and office), and smart city spaces (e.g., extended infrastructure and facilities like a subway station, shopping mall and so on).

Smart spaces concept in SOFIA. In SOFIA, a smart space is merely an information search extent where semantic information is presented in the form of an ontology graph or, equivalently, of triples (subject-predicate-object). Information within the smart space is stored in Semantic Information Brokers (SIBs). They provide a service interface for agents known as Knowledge Processors (KPs). KPs can access the information within the smart space by connecting to the SIB and invoking the operations offered by the Smart Space Access Protocol (SSAP) interface. Through SSAP operations, KPs can perform session management (join, leave) and transactional functions (insert, remove, update, query, subscribe, unsubscribe) both as producer and as consumer. Service discovery is accomplished using the mechanisms available at the service level (e.g., NoTA [4]). To enable their reactive behaviour the KPs make subscriptions and are then notified whenever specific events happen. This allows the KP to react appropriately and timely to changes in the smart space, while avoiding unnecessary communication [10].

This paper is organized as follows: section II describes in depth the Smart City Architecture with particular focus on the main building blocks and concepts. Section III presents a storyline and focuses on the flow of events in the Smart City Architecture. Finally, sections IV and V report conclusions and acknowledgements.

II. Smart City Architecture

The main goal of the Smart City architecture, is to provide a framework for the implementation of information services for monitoring public areas and infrastructures. The proposed solution, depicted in Figure 1, is an Event Driven

978-0-7695-4096-2/10 $26.00 © 2010 IEEE
DOI 10.1109/SENSORCOMM.2010.50

Architecture (EDA). In our context, an event is an observable change in the state of an IT system; it can be triggered both by real world events, such as presence detection, timeouts etc. or by internal events like the reception of a message (e.g., a command) or the completion of a task.

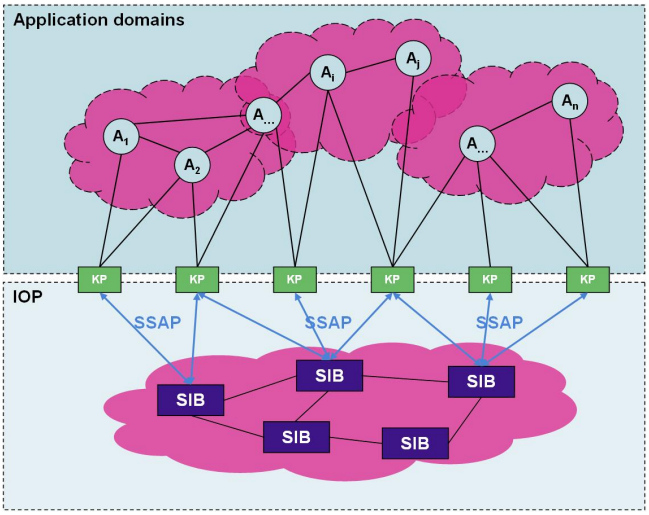

Figure 1. The Smart City high level architecture

A notification is produced by the observer of an event (e.g., a sensor) and describes that event (e.g., smoke detection) according to a suitable ontology. An ontology provides a shared vocabulary, which can be used to model a domain that is, the type of objects and/or concepts that exist, and their properties and relations (see [8], [7], [9] for an introduction to ontologies). Informally, such ontology delineates the shared language used to represent events among the different actors of the Smart City architecture.

The main building blocks of the Smart City architecture are Knowledge Processors(KPs) and Semantic Information Brokers(SIBs).

KPs are responsible to produce (insert/remove) and/or to consume (query/subscribe) notifications of events. Notifications describe the events as observed locally by KPs and the decision to publish a notification is a core part of the publisher KPs internal logic. Notifications are not addressed to a particular set of receivers, but are simply stored in the SIBs. SIBs implement a publish/subscribe paradigm, conveying notifications from producer KPs and delivering every published notification to all consumers KPs having registered matching subscriptions. A subscription, defines the classes of events of interest and allows the definition of filters to further select only relevant notifications. A consumer KP receives from a SIB notifications on the subscribed events. A set of suitable security policies allows the system to restrict the notifications only to the consumers that have subscribed with appropriate credentials.

The cooperation between KPs and SIBs, together with

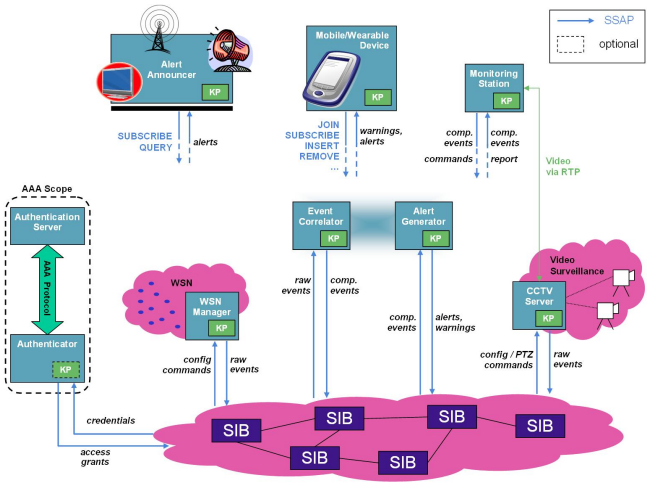

Figure 2. The Smart City full architecture

suitable routing strategies, creates the Interoperability Open Platform (IOP) as depicted in Figure1. IOP allows different application domains and sub-systems to inter-operate and to share information; applications can access shared informations stored in SIBs through their own KPs. Figure 2 depicts the full architecture of Smart City project, showing both Application an IOP levels, involving many heterogeneous systems that cooperate to achieve the Smart City objectives.

In the following, we present a short overview of the main Smart City functional modules.

A. *Data Aggregation and correlation*

The Smart City space can be considered an ubiquitous computing environment where clouds of sensors are installed in many different areas and provide information about the environment to interested devices, applications and users. The large number of sensors may potentially produce a large amount of data.

Composite events prevent subscribers from being overwhelmed by a large number of raw event publications by providing them with a higher-level abstraction. A composite event is published whenever a certain pattern of events occurs. This means that subscribers can subscribe directly to complex event patterns, instead of having to subscribe to all the raw events that make up the pattern.

Figure 3 shows a possible information flow where data are aggregated and correlated, increasing at each step their level of abstraction; from raw physical data (e.g., temperature) to high level event (e.g., fire detection). A cloud of heterogeneous sensors performs measurements of the environment (temperature, humidity, mechanical stress, etc.). These measurements are managed by the WSN subsystem, which publishes the raw events (i.e., the measurements itself) in the SIB. The WSN subsystem might also perform local computation such as filter-out some readings or aggregate

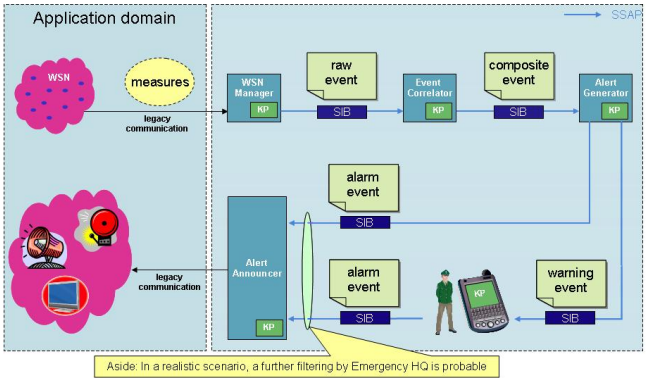

Figure 3. Subway Station events flow

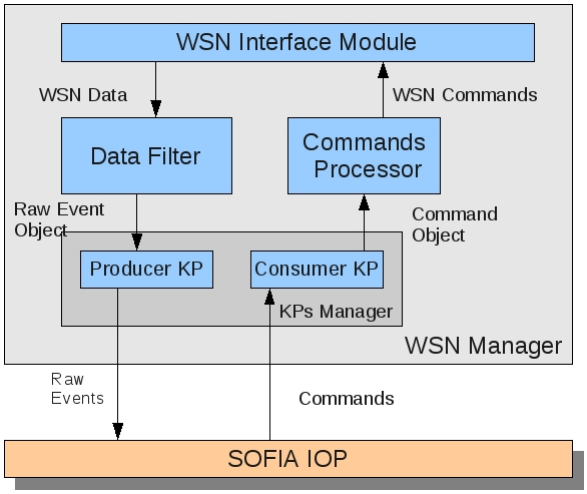

Figure 4. WSN Manager Architecture

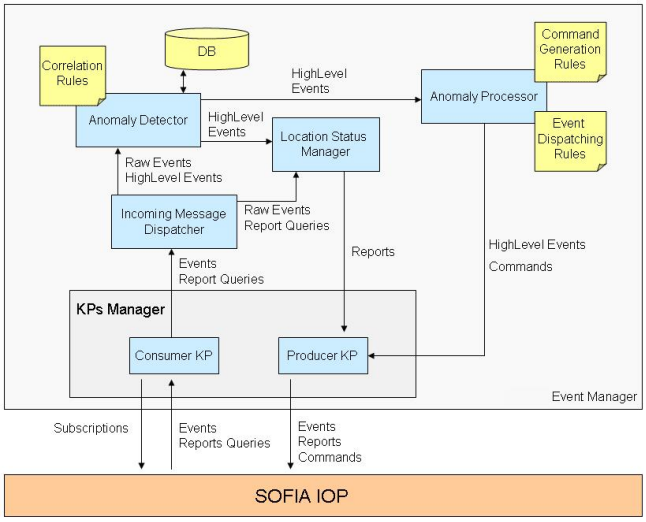

Figure 5. Event Manager Architecture

data. Raw and heterogeneous events (i.e., provided by different kind of sensors) are retrieved from the SIB by an Event Correlator which performs a further step of data aggregation, according to a suitable set of rules which exploit: a) temporal redundancy (measurements performed in consecutive periods of time), b) spatial redundancy (measurements performed by sensor displaced in near but different locations) and c) dimensional redundancy (measurement of a different physical dimension performed by different clouds of sensors - e.g., CO2 and temperature to detect a fire). The result of data correlation, enriched with a reliability index which gives a score to the likelihood of the detected phenomenon, is a composite event published in the SIB. Eventually, Alert Announcers inform the end-users of dangerous and/or critical situations, exploiting effective HCI techniques.

B. Wireless Sensor Networks

The WSN subsystem includes a WSN manager and one or more WSNs. The WSN manager implements both a consumer KP and a producer KP so that it can interact with the IOP in order to receive commands or dispatch raw events. Its architecture is depicted in Figure 4. The WSN interface module acts as a sink for all the underlay WSNs: it parses messages incoming from the WSNs, possibly preprocesses them and finally produces a suitable event. In particular, data incoming from the WSN are dispatched to the Data Filter module that performs the required preprocessing to generate a specific Raw Event that is published into the SIB by the Producer KP.

The WSN Manager can also receive commands notification through the Consumer KP. Commands may change the internal behaviour of the WSN Manager, tune some parameters of the filtering procedure, or instruct the WSNs with specific commands (e.g., increase the duty cycle, reduce the sampling period, enable/disable some sensing capabilities).

C. Event Manager

An Event Manager is a module in charge of merging and correlate events generated by raw data sources such as the WSN Manager or a CCTV subsystem. These events are correlated by means of some static criteria in order to detect anomalies in the current status of the monitored areas. The events generated by these anomalies are exchanged with other application layer modules through the SOFIA IOP. Furthermore, the Event Manager can also react to some specific events, generating configuration commands.

The general architecture of an Event Manager is shown in Figure 5. The interaction with the SOFIA IOP is handled by the KPs Manager, including both a Producer KP and a Consumer KP. The Producer KP publishes high-level events, reports and commands into the SIBs of the IOP. On the other hand, the Consumer KP is primarily concerned with the retrieving of all the events that are associated to the areas controlled by the event manager. The events retrieved by the

Consumer KP are translated into internal data structures and dispatched to the sub-modules dedicated to the processing. The Location Status Manager (LSM) is the sub-module that, takes into account both the events produced by the raw data sources and the high-level events detected by the Event Manager itself and elaborates, updates and stores all the information associated to each location included in the covered area. The Anomaly Detector (AD) sub-module is the functional entity that correlates the events received from the external modules according to the pre-configured static rules defining the association between the raw events and the corresponding high-level events. The resulting anomalies are stored in a database and forwarded to the Anomaly Processor (AP) for further elaboration. This sub-module generates the high-level events that are delivered to the external modules through the IOP and, if needed, a set of associated commands to manage or configure the raw data sources.

D. Alert announcers

The Alert Announcers are application layer entities able to notify the users of specific events (including warnings and alarms). Basically two main Alert Announcers have been proposed: the Public Alert System and the Event Mobile Broadcast System. The *Public Alert System* is realized with Conante's LumEnActive [2], a novel digital signage display that is based on steerable digital projection. The ability of LumEnActive to project into different directions has the advantage that the area to which the system projects to can be different for standard operation and alert announcements. The *Event Mobile Broadcast System* performs Event notifications from the IOP to users' mobile devices using standard multicast channels (e.g., Wi-Fi, Bluetooth, SMS).

In the following we present a storyline that describes the flow of events in this architecture, focusing on the interaction between the Wireless Sensor Network and the Event Manager.

III. Storyline Subway station

A subway station is a public and densely populated area, that must be constantly controlled through several monitoring systems. Existing solutions in this area are mainly based on dedicated and independent monitoring modules unable to cooperate. Our objective is to demonstrate how our unified solution, enabled by the SOFIA infrastructure, can enhance the detection of anomalous events and simplify both the operators tasks and the communications to passengers in case of emergency.

A. Testbed overview

Our testbed includes two types of raw data sources: a WSN Manager able to manage multiple sensor networks (i.e., temperature and humidity sensors, smoke sensors, presence sensors etc.) and a CCTV server connected to several distributed video cameras. The WSN Manager produces periodical reports about the environmental condition of the monitored areas and is also able to notify anomalous measurements in case of values exceeding some given thresholds. On the other side the CCTV server is able to process the video captured from the video cameras in order to perform motion detection or recognize the presence of a particular pattern in the video images. Both the WSN Manager and the CCTV server can be dynamically configured to enable or disable the generation of specific types of event or to modify the values of the thresholds used to detect the anomalies.

The raw data produced by both the WSN manager and the CCTV server are jointly processed by the Event Manager, that is able to merge and correlate them in order to detect the anomalous conditions and produce some warning or alert events. Exploiting its broader vision about all the raw data sources, the event manager can generate high-level events with an higher efficiency and a reduced error (e.g., false positives, false negatives). In our testbed we show the correlation of the raw data for the automatic recognition of the events like fire detection and presence detection, two relevant use-cases for the selected scenario. In particular, the presence detection alarm is generated by the Event Manager in case of combined recognition of three raw events generated by both the raw source modules: motion detection from the CCTV server, presence and noise detection from the dedicated sensor networks controlled by the WSN manager. An important feature of our system is the easy (or automatic in some circumstances) activation and deactivation of a specific type of detection capability for selected regions in the covered area. This is a reasonable requirement for the subway station scenario, where the presence detection must be enabled only in specific private areas in pre-configured time slots or in public areas if they have been closed and evacuated. In the latter case, the system is able to automatically activate the presence detection procedures in the required areas whenever an evacuation order is delivered through the IOP.

The following subsections presents in details the functionalities of the WSN Manager and the Event Manager (introduced in the previous sections) in the selected scenario.

B. WSN Manager

The general structure of the WSN Manager has been described in section II-B: the consumer KP receives commands via the IOP and the producer KP delivers raw data events generated by the underlay WSNs. Here we focus on the underlay sensor network deployed in the subway station scenario. Our testbed is made of TelosB [3] sensor nodes running TinyOS [6] code and equipped with temperature, humidity,light and infrared sensors. In particular infrared sensors are useful for presence detection events, while temperature sensors are used for fire detection purposes. The

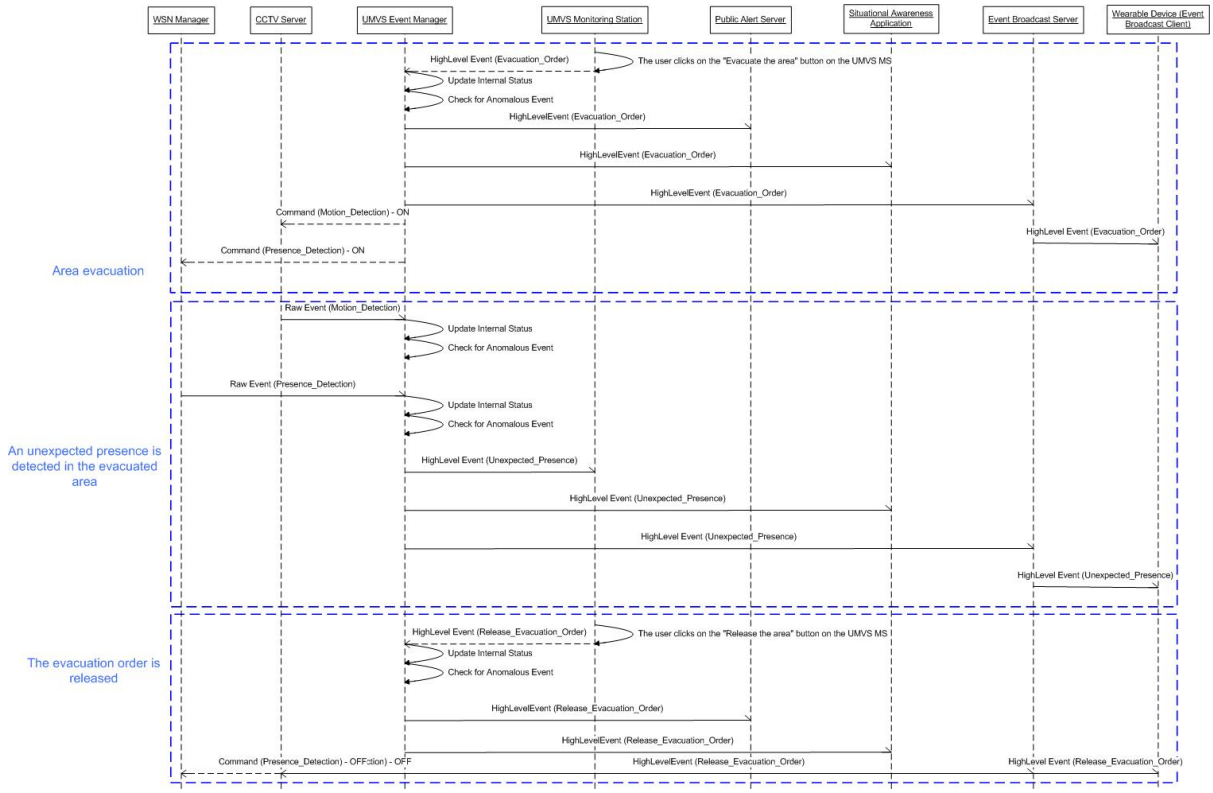

Figure 6. The Message Sequence Chart for the testbed

current network is a star topology made of a *Sink* and many *Collectors* that monitor the environment.

When a new Collector node is turned on in the network, it starts a JOIN procedure for discovering its Sink node. Once the join procedure is successfully completed, a configuration procedure is started by the Sink to configure the behaviour of the new node. Which sensors are sampled, the sampling period and interval are all configurable options that the WSN Manager can easily set at execution time, as well as the working mode and the duty cycle of the Collector nodes. While the Sink node is always awake, two working modes are provided for the Collectors:

1) Real-time: the node follows a given duty cycle using the low power listening communication (LPL). When a communication between the Sink and the Collector node is required, the Sink node sends out a sequence of preambles for waking up the target node (or all the nodes in case of broadcast transmissions).
2) Periodic: Collectors communicate with the sink only when a transmission interval is scheduled. In each transmission interval, Collectors deliver all the sampled data to the sink. The sink acknowledges the reception and at the same time can possibly piggyback a configuration command.

The Collector application periodically takes some sensor readings, logs them to the flash, and waits the next transmission interval to deliver the collected samples to the Sink. Whenever the Sink node receives any packet from the Collector application, it forwards these packets back to the WSN Interface module in fig 4 for further processing. We stress that, a fine tuning of the duty cycles can allow to reach a suitable trade-off between network reactivity (i.e., low data latency) and energy savings.

C. Event Manager

The Event Manager retrieves the raw events generated by the CCTV server and the WSN manager through the IOP; in particular, data are automatically collected by the Event Manager KP through persistent subscriptions with the SIBs or are retrieved on-demand with direct queries. These raw events are further processed and inter-correlated according to the criteria described in the semantic model and, as result, the module produces the associated composite events able to describe the current status of the subway station environment or the detected anomalies. The Event Manager also generates automatic commands for the underlying sub-modules in order to enable/disable the video analysis for motion detection or the WSN procedures to detect for a given event, and to modify the configuration of the threshold values for the WSN Managers. These commands can be

dynamically generated as a consequence of specific events (triggered by an operator launching an alarm or detected by the system itself) and are described in the semantic model together with their relationship to the originating events. The semantic model also describes the correlation rules for the processing of the raw events (i.e., temperature, smoke, noise, motion) and the generation of the corresponding high-level events (i.e., fire, presence). In particular, the Event Manager analyses the events included in a given time interval in order to merge them and produce a set of high-level events with an associated Reliability Index (RI), that describes the likelihood of each event. An alarm is generated whenever the RI is bigger than a given threshold. The computation of the RI takes into account multiple parameters of the raw events that can be associated to a potential high-level event, such as: number of raw events of a given type detected by a specific source module; number of different source modules that have detected the same raw event type; reliability index associated to the raw events; time intervals between the raw events; number of different raw events that have been detected by the modules; presence of isolated bursts of raw events (they could indicate just a peak value or a temporary anomaly) raw events order; presence of a specific event. The Event Manager could also actively ask for further data from one or more source modules whenever the required information allows the module to take more efficient decision about the nature of a detected high level event. The Message Sequence Chart for the testbed is shown in figure 6, where the interactions between the modules KPs and the SIBs are omitted in order to have a clear view of the overall interaction among the modules. In any case all the message exchanges shown in the figure are actually mediated through the IOP. In the first phase an evacuation order is generated from the Monitoring Station, triggered by an operator, and it is distributed towards a variety of subscribed client applications through the IOP communication facilities. Since this is an alarm addressed to both the operators and the public, it is received also by the client applications running on the passengers mobile phones and by the subway station Public Alert System. The Event Manager automatically enables the presence detection procedures for the evacuated areas on the proper WSN manager and the motion detection on the CCTV server. In the second phase an unexpected presence in a closed area is recognized by both the WSNs and the video surveillance system. The event manager detects the alert, and publishes an alarm event through the IOP to the enabled client applications. In this case, it is an alarm addressed only to the operators, so it is received only by the associated reduced subset of devices. In the last phase the evacuated areas are released, through a manual intervention on the Monitoring Station. The related messages are again distributed by the IOP towards the client applications associated to the operators and to the public; finally the Event Managers disables all the presence detection procedures on the source modules.

IV. Conclusion and Future Work

We presented a unified system for monitoring and managing a public space, based on a middleware managing heterogeneous data sources through an Interoperability Open Platform (IOP) that uses a semantic description framework and access policies. The IOP platform makes the integration of heterogeneous sensors and subsystems an easier task, and, enforcing it by means of semantic support, allows the formation of smart spaces. The IOP is openly developed in the SOFIA consortium and a first version is accessible from sourceforge [5].

V. Acknowledgments

This work was funded by the European Commission, within the framework of the ARTEMIS JU SP3 SOFIA project [1]. The authors would like to thank all project partners who contributed to the definition and implementation of SOFIA Open Innovation Platform.

References

[1] Artemis ju sp3 sofia project. http://sofia-project.eu/. Last accessed, April 2010.

[2] Conante lumenactive. http://www.lumenactive.de/. Last accessed, April 2010.

[3] Crossbows telosb mote platform. http://www.xbow.com/Products/productdetails.aspx?sid=252. Last accessed, April 2010.

[4] Nota world - open architecture initiative. http://www.notaworld.org/. Last accessed, April 2010.

[5] Smart-m3. http://sourceforge.net/projects/smart-m3/. Last accessed, April 2010.

[6] Tinyos. http://www.tinyos.net/. Last accessed, April 2010.

[7] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, March 2003.

[8] Ronald Brachman and Hector Levesque. *Knowledge Representation and Reasoning (The Morgan Kaufmann Series in Artificial Intelligence)*. Morgan Kaufmann, May 2004.

[9] Asuncion Gomez-Perez, Oscar Corcho, and Mariano Fernandez-Lopez. *Ontological Engineering : with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web. First Edition (Advanced Information and Knowledge Processing)*. Springer, July 2004.

[10] Alessandra Toninelli, Susanna Pantsar-Syväniemi, Paolo Bellavista, and Eila Ovaska. Supporting context awareness in smart environments: a scalable approach to information interoperability. In *M-PAC '09: Proceedings of the International Workshop on Middleware for Pervasive Mobile and Embedded Computing*, pages 1–4, New York, NY, USA, 2009. ACM.